CPC Loss

November 23, 2021

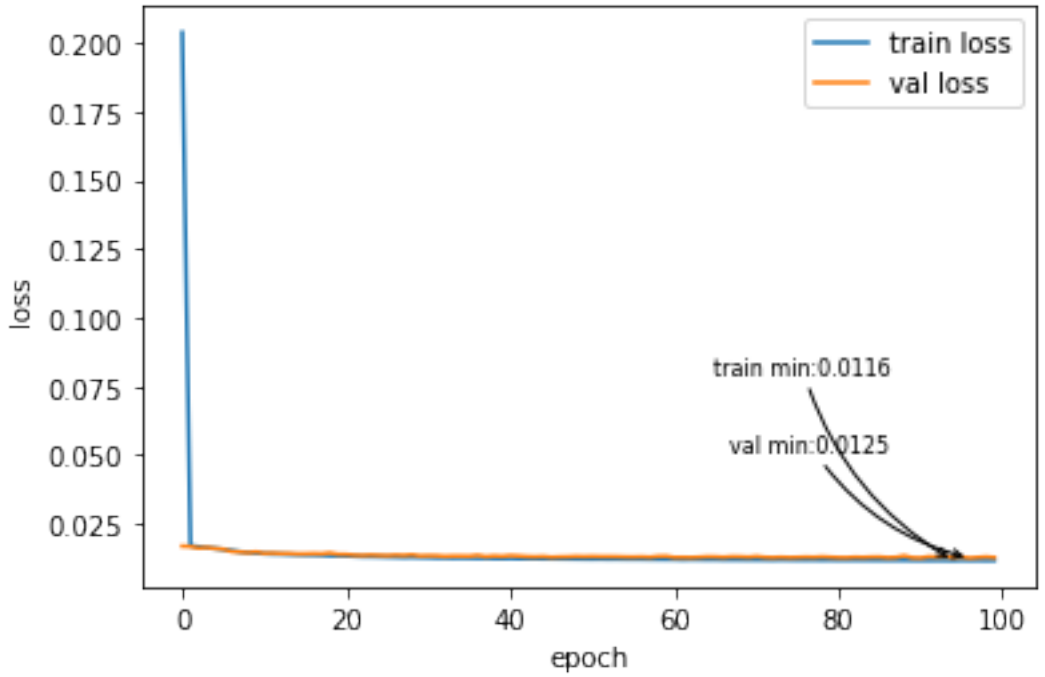
[ ]:
```
import numpy as np
import math
import matplotlib.pyplot as plt
import pickle
import torch
import os
```

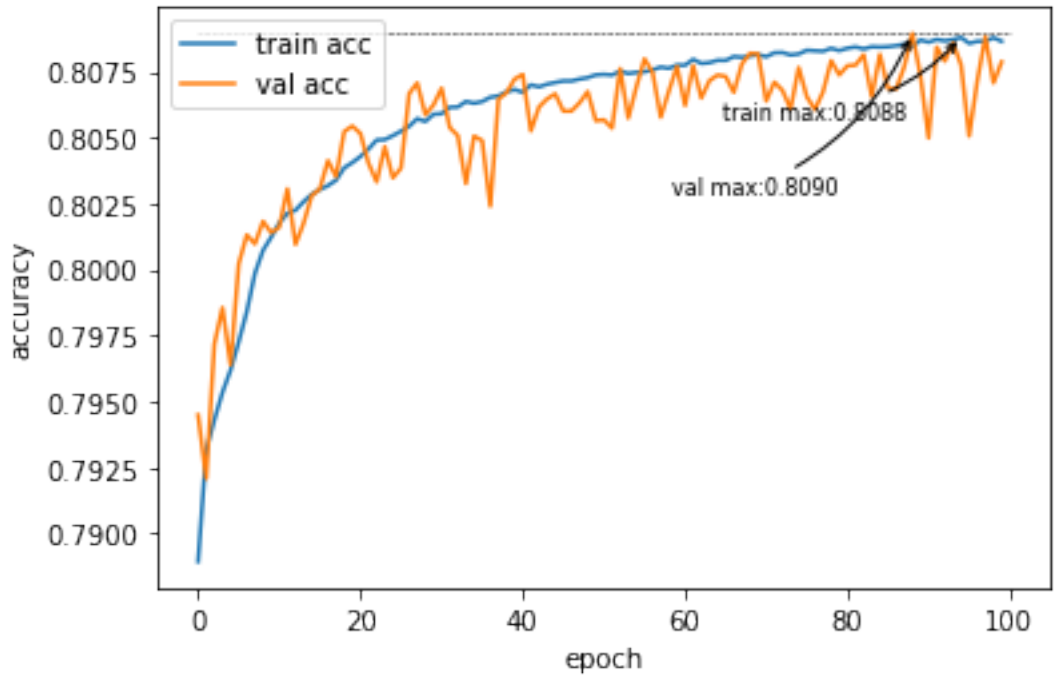
[2]:
```
def load_many_pickles(paths, multi=False):
    lists = []
    for path in paths:
        loss_path = os.path.join(path, 'losses.pkl')
        acc_path = os.path.join(path, 'accuracies.pkl')
        lists.append(_load_pickles(loss_path, acc_path, multi))
    return [np.concatenate(a, axis=0) for a in zip(*lists)]

def load_pickles(path, multi=False):
    loss_path = os.path.join(path, 'losses.pkl')
    acc_path = os.path.join(path, 'accuracies.pkl')
    return _load_pickles(loss_path, acc_path, multi)

def _load_pickles(loss_path, acc_path, multi):
    with open(loss_path, 'rb') as f:
        loaded = pickle.load(f)
    with open(acc_path, 'rb') as f:
        loaded2 = pickle.load(f)
    if multi: #potential different handling for multiple losses accuracies
        train_losses = loaded[0]
        val_losses = loaded[1]
        train_acc = loaded2[0]
        val_acc = loaded2[1]
    else:
        train_losses = [l for l in loaded[0]]
        val_losses = [l for l in loaded[1]]
        train_acc = [l for l in loaded2[0]]
        val_acc = [l for l in loaded2[1]]
    return train_losses, val_losses, train_acc, val_acc
```

```
def plot(train_losses, val_losses, train_acc, val_acc):
    plt.plot(train_losses, label='train loss')
    plt.plot(val_losses, label='val loss')
    plt.annotate('train min:%.4f'%np.min(train_losses),
                 xy=(np.argmin(train_losses), np.min(train_losses)),␣
 ↪xycoords='data',
                 xytext=(-90, 70), textcoords='offset points', fontsize=8,
                 arrowprops=dict(arrowstyle="->", connectionstyle="arc3,rad=.2"))
    plt.annotate('val min:%.4f'%np.min(val_losses),
                 xy=(np.argmin(val_losses), np.min(val_losses)), xycoords='data',
                 xytext=(-90, 40), textcoords='offset points', fontsize=8,
                 arrowprops=dict(arrowstyle="->", connectionstyle="arc3,rad=.2"))
    plt.xlabel("epoch")
    plt.ylabel('loss')
    plt.legend()
    plt.show()
    plt.plot(train_acc, label='train acc')
    plt.plot(val_acc, label='val acc')
    plt.hlines(np.max(val_acc), 0, len(train_acc), linestyle="dashed",␣
 ↪color='black', linewidth=0.5)
    plt.annotate('train max:%.4f'%np.max(train_acc),
                 xy=(np.argmax(train_acc), np.max(train_acc)), xycoords='data',
                 xytext=(-90, -30), textcoords='offset points', fontsize=8,
                 arrowprops=dict(arrowstyle="->", connectionstyle="arc3,rad=.2"))
    plt.annotate('val max:%.4f'%np.max(val_acc),
                 xy=(np.argmax(val_acc), np.max(val_acc)), xycoords='data',
                 xytext=(-90, -60), textcoords='offset points', fontsize=8,
                 arrowprops=dict(arrowstyle="->", connectionstyle="arc3,rad=.2"))
    plt.xlabel("epoch")
    plt.ylabel('accuracy')
    plt.legend()
    plt.show()
```

0.1 Losses for multi target

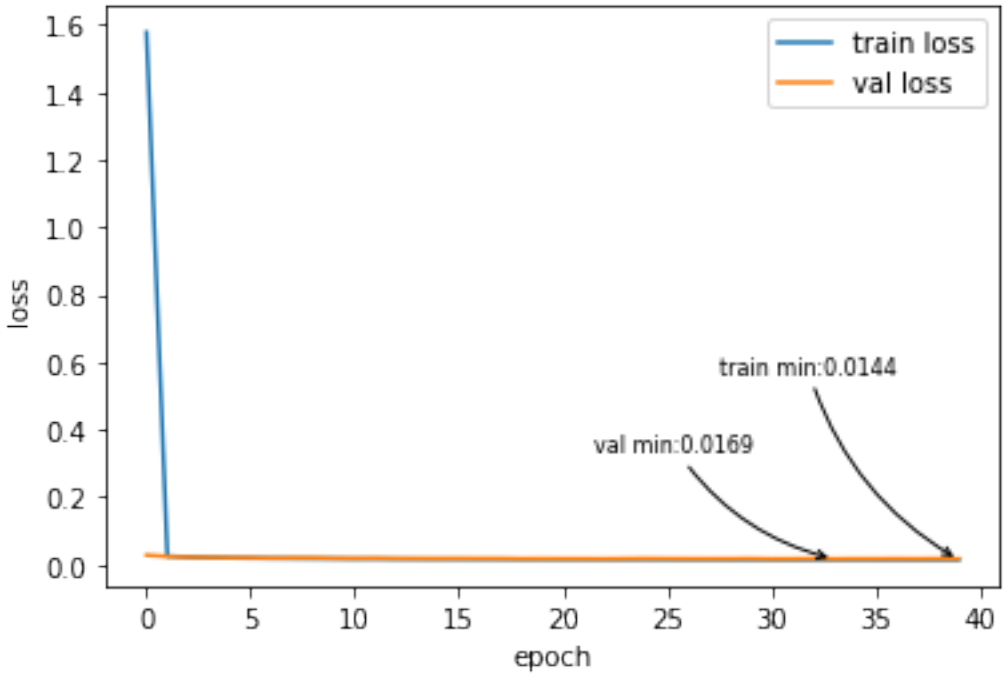
[4]:
```
#Downstream on cpc(baseline dataset) batch_size 16->64 windowslength 236
"""
latents used as prediction method (flatten + double linear no RELU inbetween)␣
 ↪(relu made it smoother but accuracy was worse)
layers frozen
"""
train_losses, val_losses, train_acc, val_acc = load_many_pickles(['/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/01_02_21-18'])
plot(train_losses, val_losses, train_acc, val_acc)
```

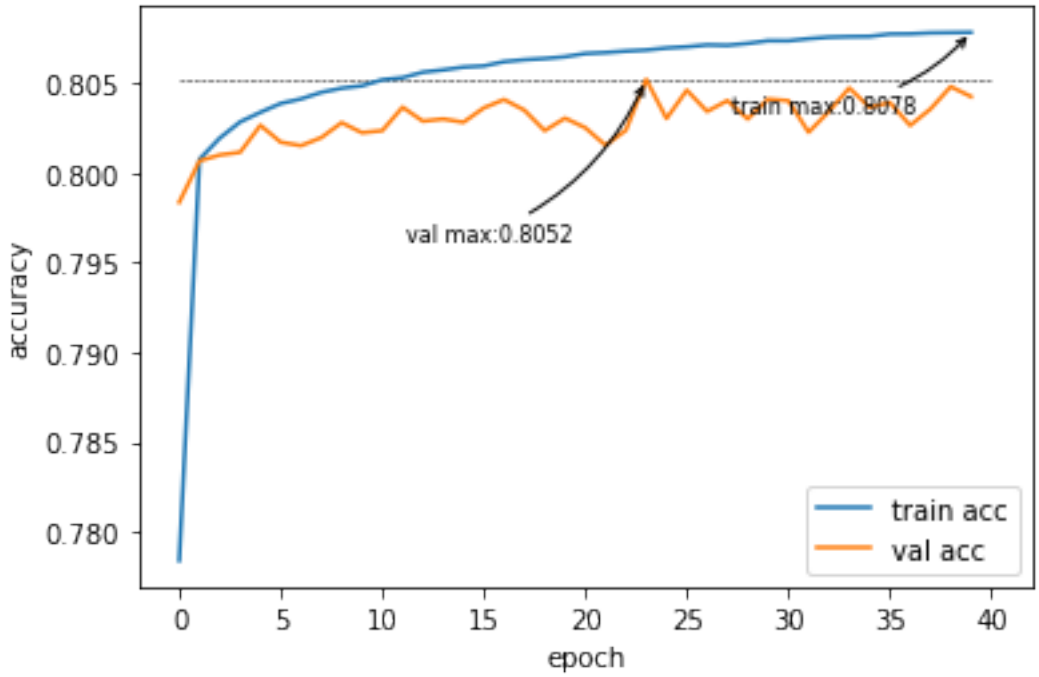




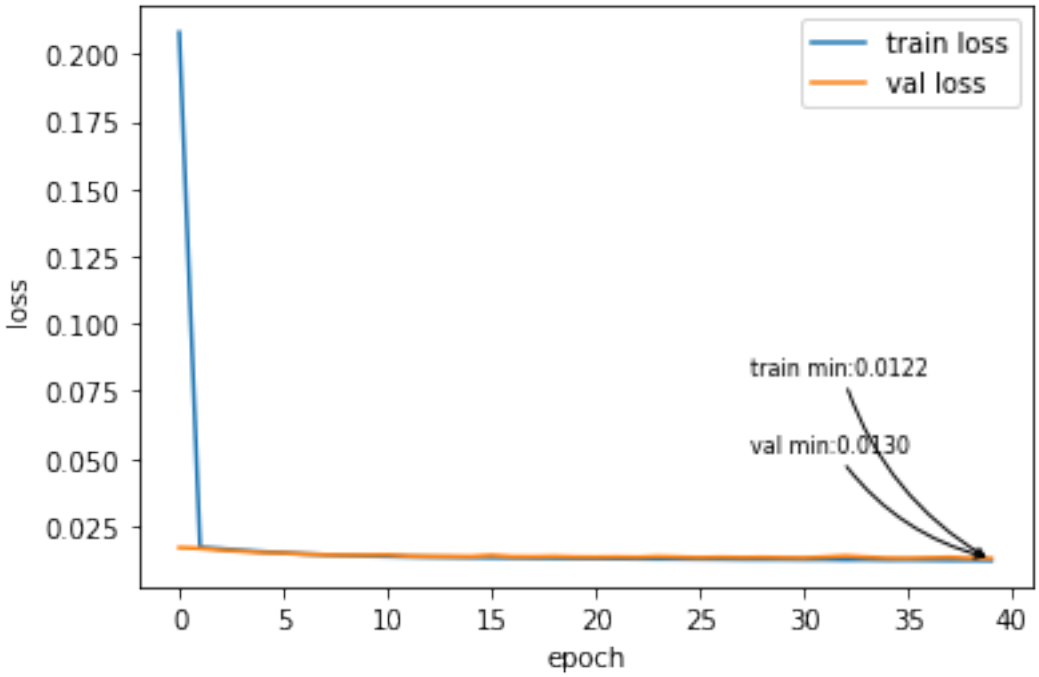
[43]:
```
#Downstream on cpc(baseline dataset) batch_size 16->64 windowslength 236
"""
```

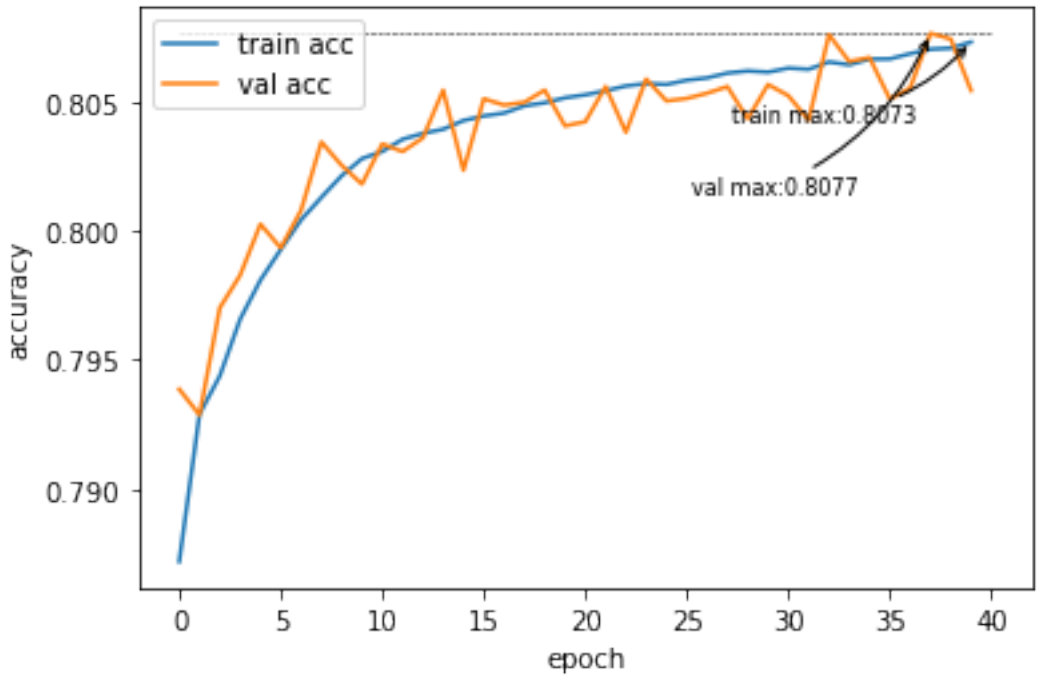
```
Domnt run again
latents used as prediction method (flatten + double linear no RELU inbetween)␣
↪(relu made it smoother but accuracy was worse)
layers frozen
"""
train_losses, val_losses, train_acc, val_acc = load_many_pickles(['/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/01_02_21-18'])
plot(train_losses, val_losses, train_acc, val_acc)
```





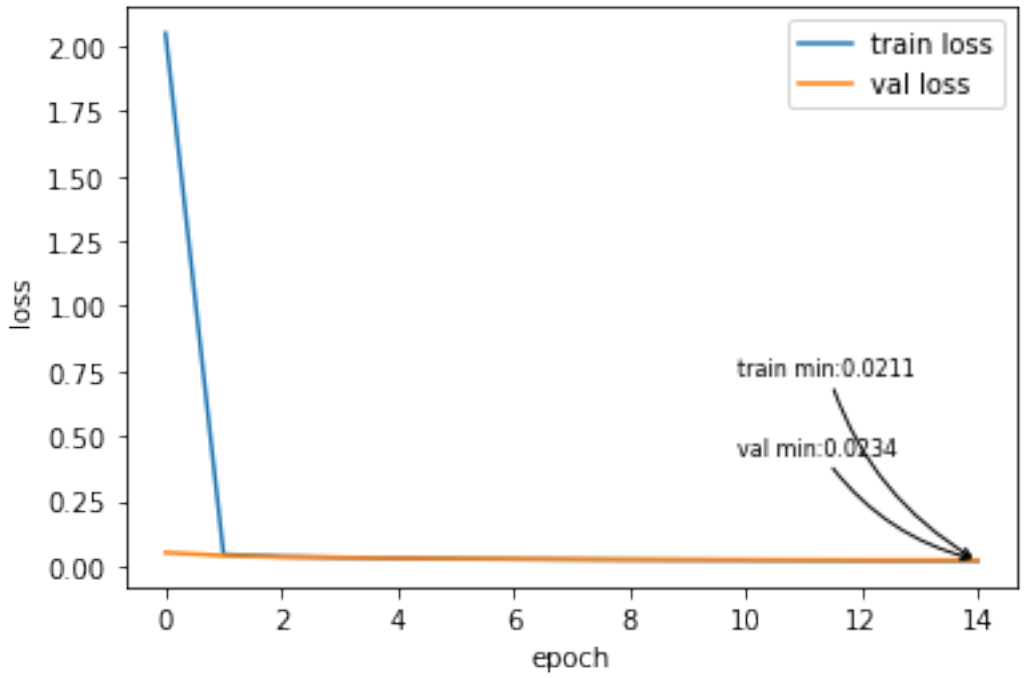
[39]:
```
#Downstream on cpc (baseline dataset) batch_size 16->64 windowslength 236
"""
layers unfrozen
"""
train_losses, val_losses, train_acc, val_acc = load_many_pickles(['/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/01_02_21-17'])
plot(train_losses, val_losses, train_acc, val_acc)
```

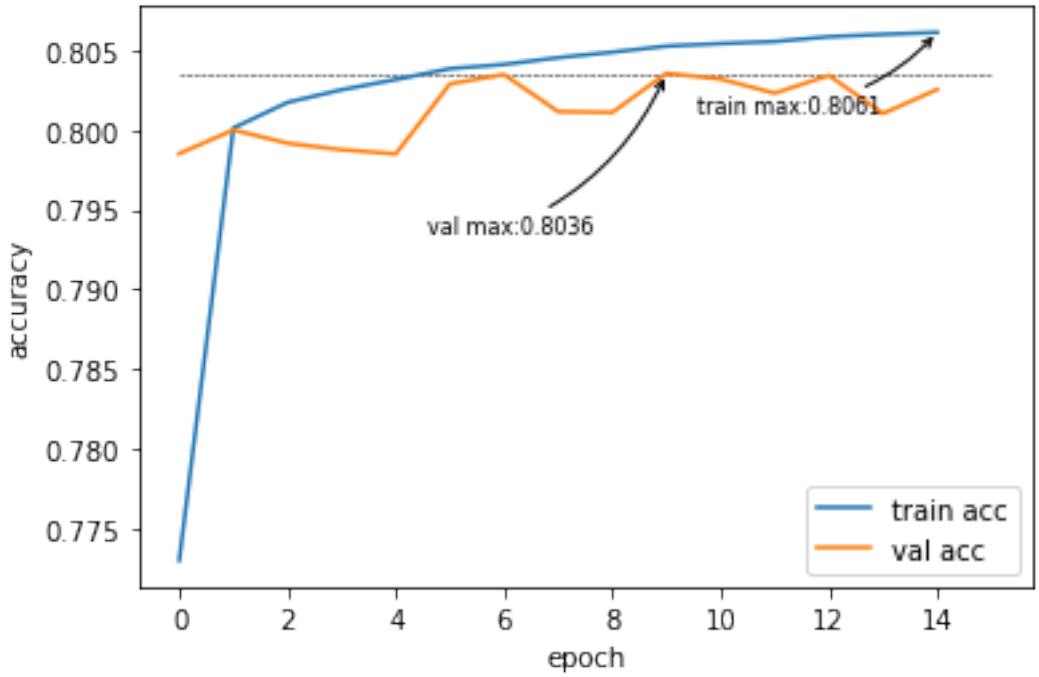




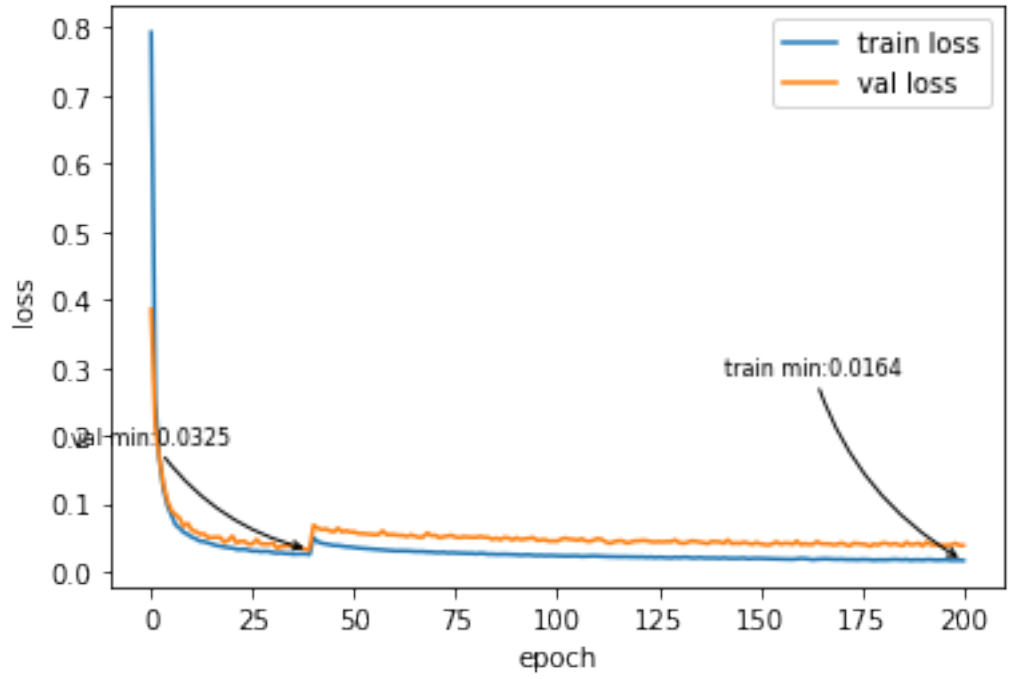
[37]:
```
#Downstream on cpc(baseline dataset) batch_size 16->64 windowslength 236
"""
```

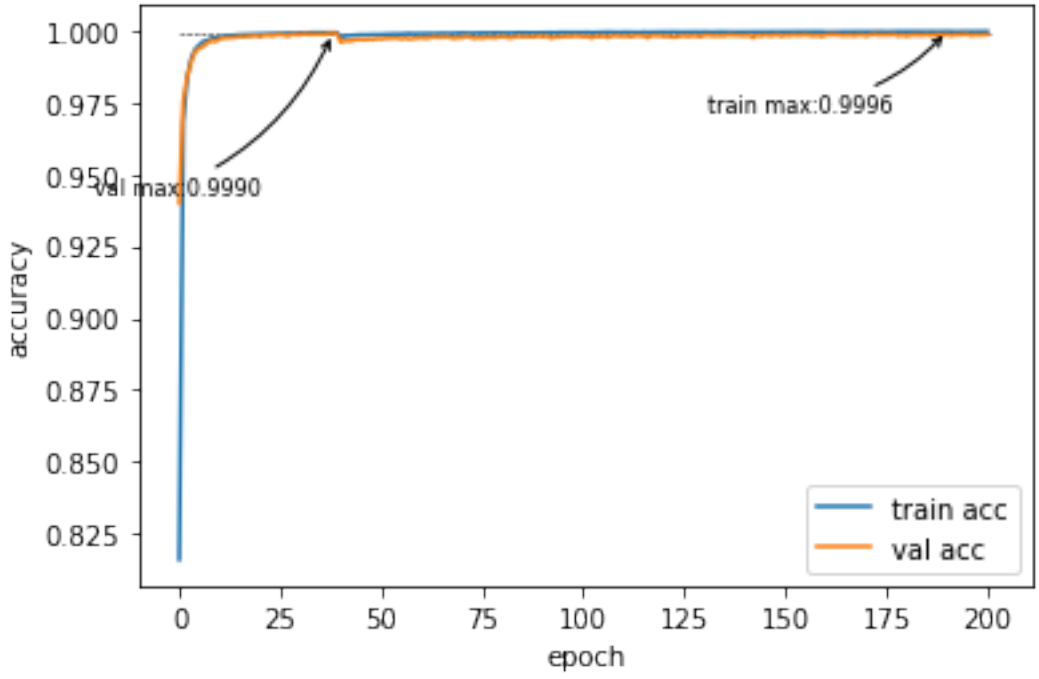
```
latents used as prediction method (flatten + linear)
layers frozen
"""
train_losses, val_losses, train_acc, val_acc = load_many_pickles(['/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/01_02_21-16'])
plot(train_losses, val_losses, train_acc, val_acc)
```





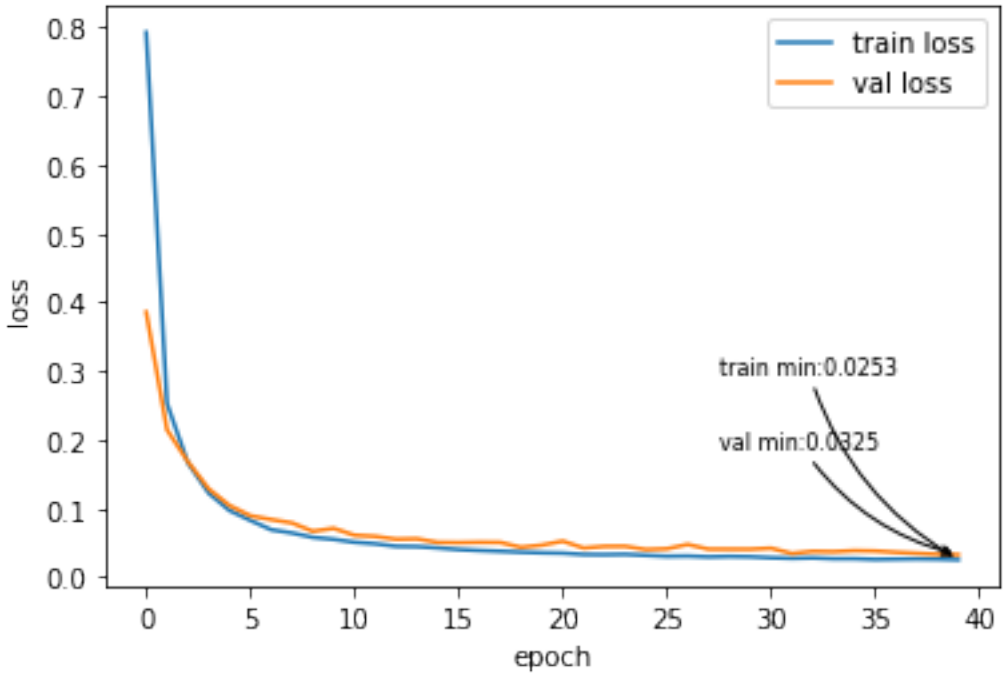
[29]:
```
#batch_size 16->64 windowslength 236
"""
CPC intersect sadly smaller batch size
Properly working? Cant be compared to old cpc directly (measure downstream␣
 ↪performance instead)
init hidden on each batch of data
"""
train_losses, val_losses, train_acc, val_acc = load_many_pickles(['/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/31_01_21-20', '/home/
 ↪julian/Downloads/Github/contrastive-predictive-coding/models/01_02_21-11'])
plot(train_losses, val_losses, train_acc, val_acc)
```

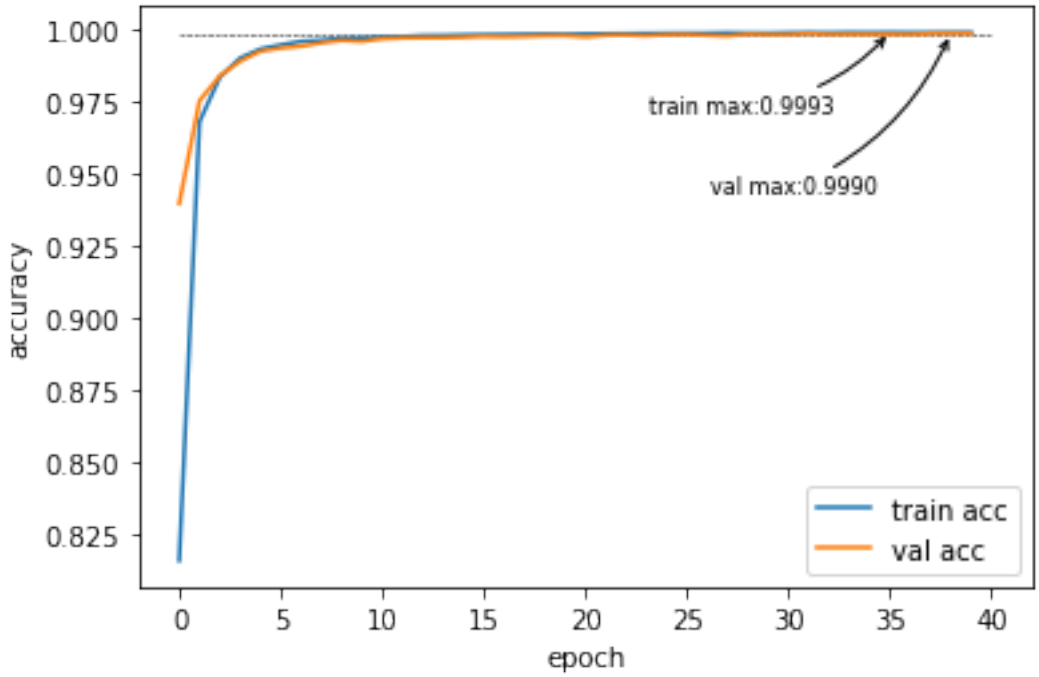




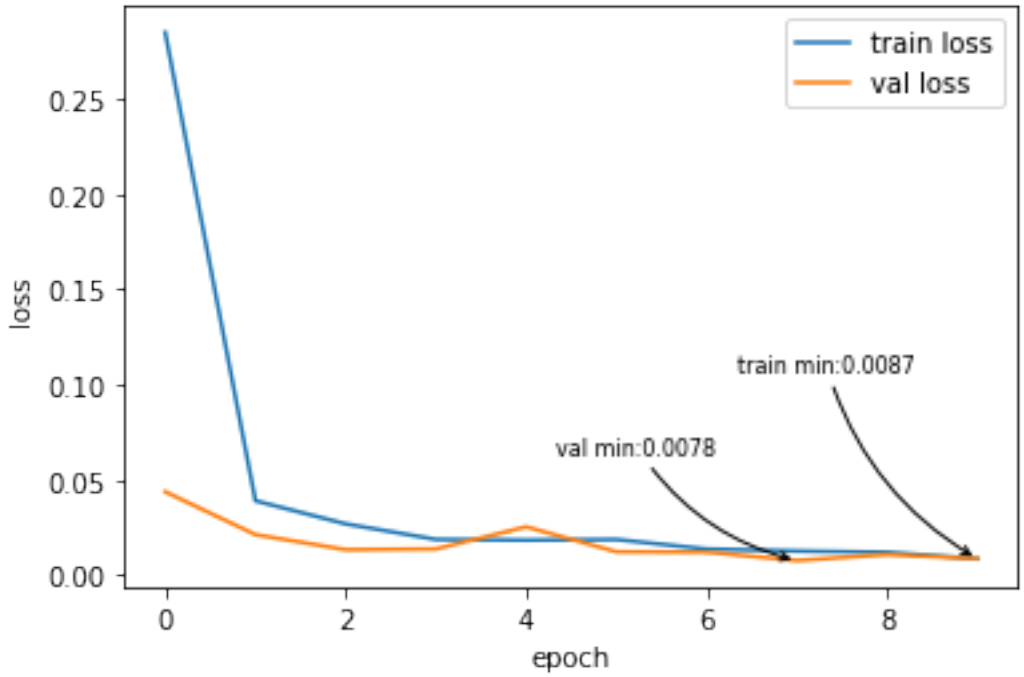
[3]:
```
#batch_size 16 windowslength 236
"""
```

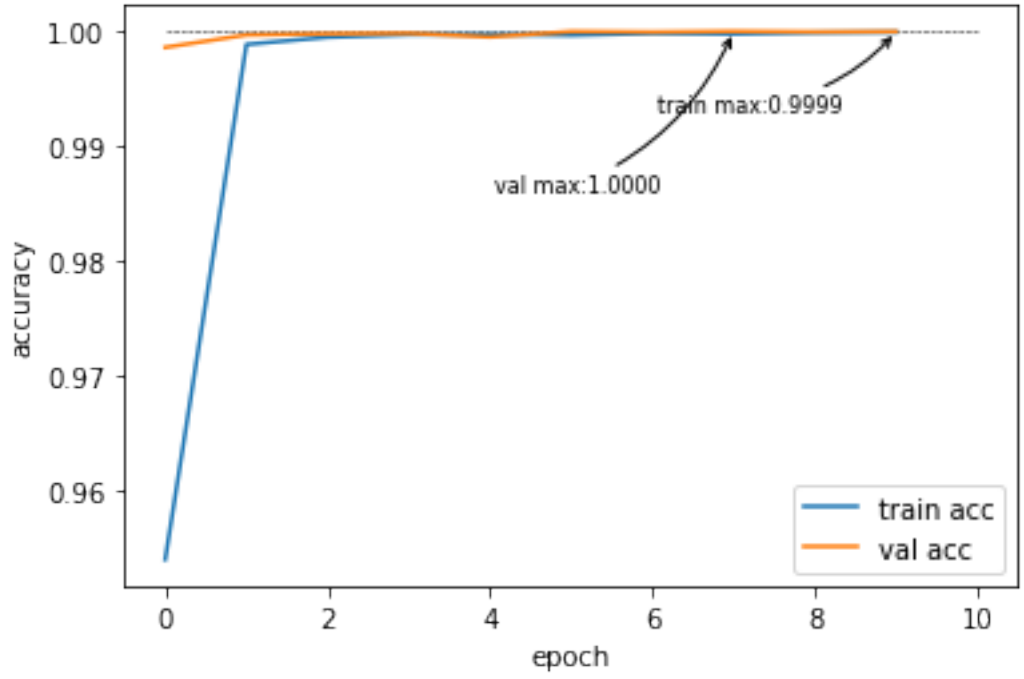
```
CPC intersect sadly smaller batch size
Properly working? Cant be compared to old cpc directly (measure downstream␣
 ↪performance instead)
init hidden on each batch of data
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/31_01_21-20')
plot(train_losses, val_losses, train_acc, val_acc)
```





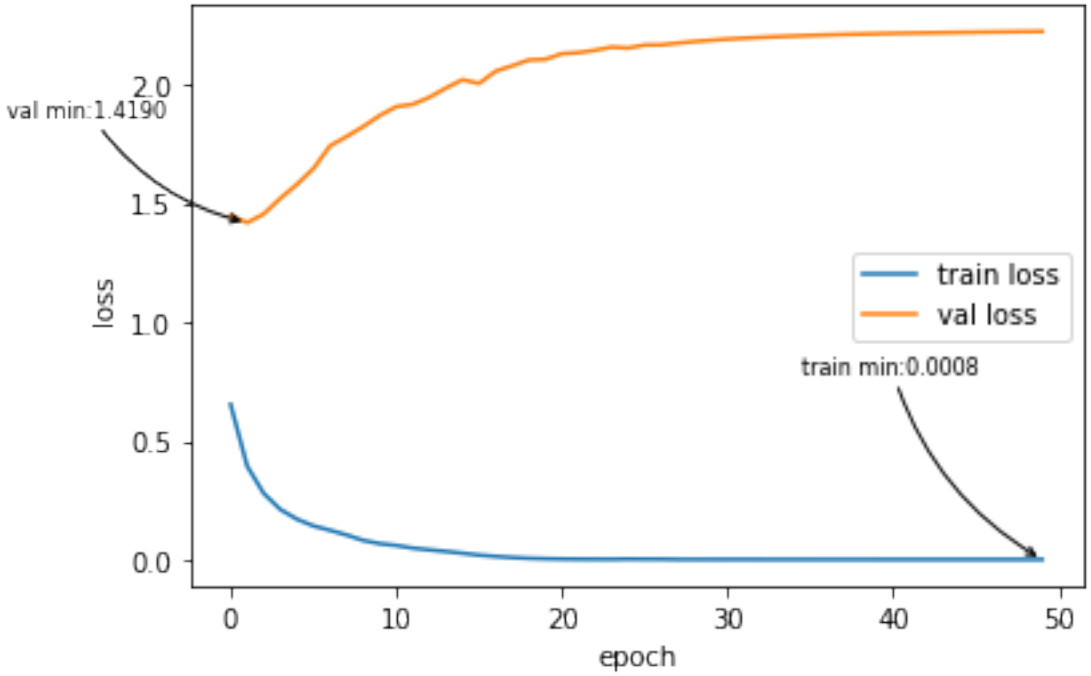
[7]:
```
#batch_size 128 windowslength 236
"""
CPC intersect sadly small batch size
New encoder
init hidden on each batch of data
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/29_01_21-17')
plot(train_losses, val_losses, train_acc, val_acc)
```

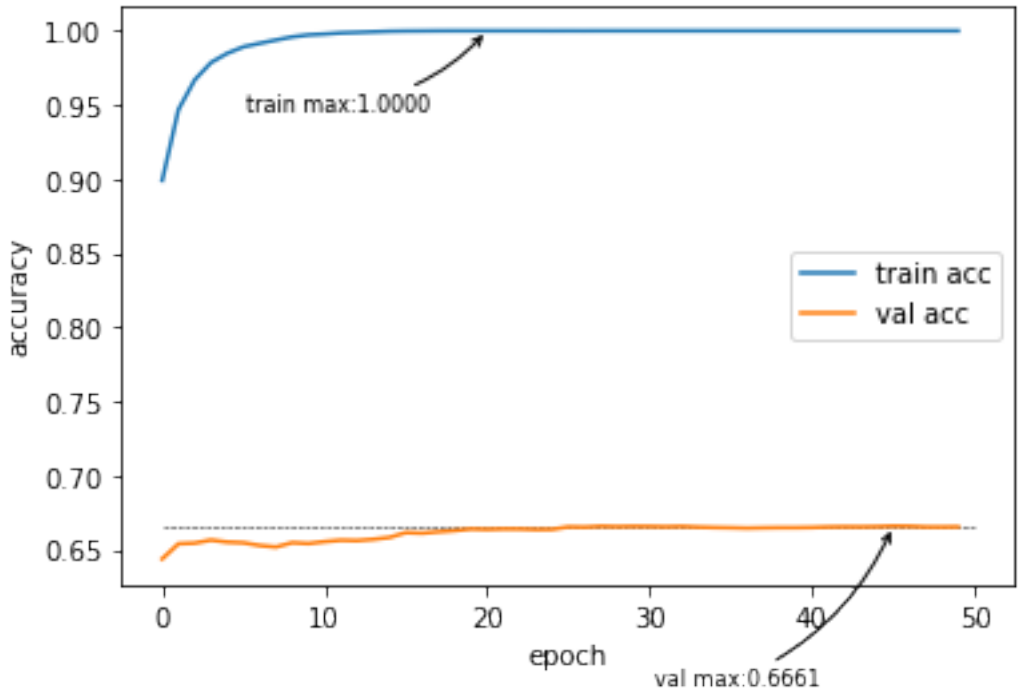




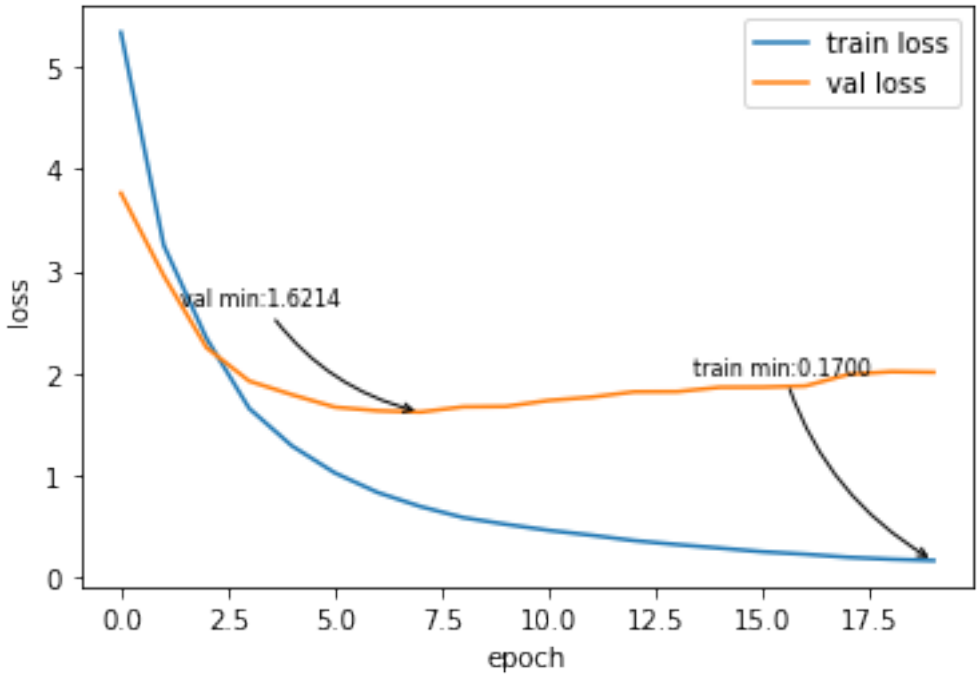
[6]:
```
#Baseline v14 on Ptbxl 52 classes no pca, batch_size 128 windowslength 236
"""
```

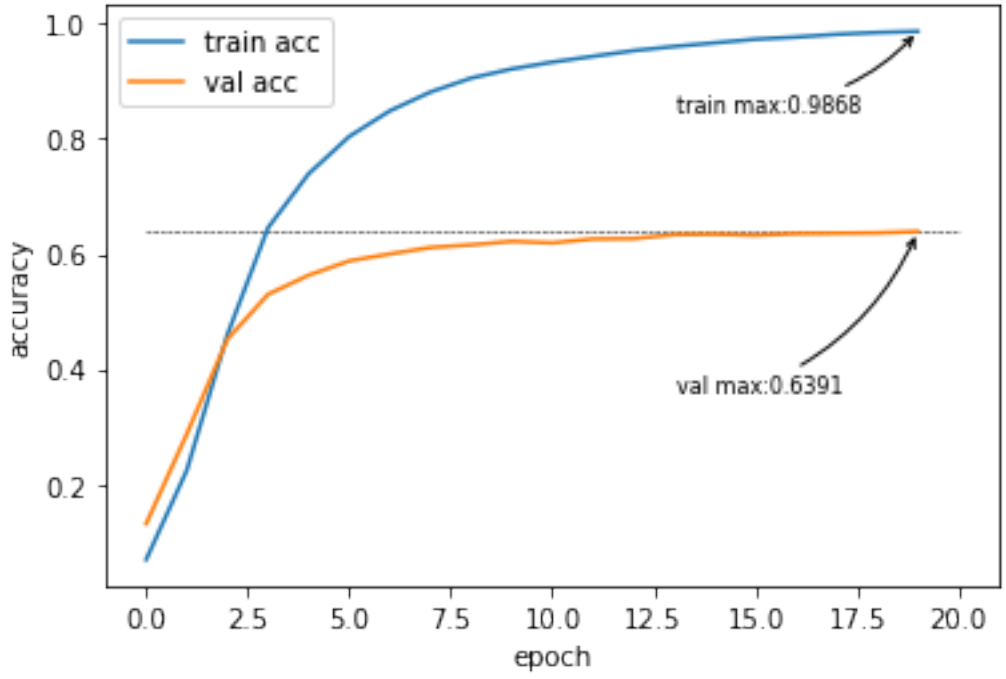
```
Trying CPC again with cleaned code
New encoder, challenge dataset ptbxl train, val challenge china
init hidden on each batch of data
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/29_01_21-13')
plot(train_losses, val_losses, train_acc, val_acc)
```





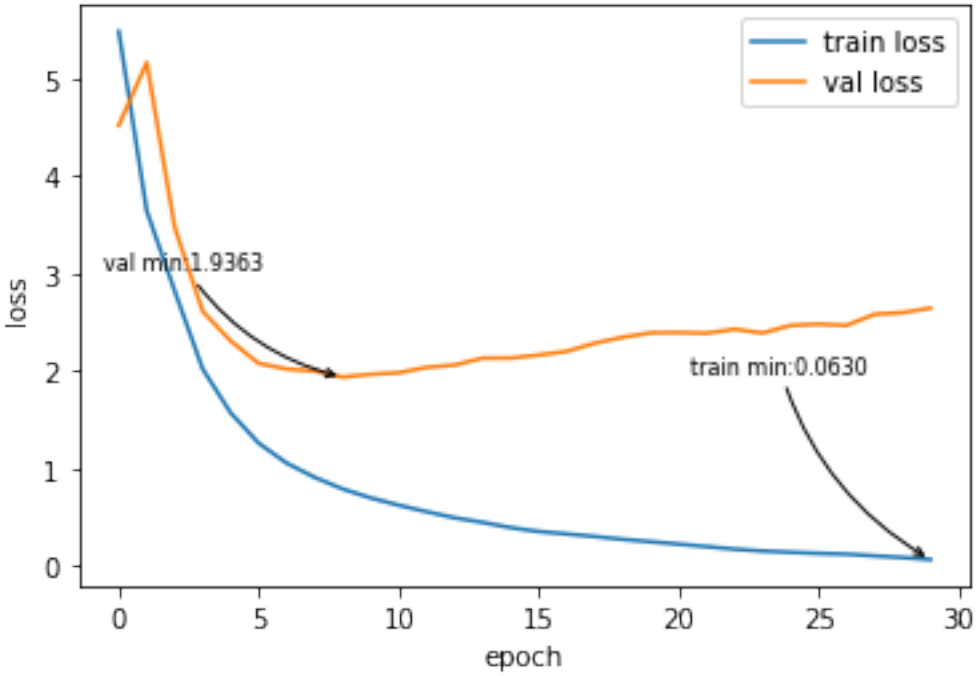
[3]:
```
#on Ptbxl 52 classes no pca, batch_size 128 windowslength 236
"""
Trying CPC again with cleaned code
New encoder, challenge dataset ptbxl train, val challenge china
init hidden on each batch of data
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/28_01_21-20')
plot(train_losses, val_losses, train_acc, val_acc)
```

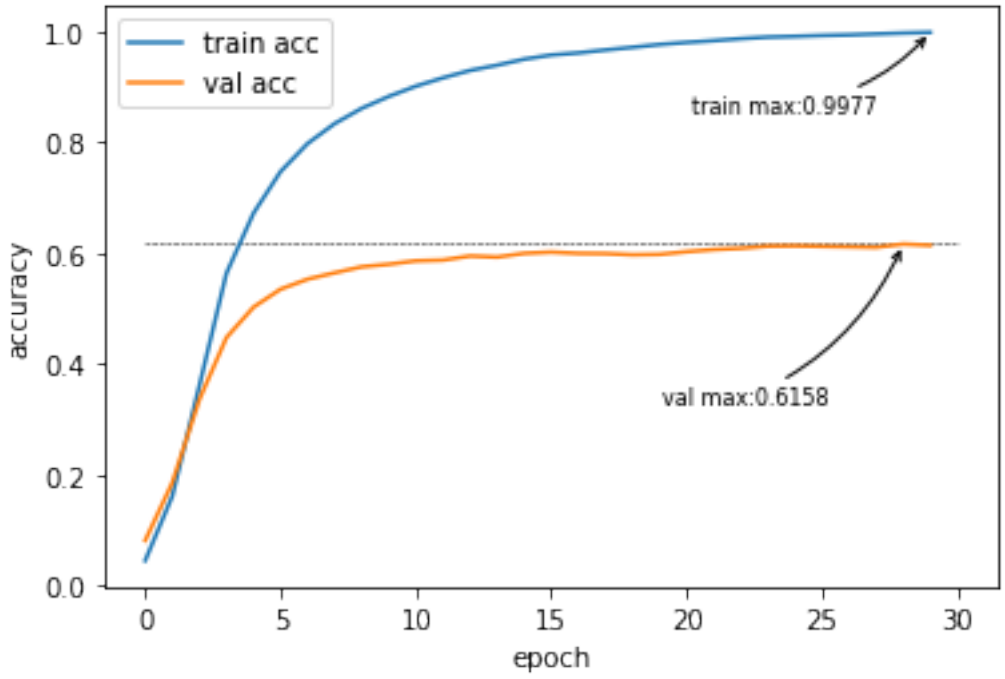




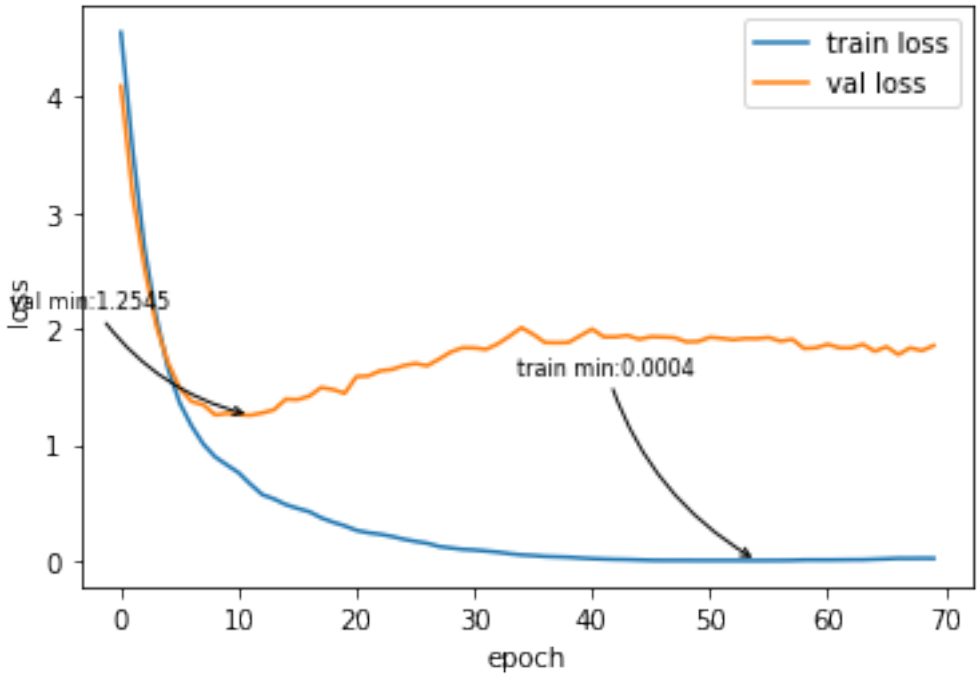
[3]:
```
#Baseline v2encoder on Ptbxl 52 classes no pca, batch_size 128 windowslength 236
"""
```

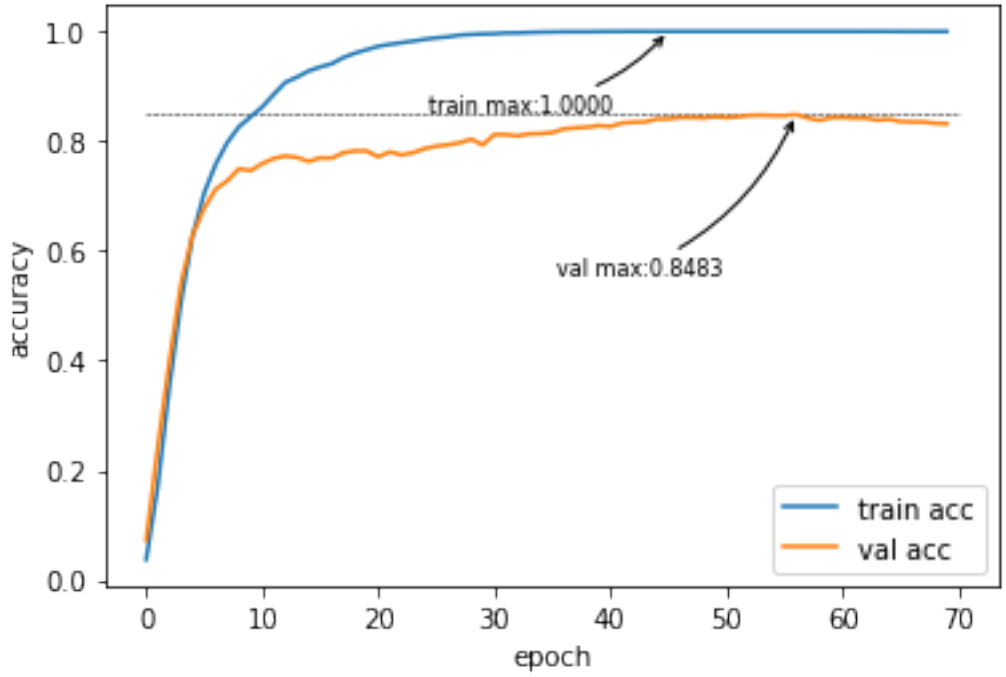
```
Trying CPC again with cleaned code
New encoder, challenge dataset ptbxl train, val challenge china

"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/28_01_21-19')
plot(train_losses, val_losses, train_acc, val_acc)
```





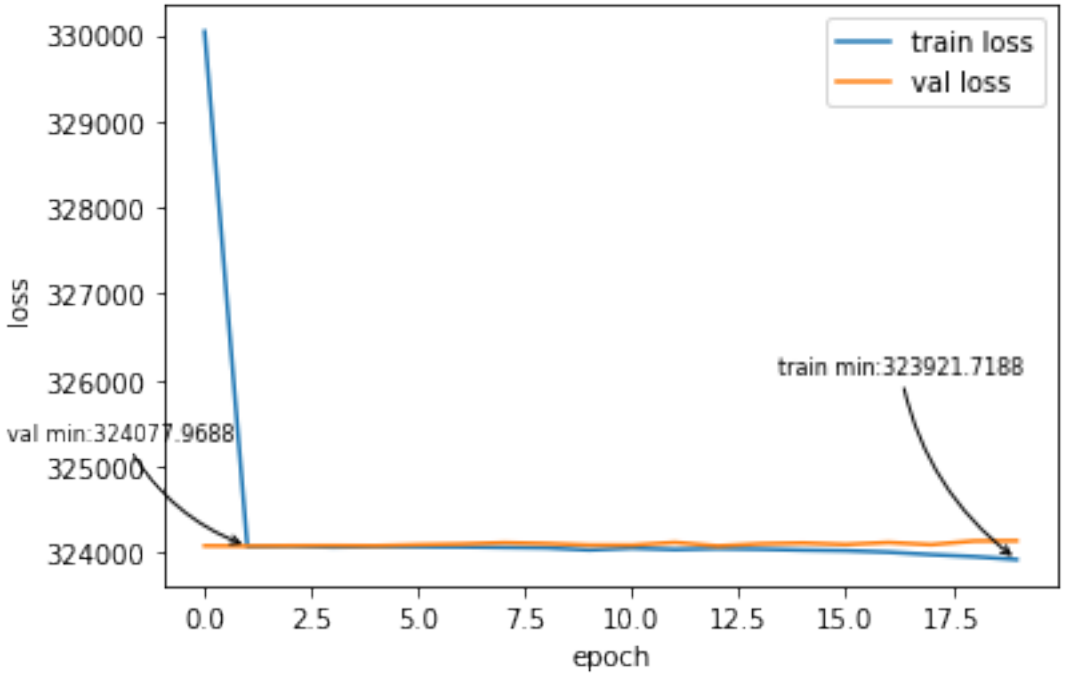
[6]:
```
#Baseline v14 on Ptbxl 52 classes no pca, batch_size 128 new accuracy function,␣
↪simple loss, windowslength 9500
"""
Trying CPC again with cleaned code and obscure loss function

"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/26_01_21-15')
plot(train_losses, val_losses, train_acc, val_acc)
```

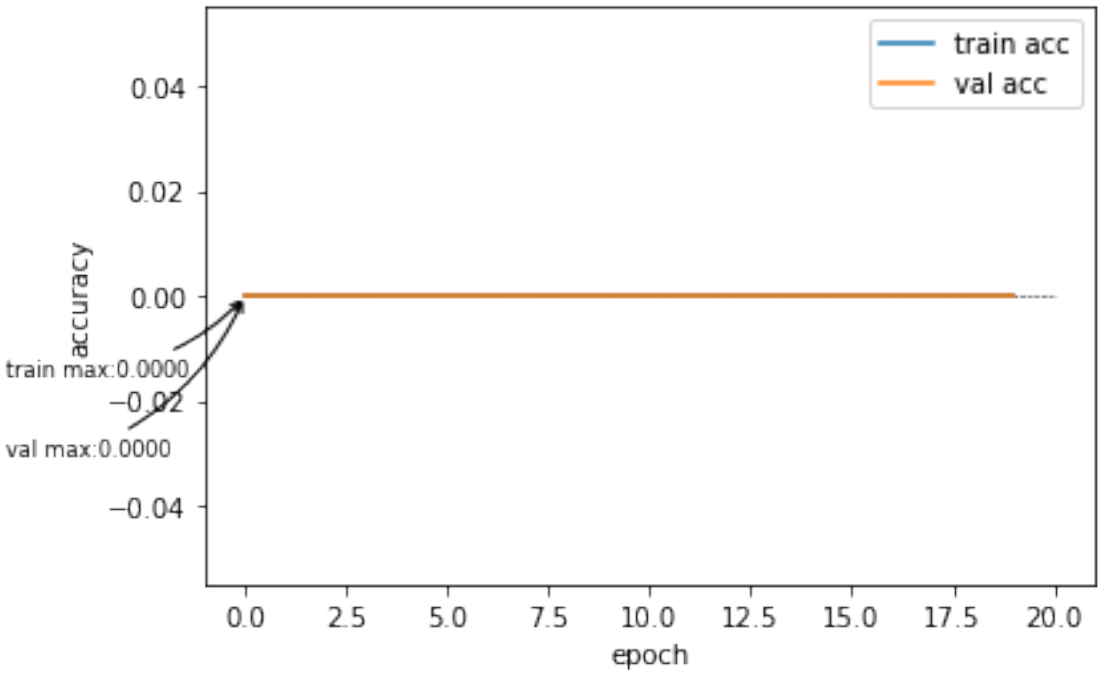




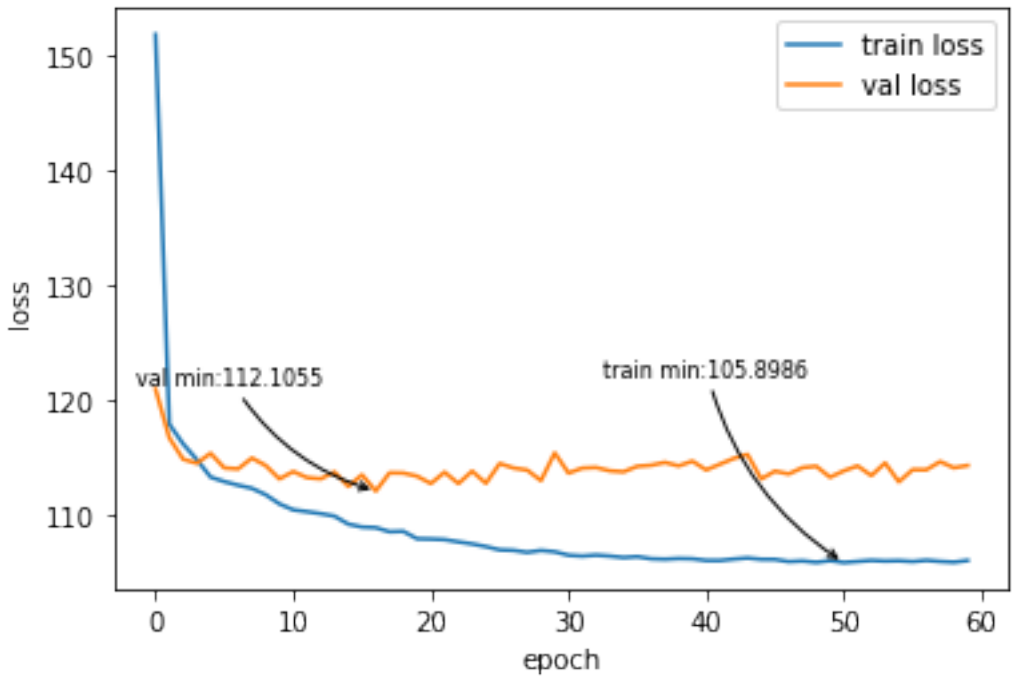
[4]:
```
#Baseline v14 on Ptbxl 52 classes no pca, batch_size 128 new accuracy function,␣
↪simple loss, windowslength 9500
```

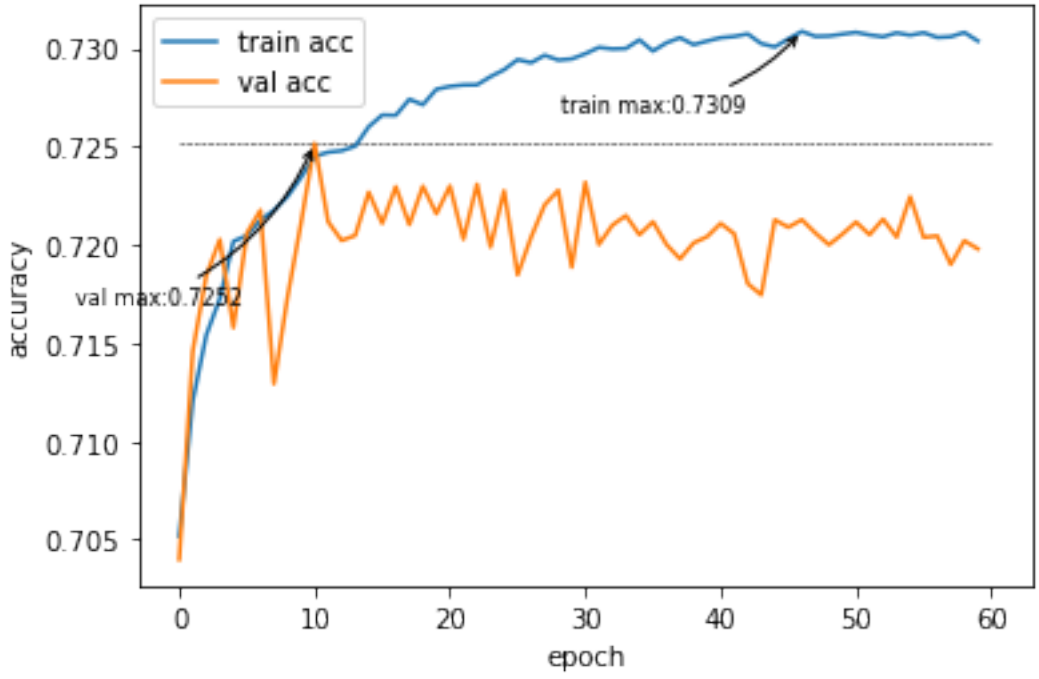
```
"""
Trying CPC again with cleaned code and obscure loss function

"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/25_01_21-19')
plot(train_losses, val_losses, train_acc, val_acc)
```





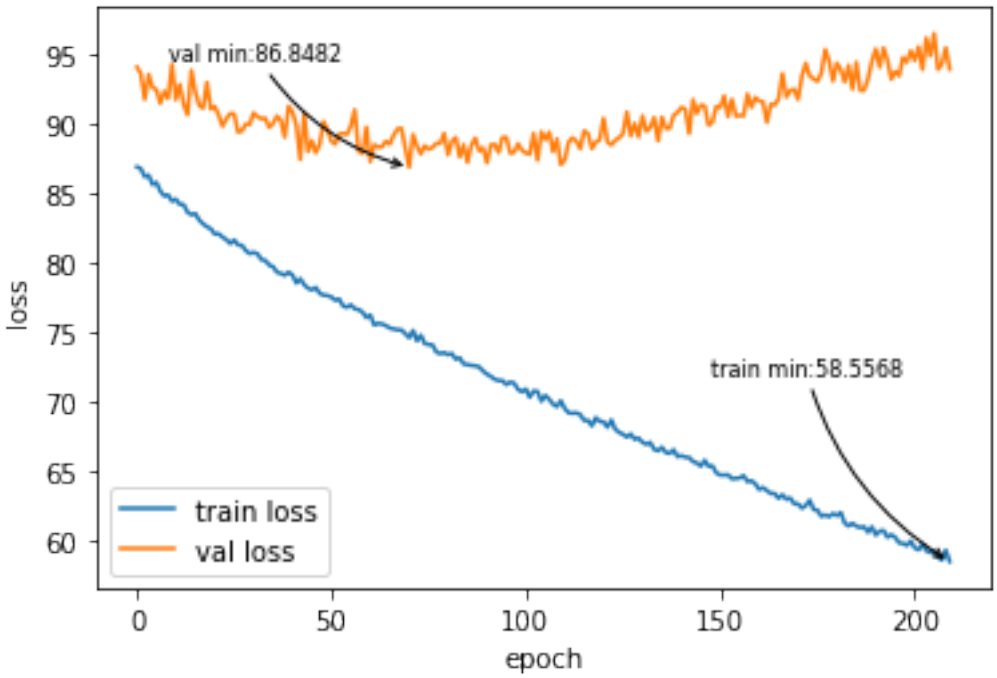
[4]:
```
#Baseline v14 on Ptbxl 52 classes no pca, batch_size 128 new accuracy function, simple loss, windowslength 9500
"""
Testing Multiscale ResNet (Conv1d implementation from github) + conv1x1 to summarize 182 values at end

"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/Downloads/Github/contrastive-predictive-coding/models/19_01_21-14')
plot(train_losses, val_losses, train_acc, val_acc)
```

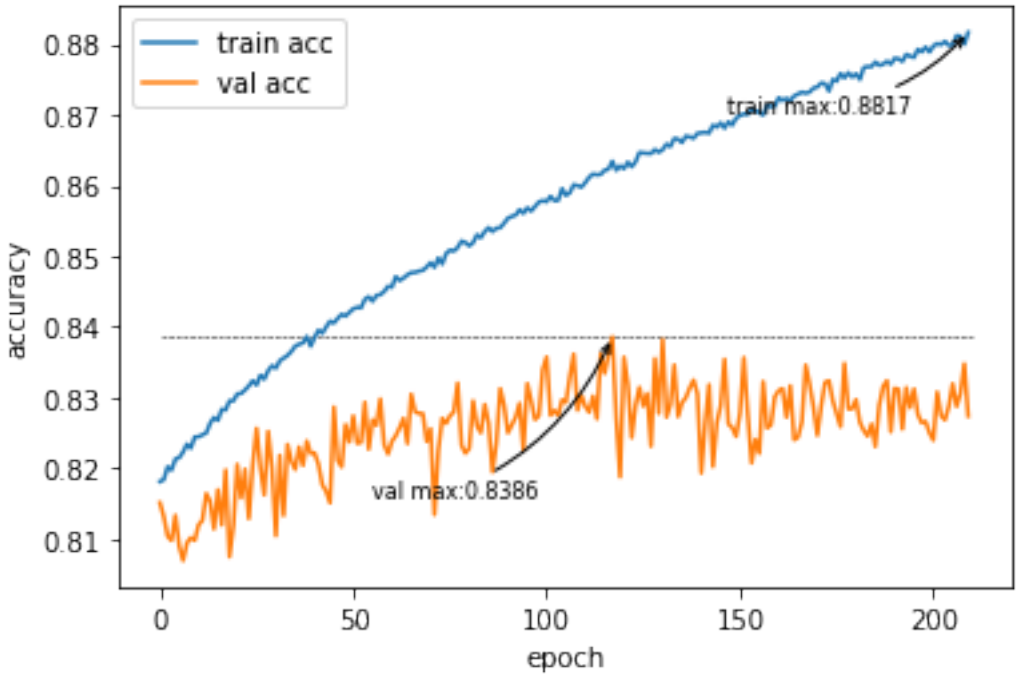




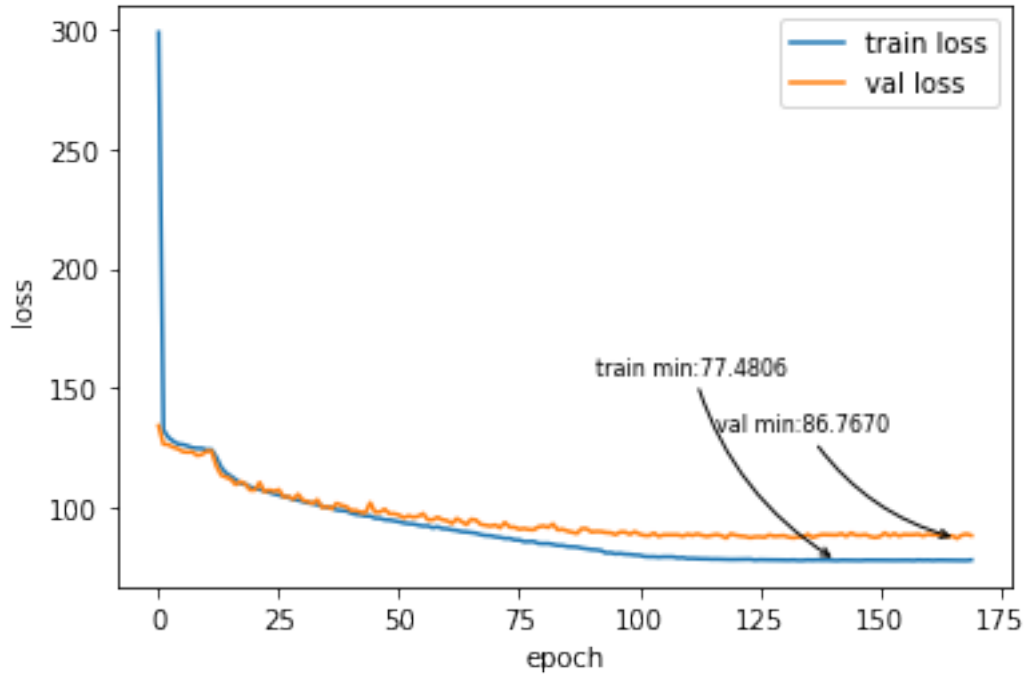
[8]:
```
#Baseline v0_3 on Ptbxl 52 classes no pca, batch_size 128 new accuracy␣
↪function, simple loss, windowslength 9500
```

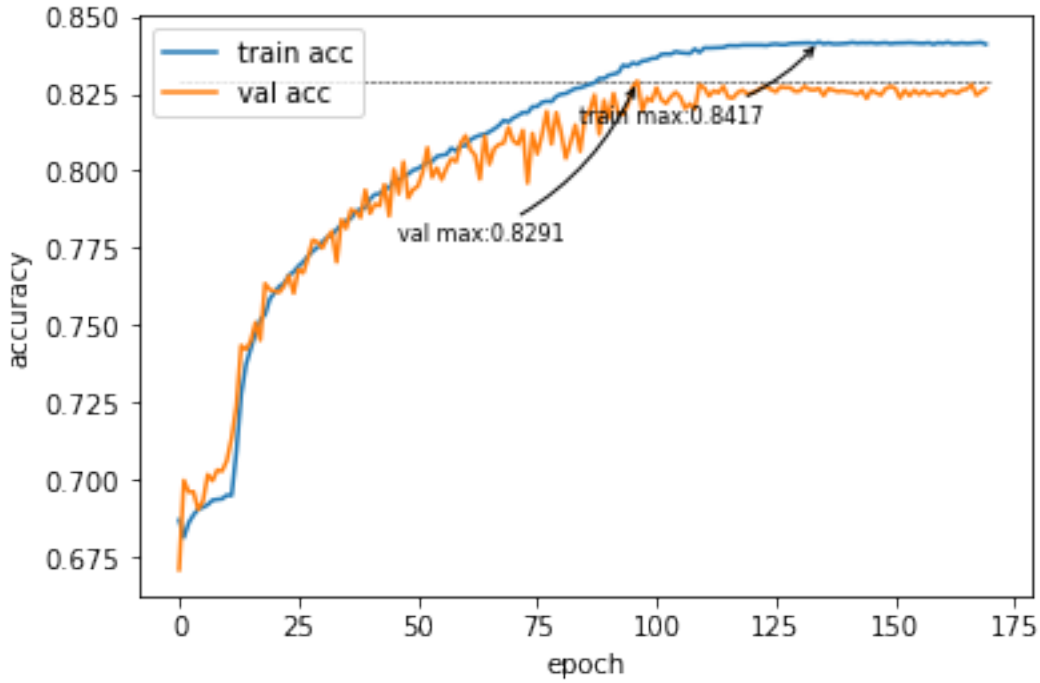
```
"""
6 Cnn1d, 3 downsample, other make features
learn rate fixed

"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/19_01_21-18')
plot(train_losses, val_losses, train_acc, val_acc)
```





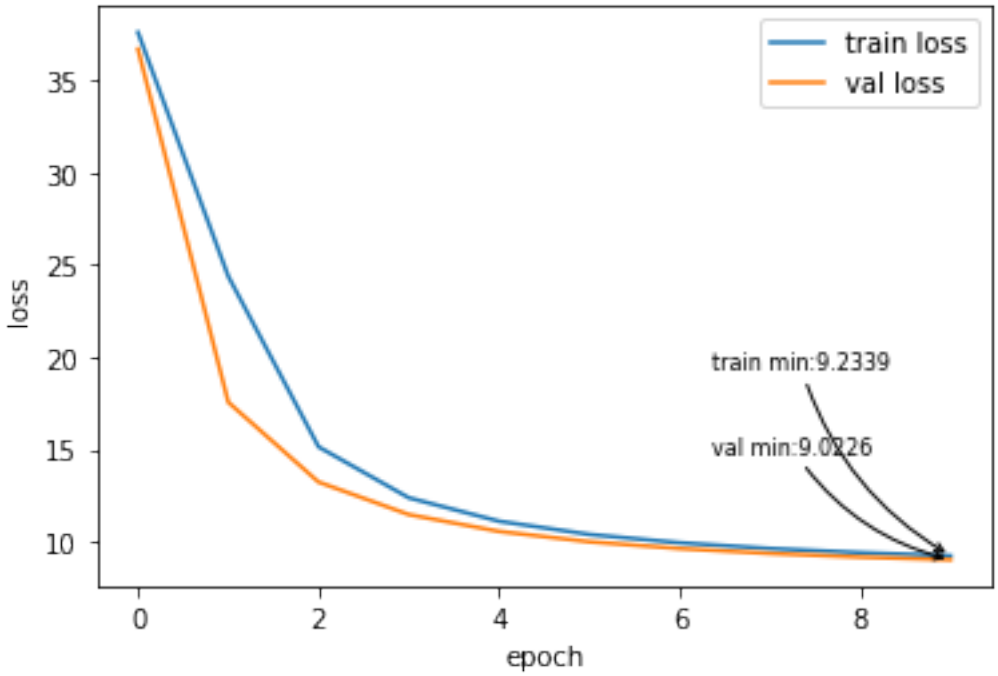
[11]:
```
#Baseline v0_3 on Ptbxl 52 classes no pca, batch_size 128 new accuracy␣
 ↪function, simple loss, windowslength 9500
"""
6 Cnn1d, 3 downsample, other make features
learn rate smaller over time (21 *0.5 here)

"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/18_01_21-14')
plot(train_losses, val_losses, train_acc, val_acc)
```

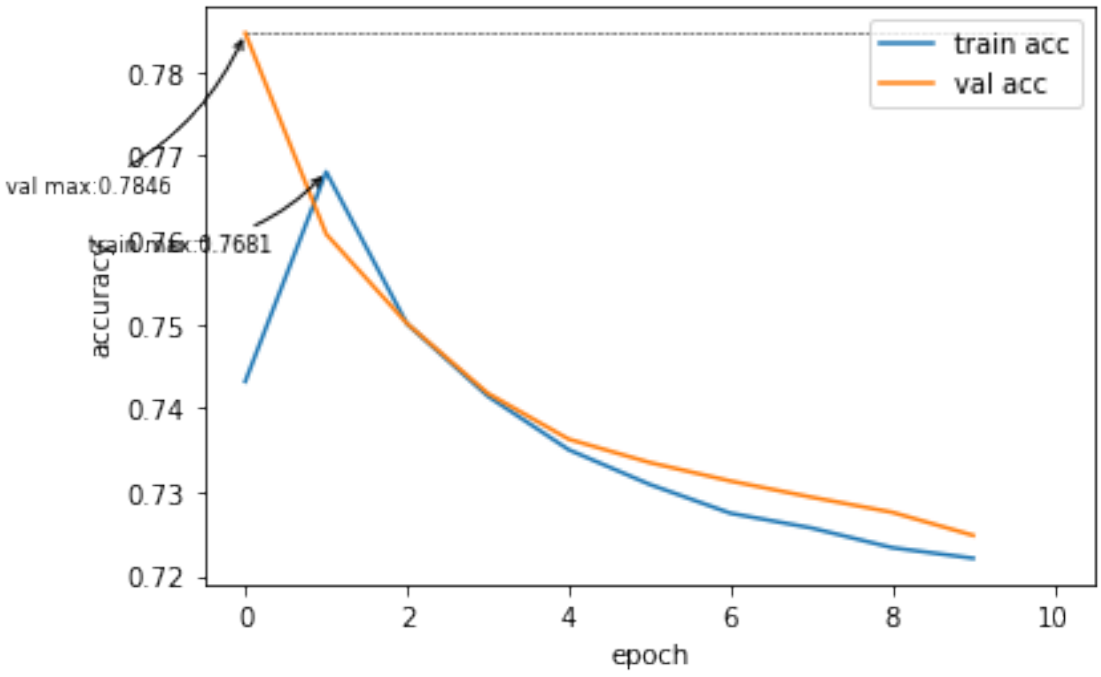




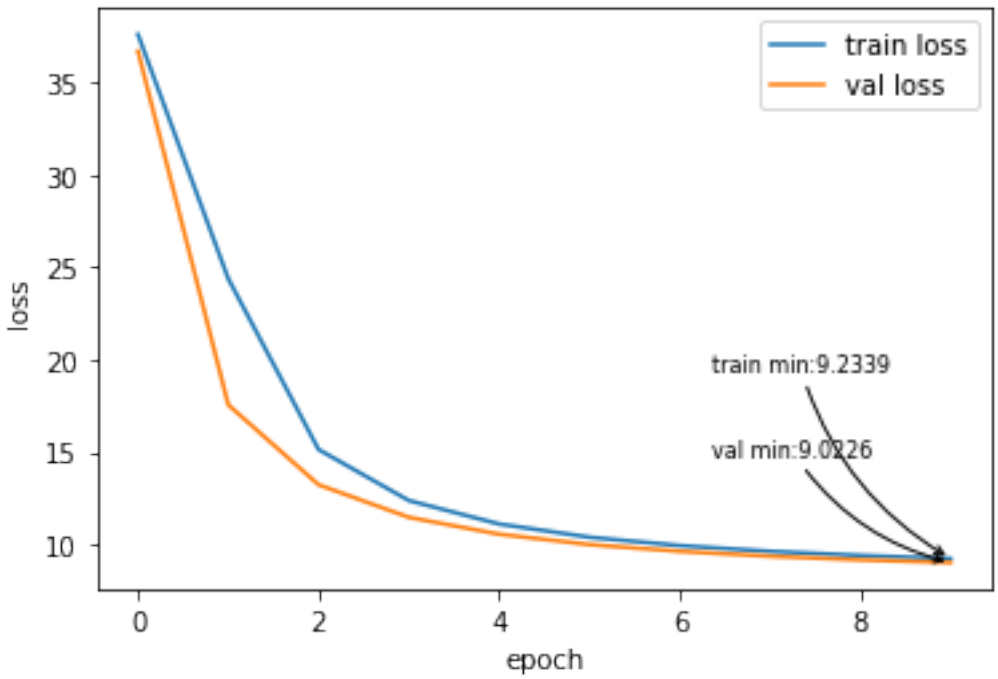
[3]:
```
#Baseline v13 on Ptbxl 52 classes no pca, batch_size 128 new accuracy function, simple loss, windowslength 9500
```

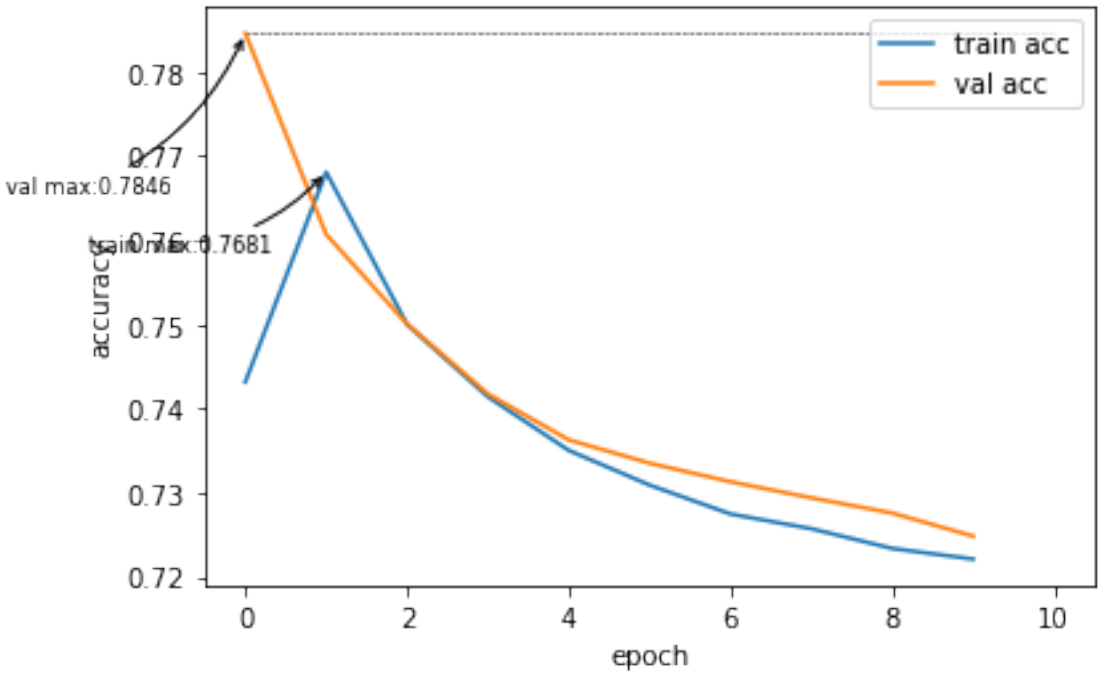
```
"""
TCN-like Block: 2 Conv1D, first stride 2, second dilation 2
weight_norm as in TCN
residual connection (needs downsample in data dimension to 4745)

No Downsampling, Linear on on 52x9500[-1]? Weird
Linear 52->52
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/14_01_21-16')
plot(train_losses, val_losses, train_acc, val_acc)
```





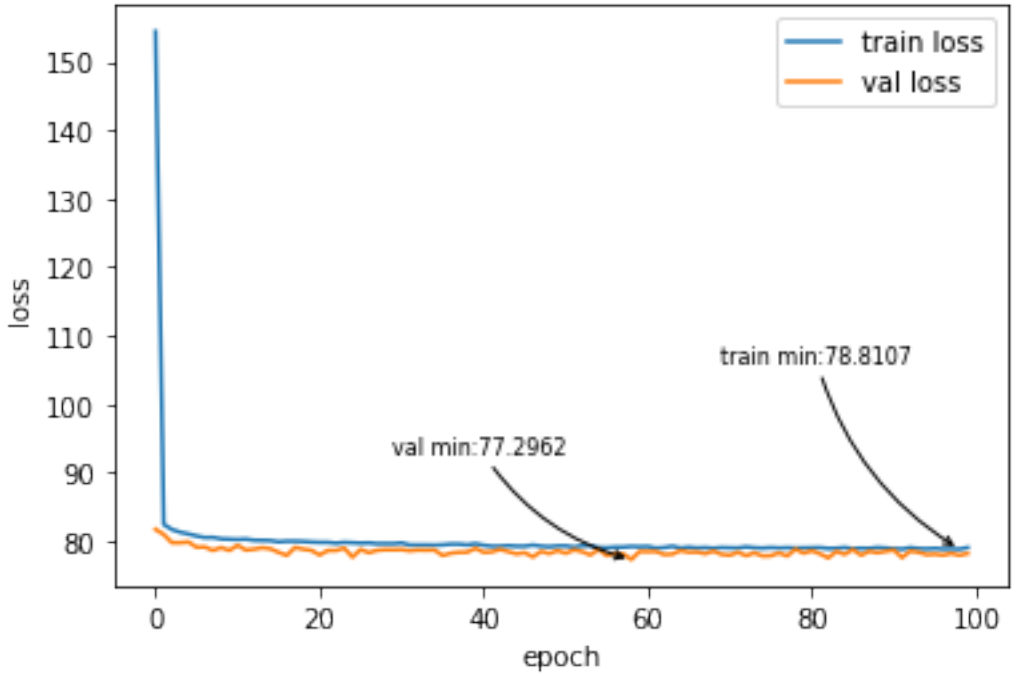
[11]:
```
#Baseline v12 on Ptbxl 52 classes no pca, batch_size 8 new accuracy function,␣
 ↪simple loss, windowslength 9500
"""
BATCH_SIZE ONLY 8! (Model too big) So 10 times more updates than Baseline v11
Input is also flattened like in Mnist example (hm)
so channel input is 1 instead of 12
TCN with 3 Blocks (12, 24, 52 channels), dropout 0.2, kernel_size 3
No Downsampling, Linear on on 52x9500[-1]? Weird
Linear 52->52
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/14_01_21-16')
plot(train_losses, val_losses, train_acc, val_acc)
```

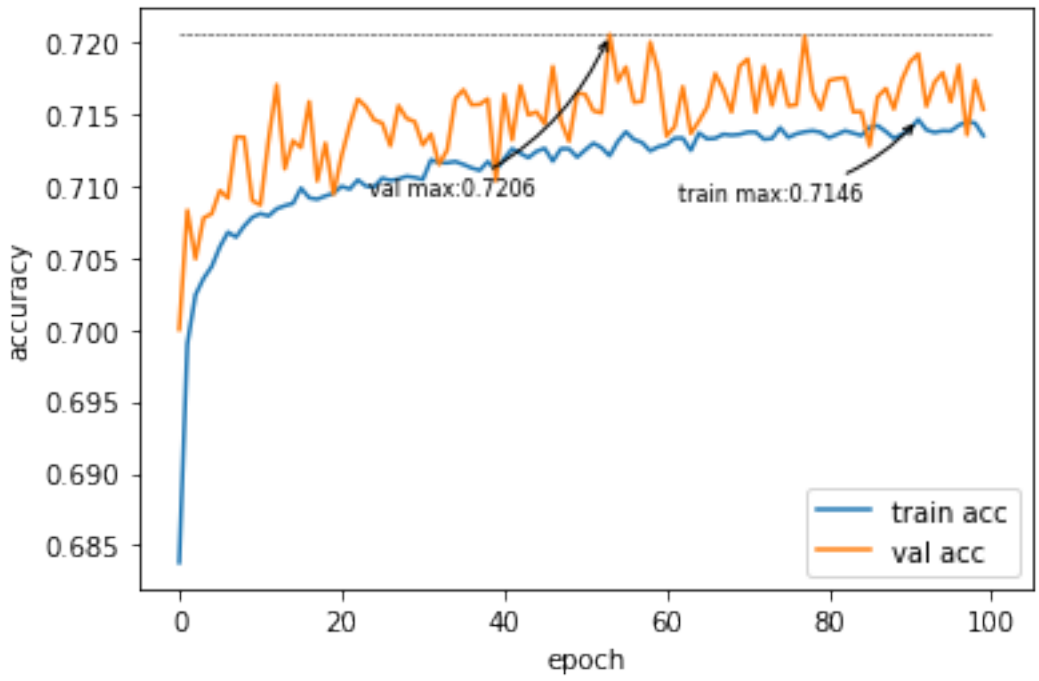




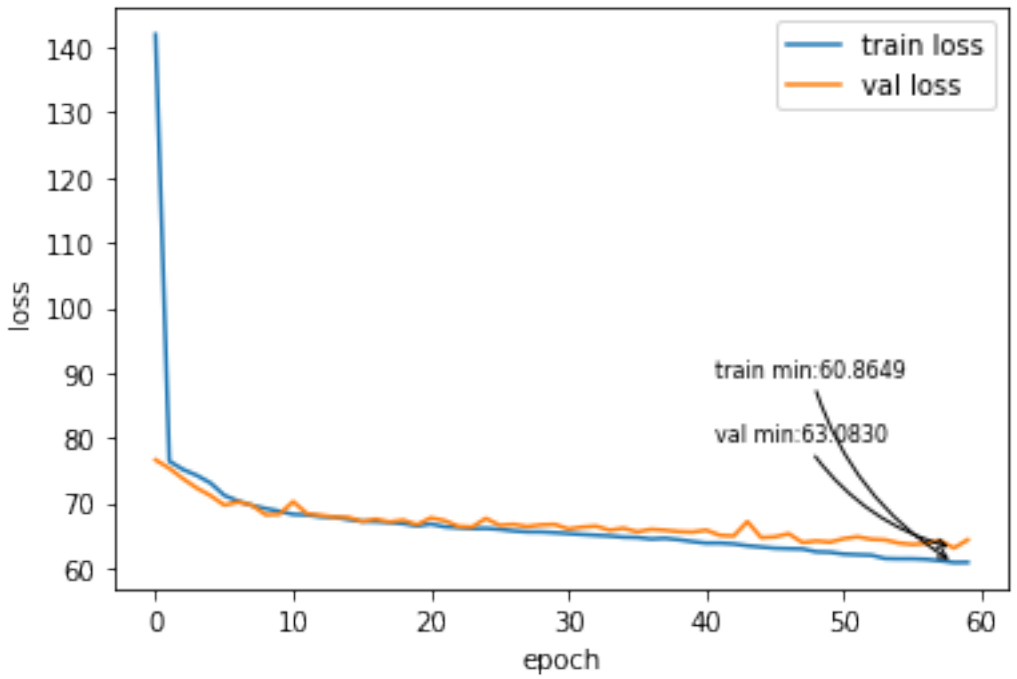
[10]:
```
#Baseline v11 on Ptbxl 52 classes no pca, batch_size 80 new accuracy function,␣
↪simple loss, windowslength 9500
```

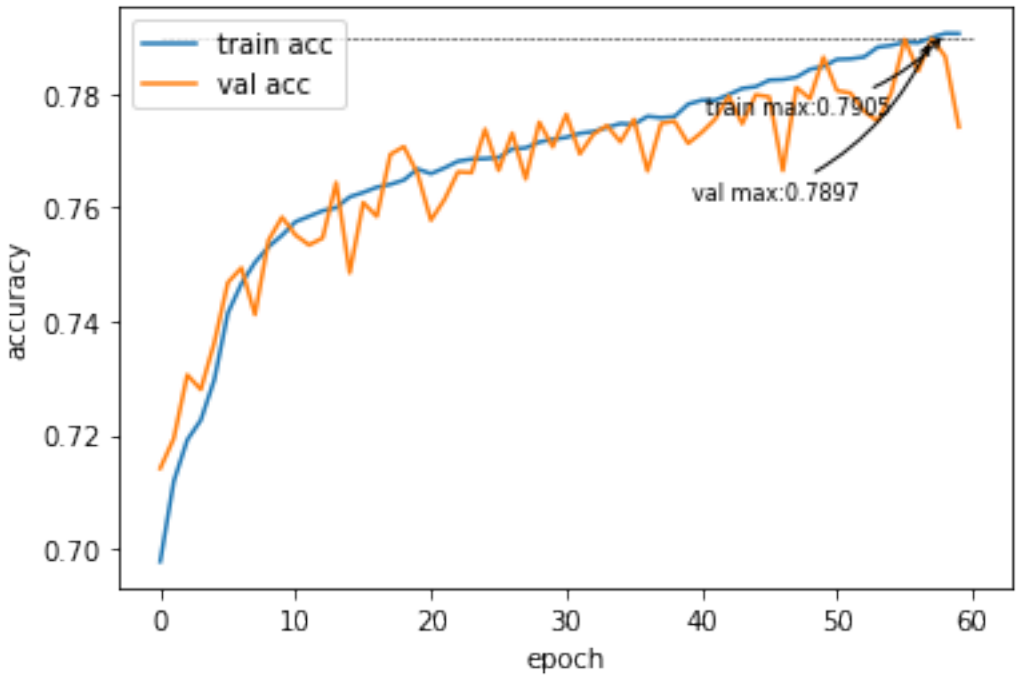
```
"""
TCN with 3 Blocks (12, 24, 52 channels), dropout 0.2, kernel_size 3
No Downsampling, Linear on on 52x9500[-1]? Weird
Very close to official tcn mnist example
Linear 52->52
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/14_01_21-15')
plot(train_losses, val_losses, train_acc, val_acc)
```





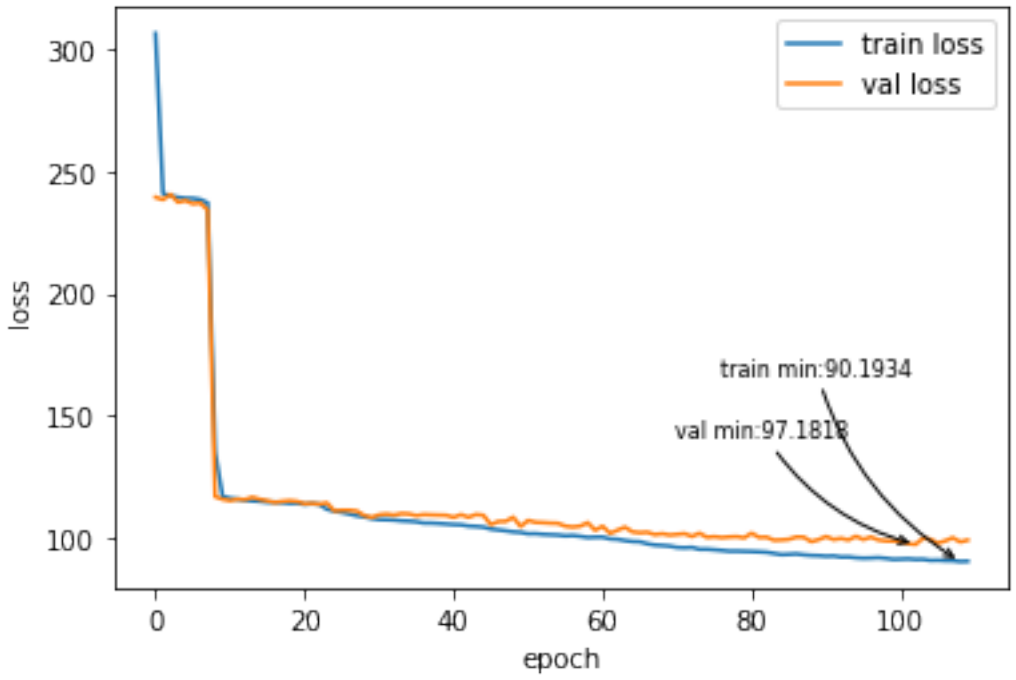
[8]:
```
#Baseline v10 on Ptbxl 52 classes no pca, batch_size 80 new accuracy function,␣
↪simple loss, windowslength 9500
"""
TCN with 3 Blocks (12, 24, 52 channels), dropout 0.2, kernel_size 3
Downsampling with Conv1x1 to summarize all 9500 values

Linear 52->52
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/14_01_21-14')
plot(train_losses, val_losses, train_acc, val_acc)
```

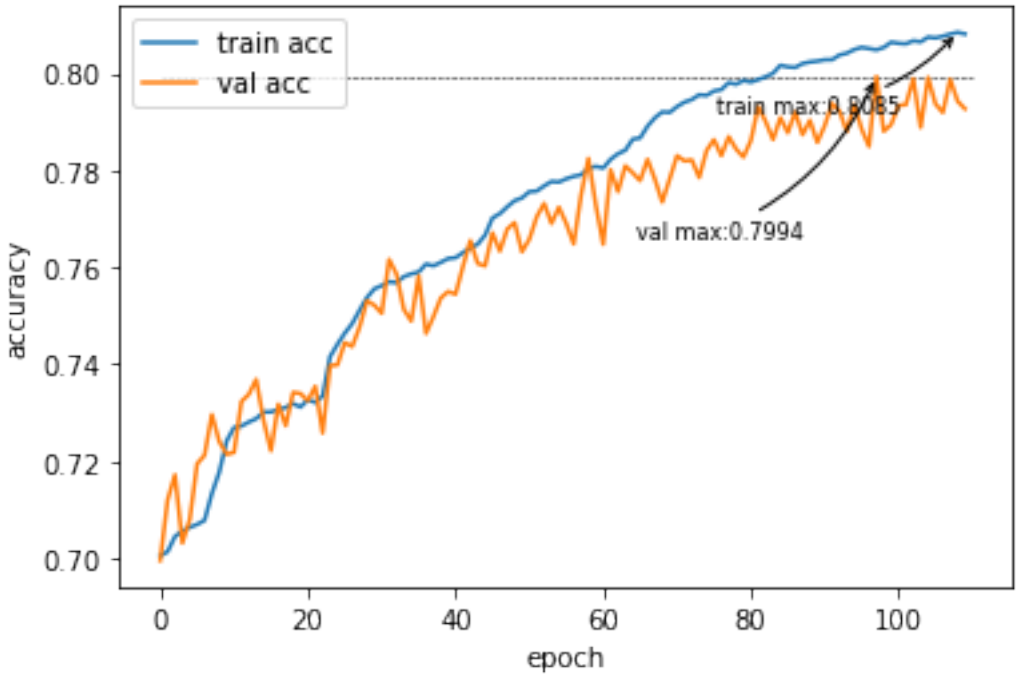




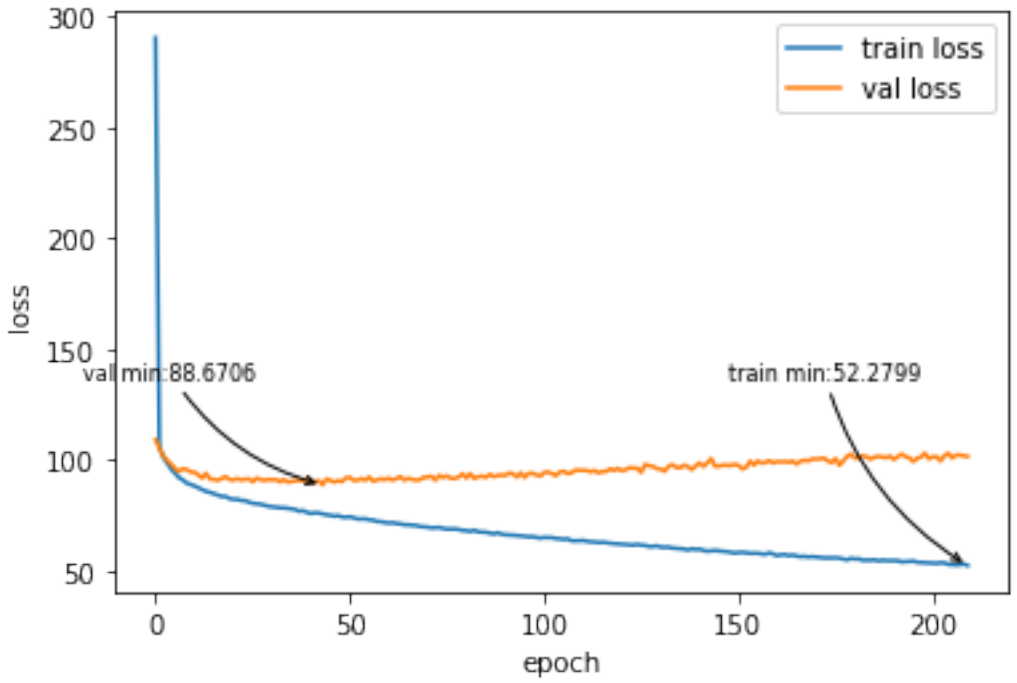
[6]:
```
#Baseline v0 on Ptbxl 52 classes no pca, batch_size 128 new accuracy function,␣
↪simple loss, windowslength 9500
```

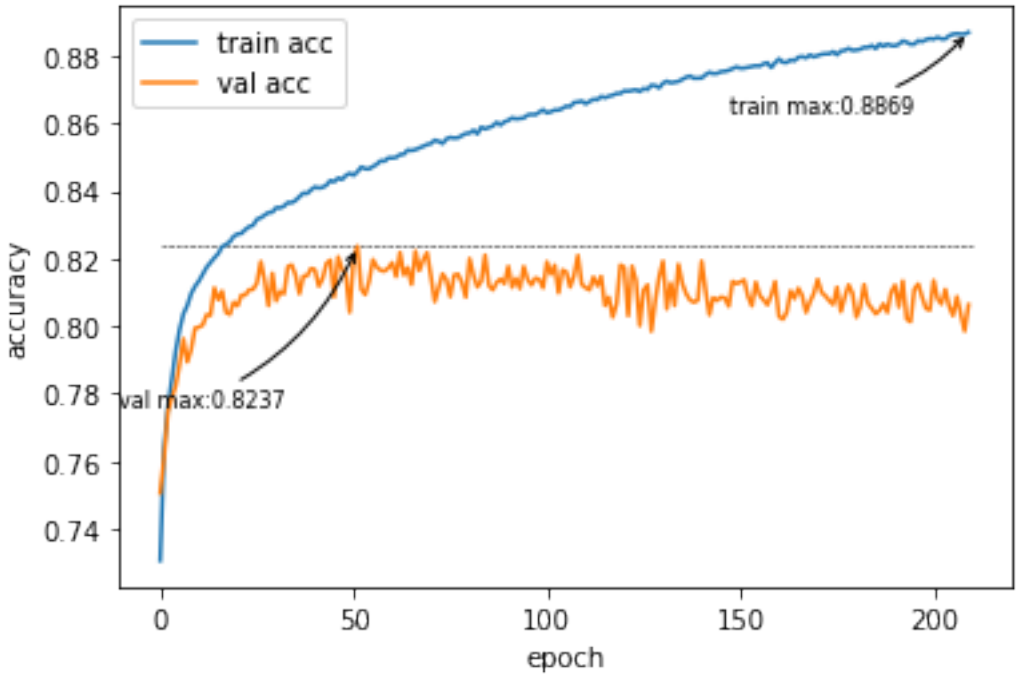
```
"""
2 Layers of Convs with filter size 7 and 3 and dilation of 1, 3
Summarizes all 9488 values with Transpose + Con1x1
Full connected has only out_channels as input
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/14_01_21-11')
plot(train_losses, val_losses, train_acc, val_acc)
```





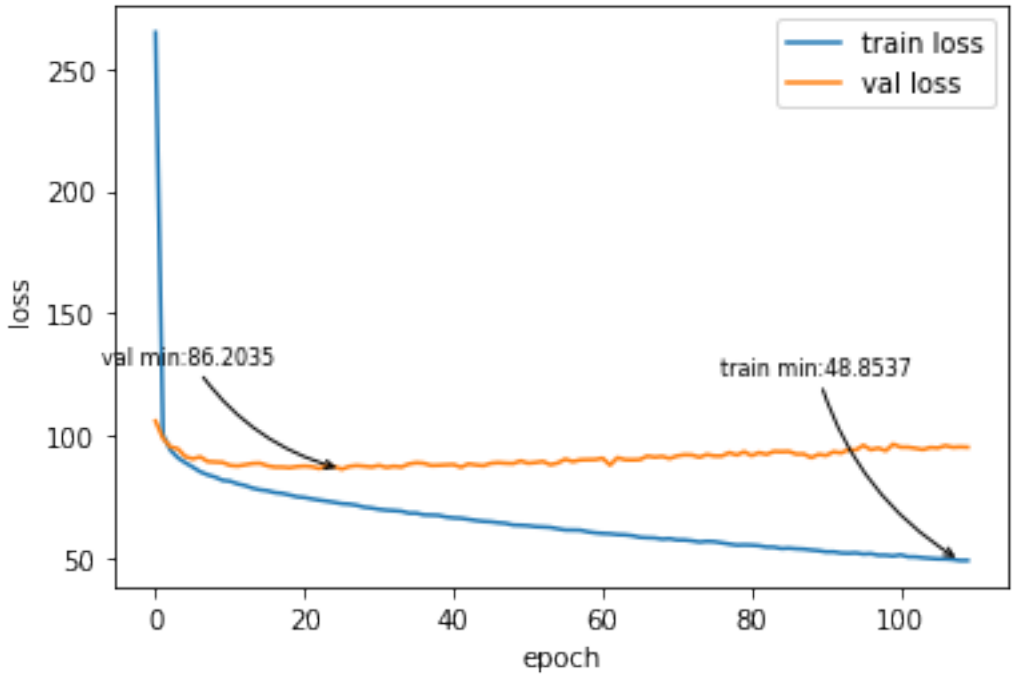
[4]:
```
#Baseline v9 on Ptbxl 52 classes no pca, batch_size 24 new accuracy function,␣
 ↪simple loss, windowslength 9500
"""
Similiar to v2, v8
4 Layers of Convs with filter size 3 and NO DILATION
Max Poolwith size 3 after every Layer (besides last) to downsample
Summarizes all 348 values with Transpose + Con1x1
Full connected has only out_channels as input
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/13_01_21-19')
plot(train_losses, val_losses, train_acc, val_acc)
```

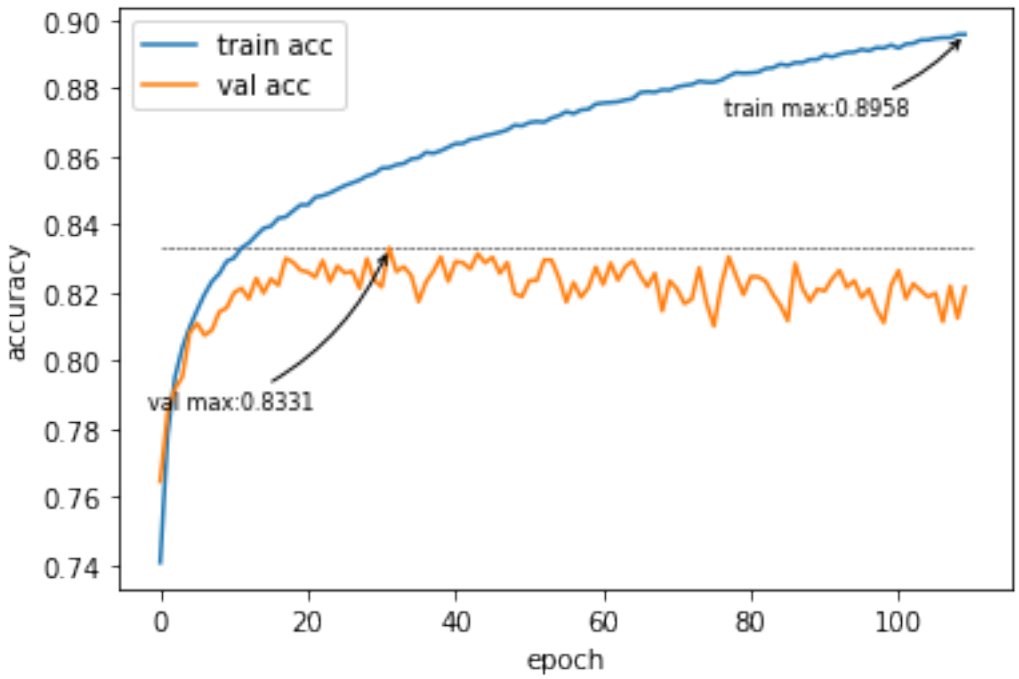




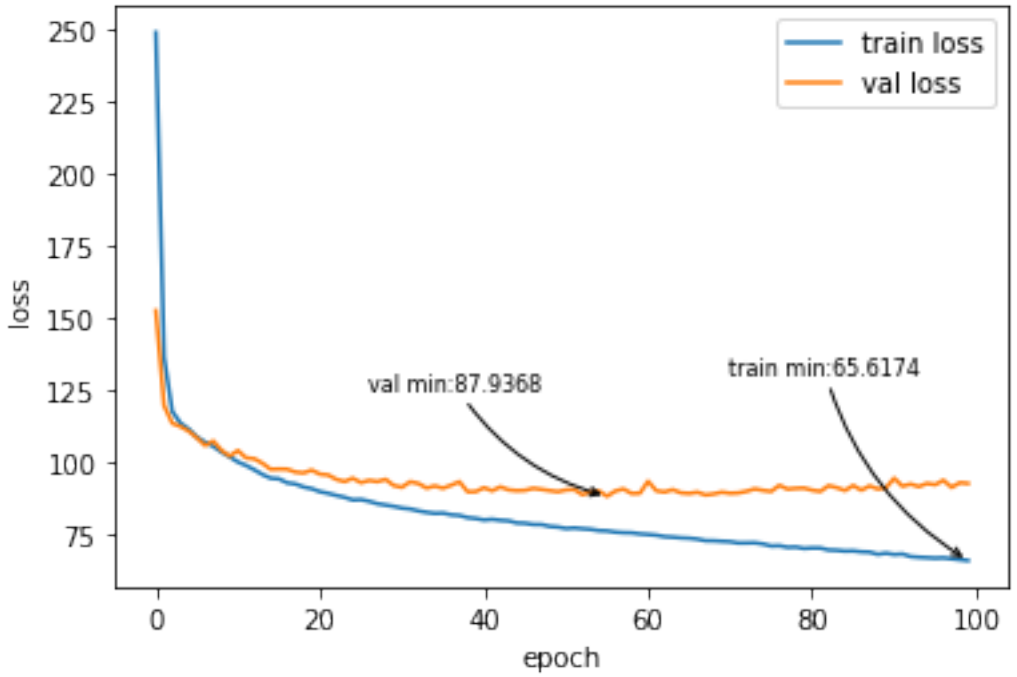
[23]:
```
#Baseline v8 on Ptbxl 52 classes no pca, batch_size 24 new accuracy function,␣
↪simple loss, windowslength 9500
```

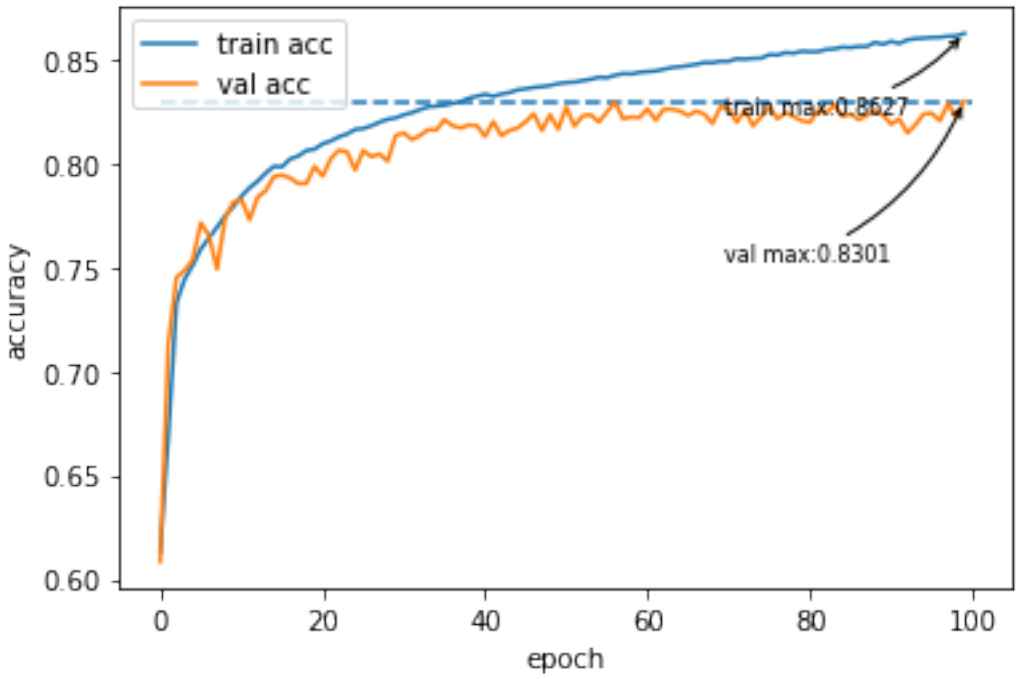
```
"""
Similiar to v2
4 Layers of Convs with filter size 3 and dilation of 2^n to get a big receptive␣
 ↪field
Max Poolwith size 3 after every Layer (besides last) to downsample
Summarizes all 334 values with Transpose + Con1x1
Full connected has only out_channels as input
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/13_01_21-18')
plot(train_losses, val_losses, train_acc, val_acc)
```





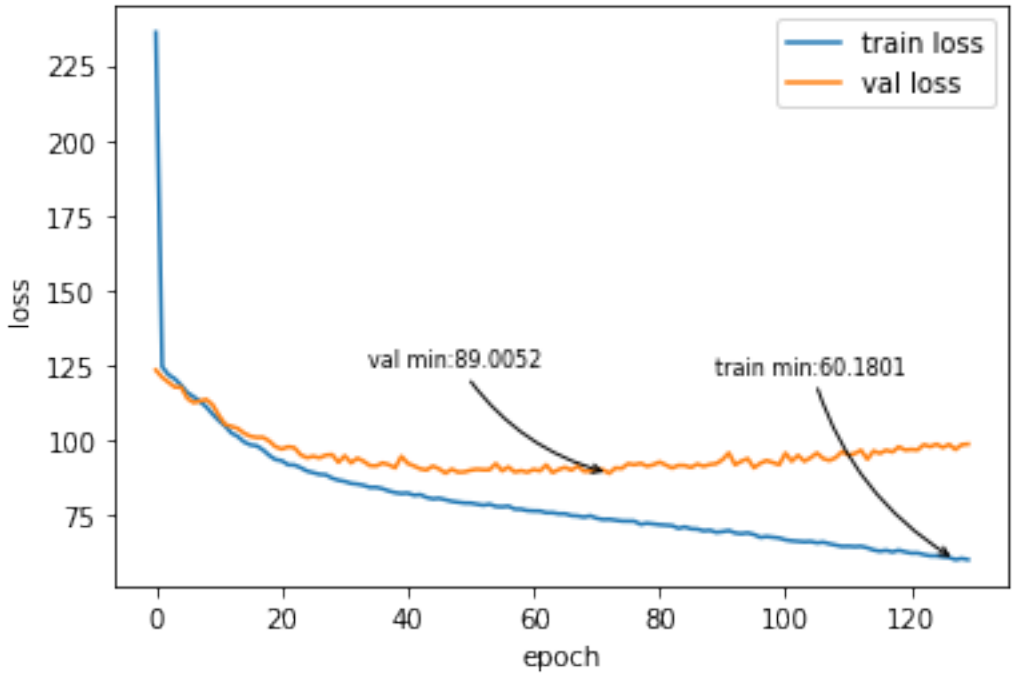
[15]:
```
#Baseline v7 (no Batchnorm) on Ptbxl 52 classes no pca, batch_size 128 new␣
 ↪accuracy function, simple loss, windowslength 9500
"""Has one less layer than v6 but a stride of 2 + kernel size of 7 in the first␣
 ↪layer
(to downsample as in resnet).
Summarizes all values using transpose + conv1x1
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/13_01_21-16')
plot(train_losses, val_losses, train_acc, val_acc)
```

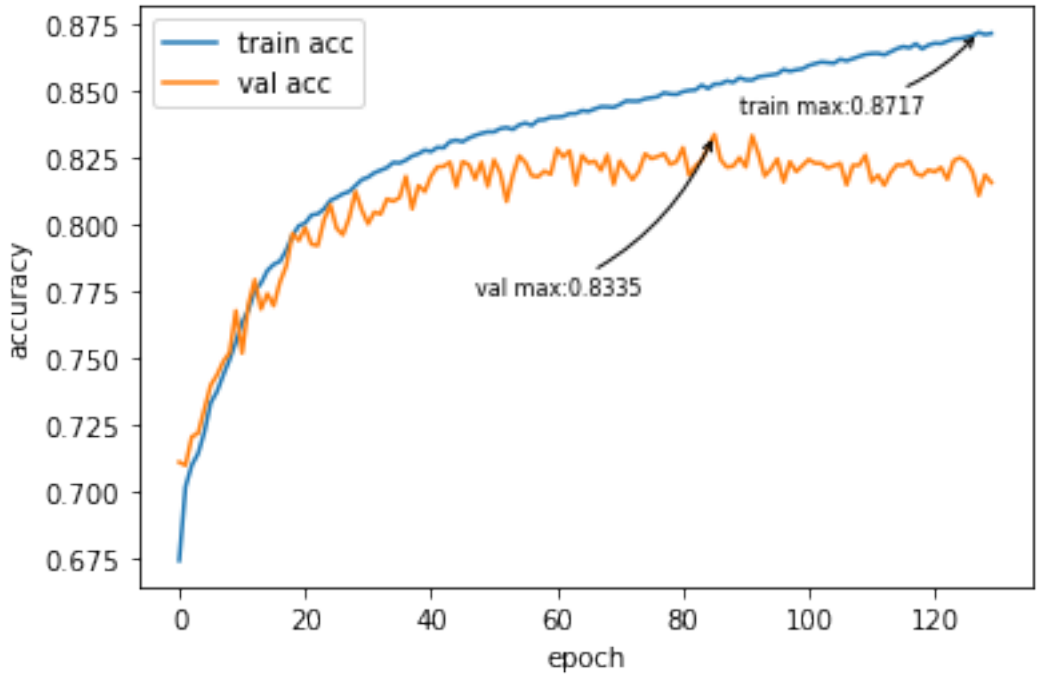




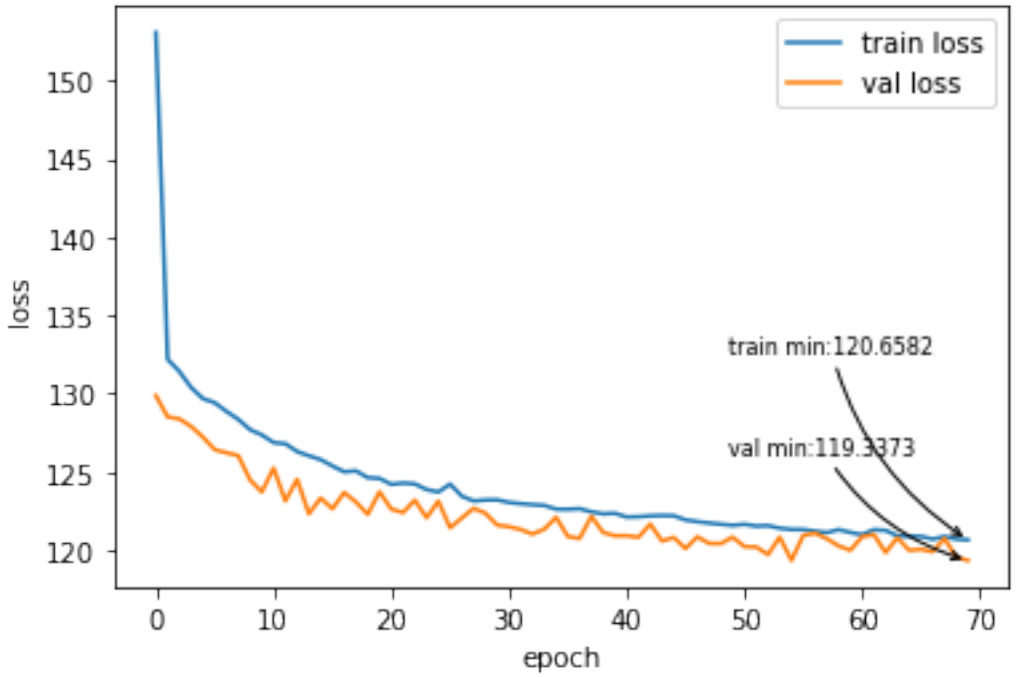
[3]:
```
#Baseline v6 (no Batchnorm) on Ptbxl 52 classes no pca, batch_size 128 new␣
↪accuracy function, simple loss, windowslength 9500
```

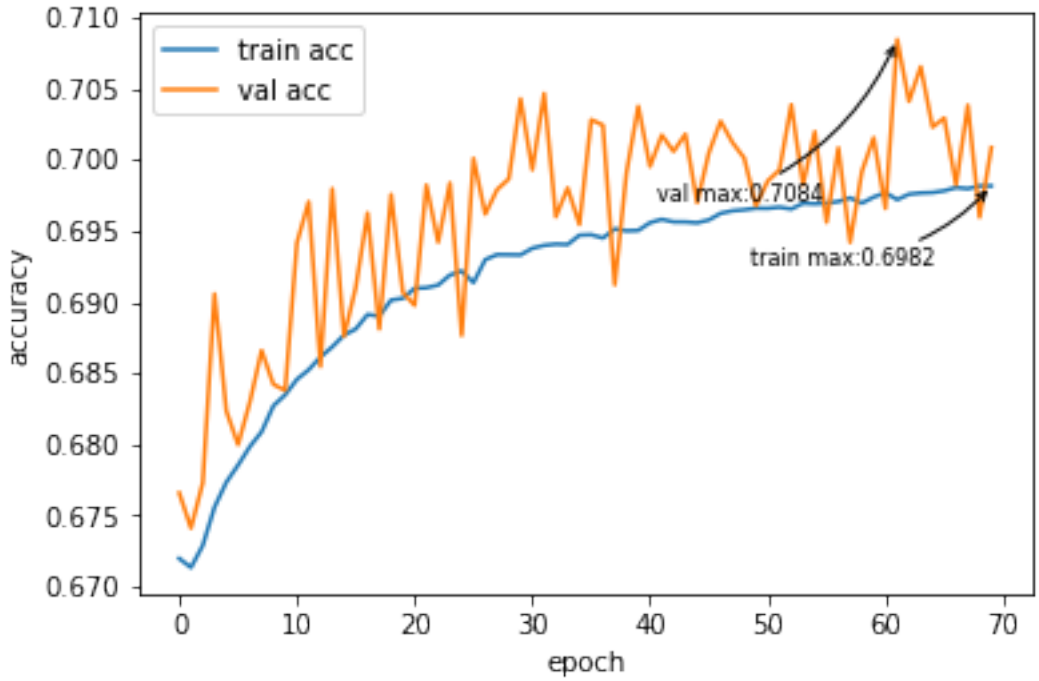
```
"""6 Layers of Convs with filter size 3 and dilation of 2^n to get a really big␣
 ↪receptive field
Summarizes all values using transpose + conv1x1
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/13_01_21-15')
plot(train_losses, val_losses, train_acc, val_acc)
```





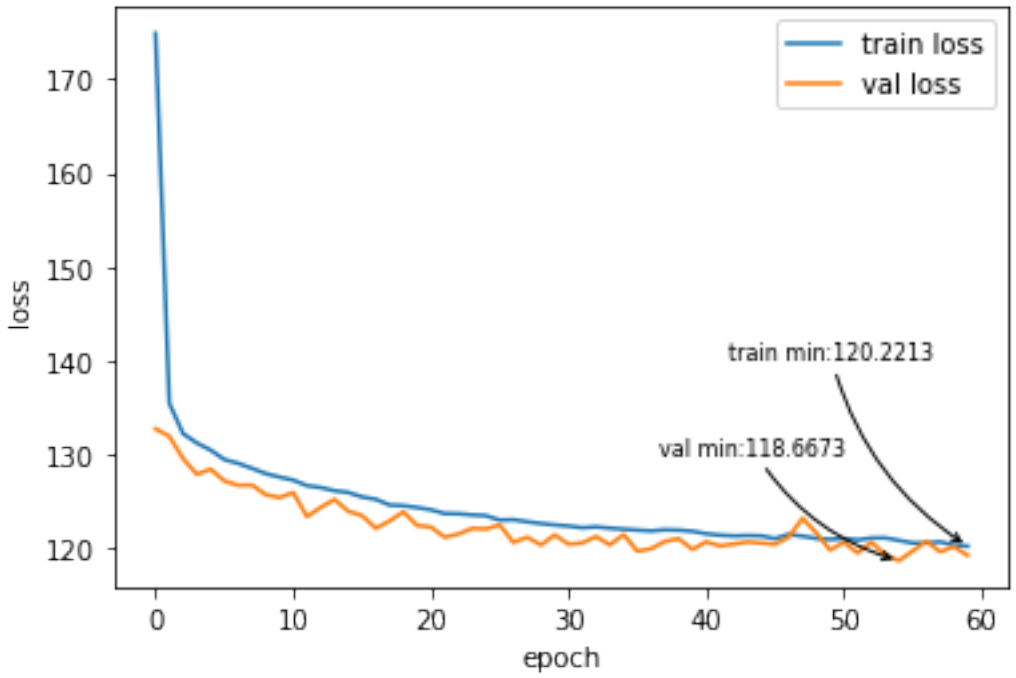
[10]:
```
#Baseline v5 (no Batchnorm) on Ptbxl 52 classes no pca, batch_size 128 new␣
 ↪accuracy function, simple loss, windowslength 9500
"""4 Layers of Convs with filter size 3 and dilation of 2^n to get a big␣
 ↪receptive field
Summarizes all channels using conv1x1
Fully connected has to over 9400 inputs!
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/12_01_21-19')
plot(train_losses, val_losses, train_acc, val_acc)
```

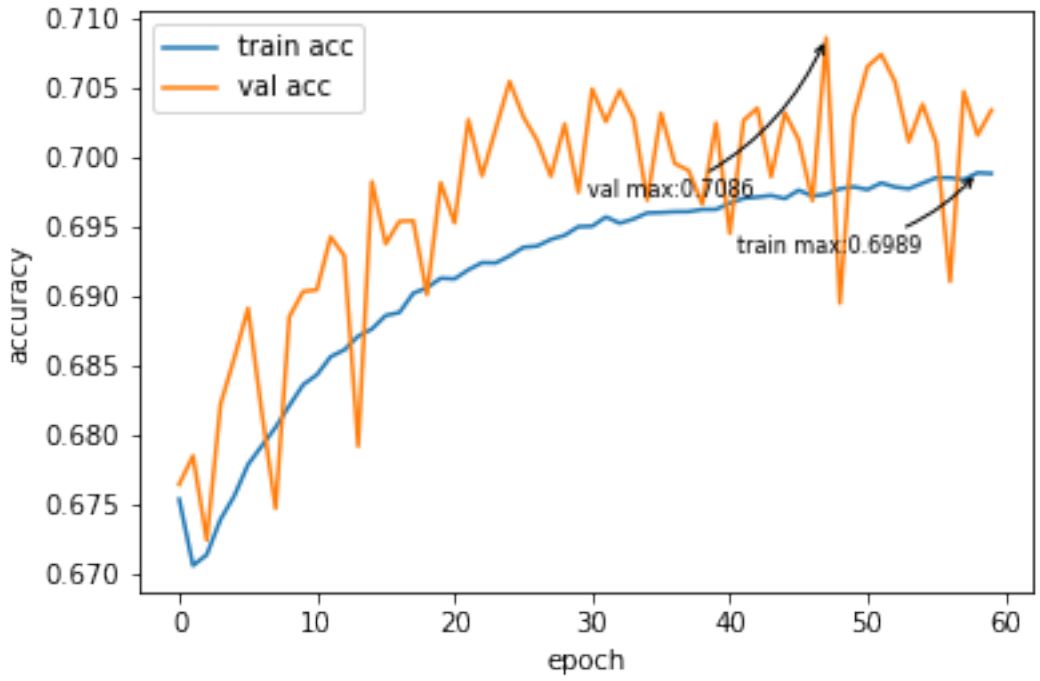




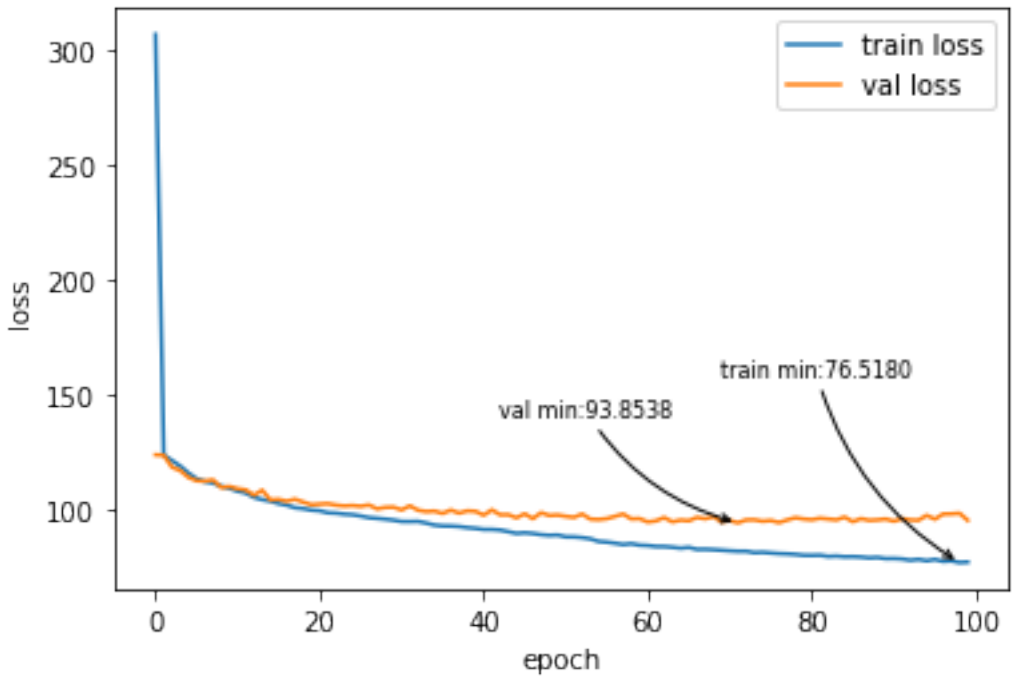
[11]:
```
#DO NOT RERUN
```

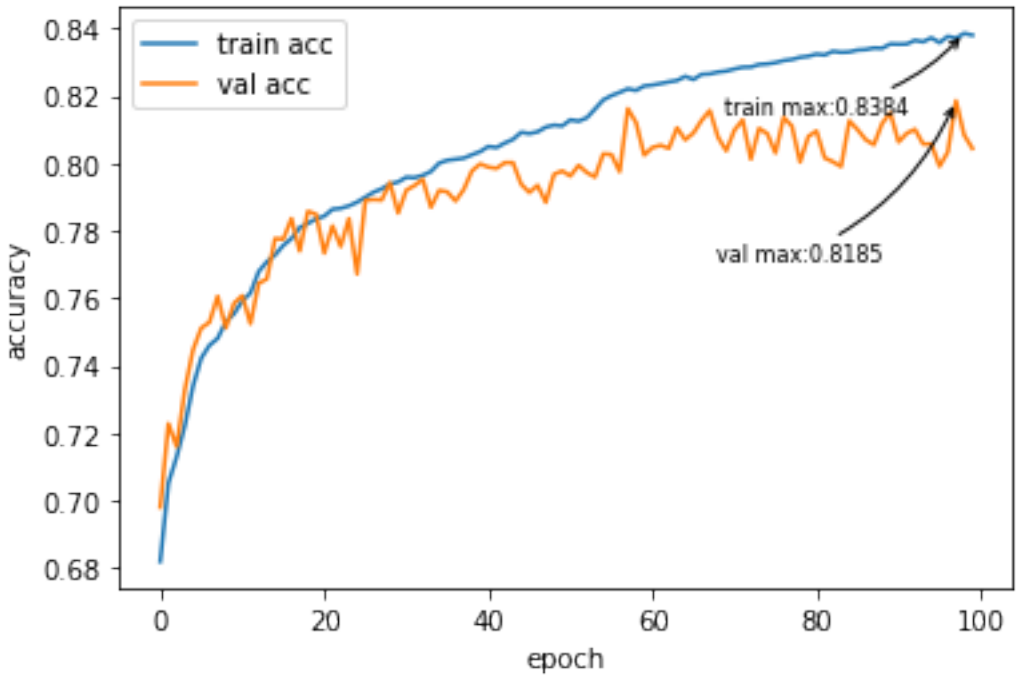
```
"""4 Layers of Convs with filter size 3 and dilation of 2^n to get a big␣
 ↪receptive field
left over maxpool not removed so incorrect
Summarizes all channels using conv1x1
Fully connected has to over 9400 inputs!
"""
#Baseline v5 (no Batchnorm/ false maxpool) on Ptbxl 52 classes no pca,␣
 ↪batch_size 128 new accuracy function, simple loss, windowslength 9500
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/12_01_21-19')
plot(train_losses, val_losses, train_acc, val_acc)
```





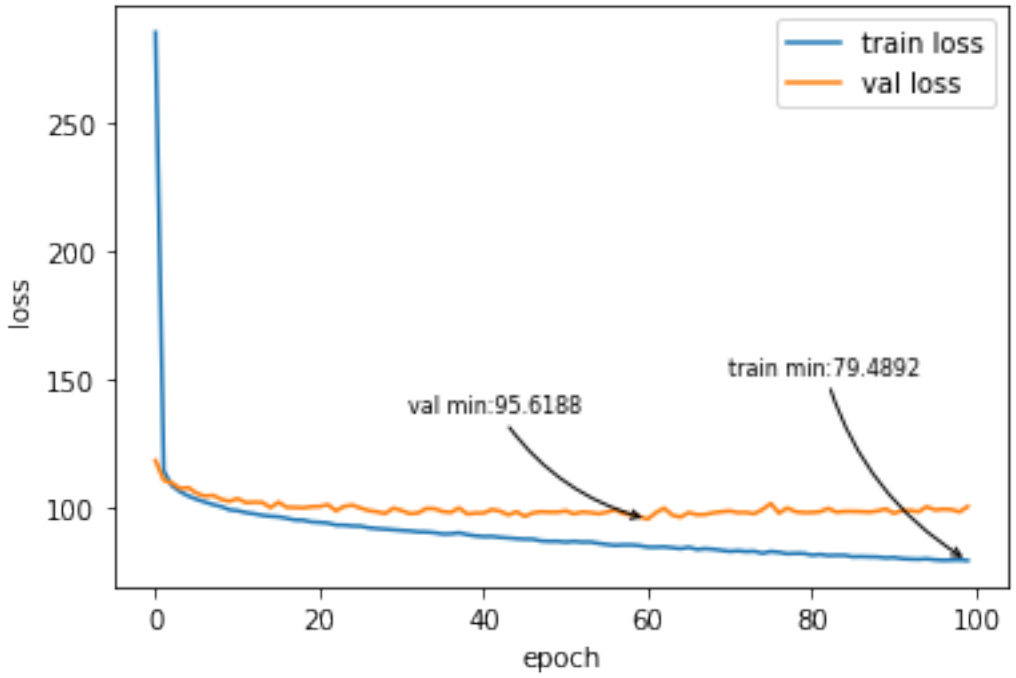
[9]:
```
#Baseline v4 (no Batchnorm v3) on Ptbxl 52 classes no pca, batch_size 128 new␣
 ↪accuracy function, simple loss, windowslength 9500
"""
4 Layers of Convs with filter size 3 and dilation of 2^n to get a big receptive␣
 ↪field
Summarizes all values using transpose + conv1x1
Full connected has only out_channels as input
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/12_01_21-18')
plot(train_losses, val_losses, train_acc, val_acc)
```

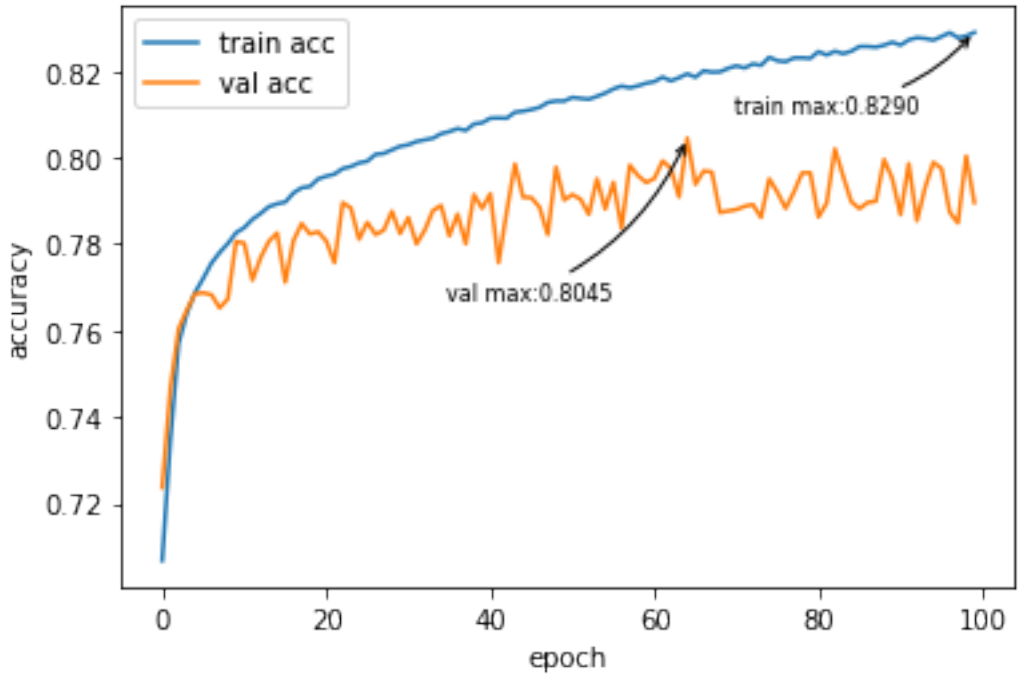




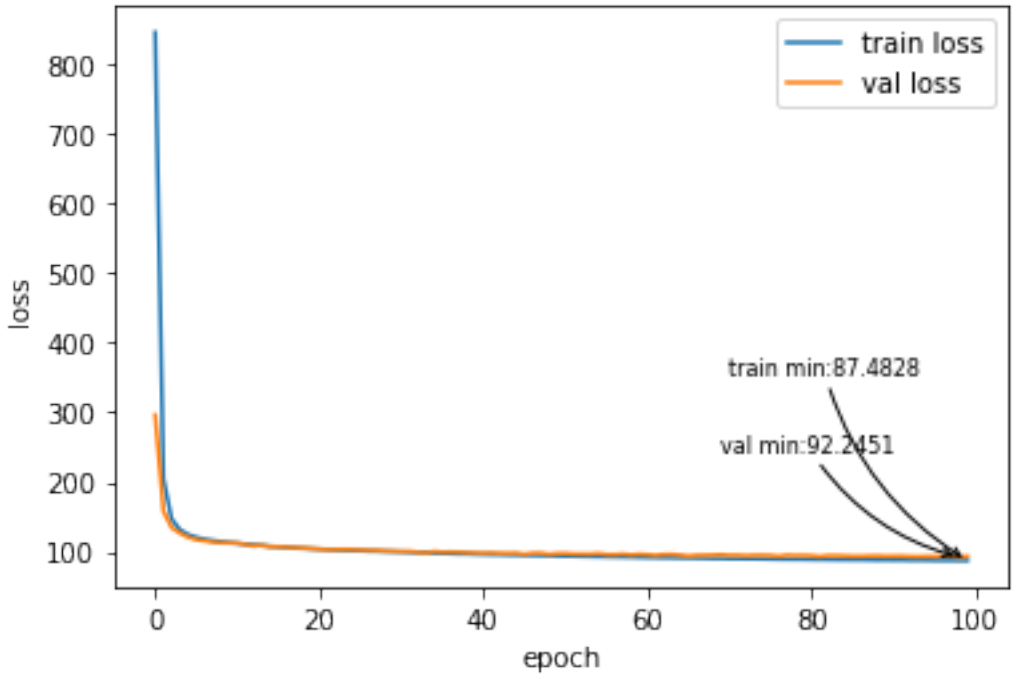
[4]:
```
#Baseline v3 on Ptbxl 52 classes no pca, batch_size 128 new accuracy function,␣
↪simple loss, windowslength 9500
```

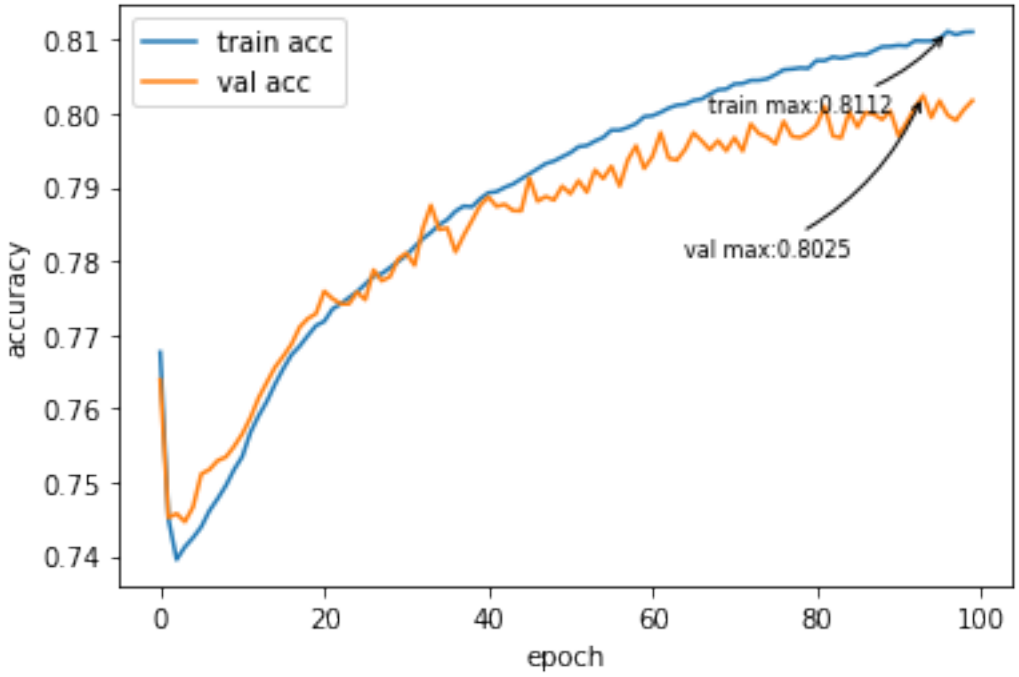
```
"""
4 Layers of Convs with filter size 3 and dilation of 2^n to get a big receptive␣
 ↪field
Batchnorm after every Layer before ReLU
Summarizes all values using transpose + conv1x1
Full connected has only out_channels as input
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/12_01_21-17')
plot(train_losses, val_losses, train_acc, val_acc)
```





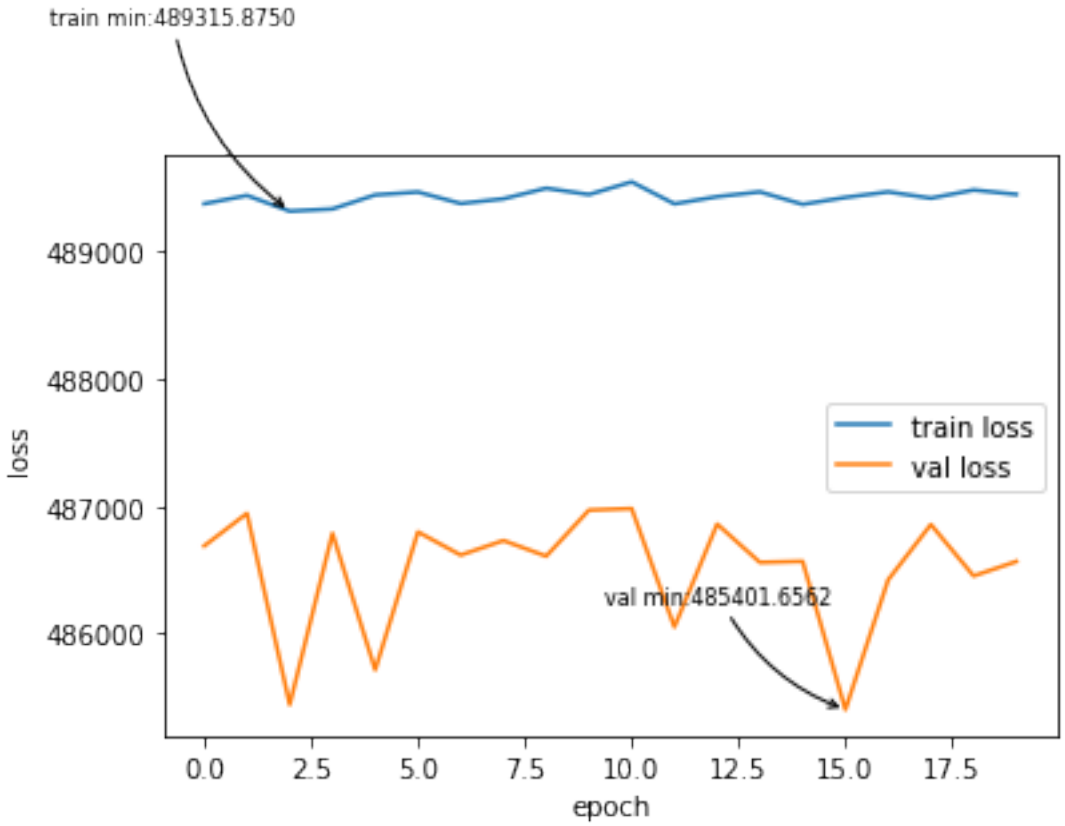
[3]:
```
#Baseline v2 on Ptbxl 52 classes no pca, batch_size 24 new accuracy function, simple loss, windowslength 9500
"""
5 Layers of Convs with filter size 3 and dilation of 2^n to get a big receptive field
Max Poolwith size 3 after every Layer to downsample
Pools all values at the end using Adaptive Average Pooling
Full connected has only out_channels as input
"""
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/Downloads/Github/contrastive-predictive-coding/models/12_01_21-15')
plot(train_losses, val_losses, train_acc, val_acc)
```

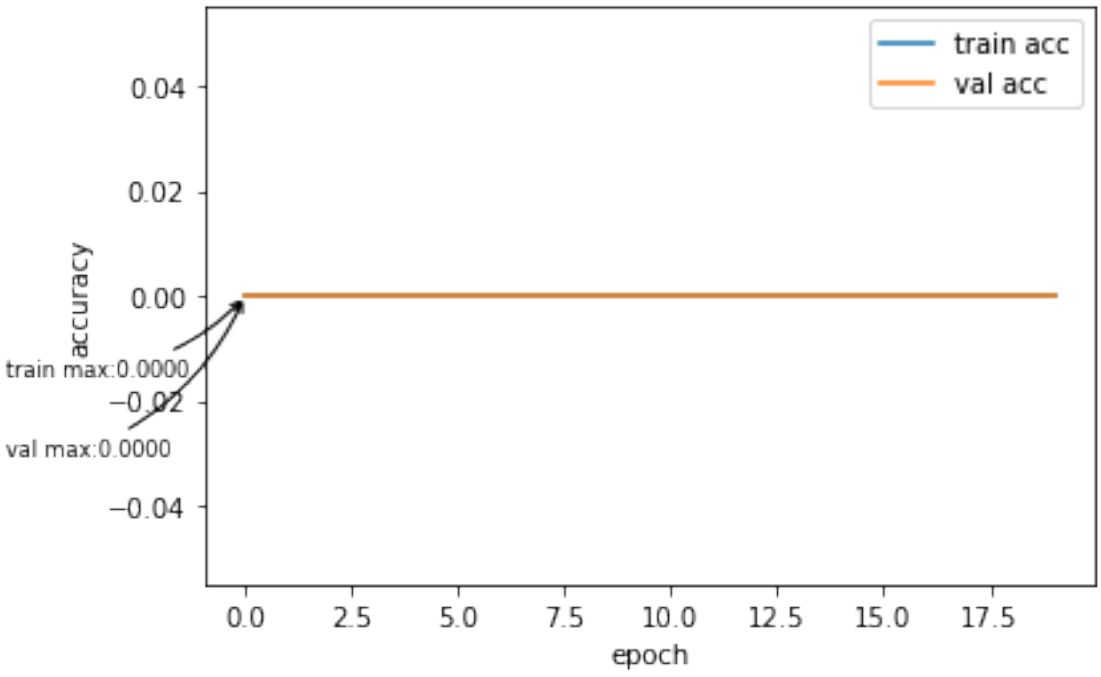




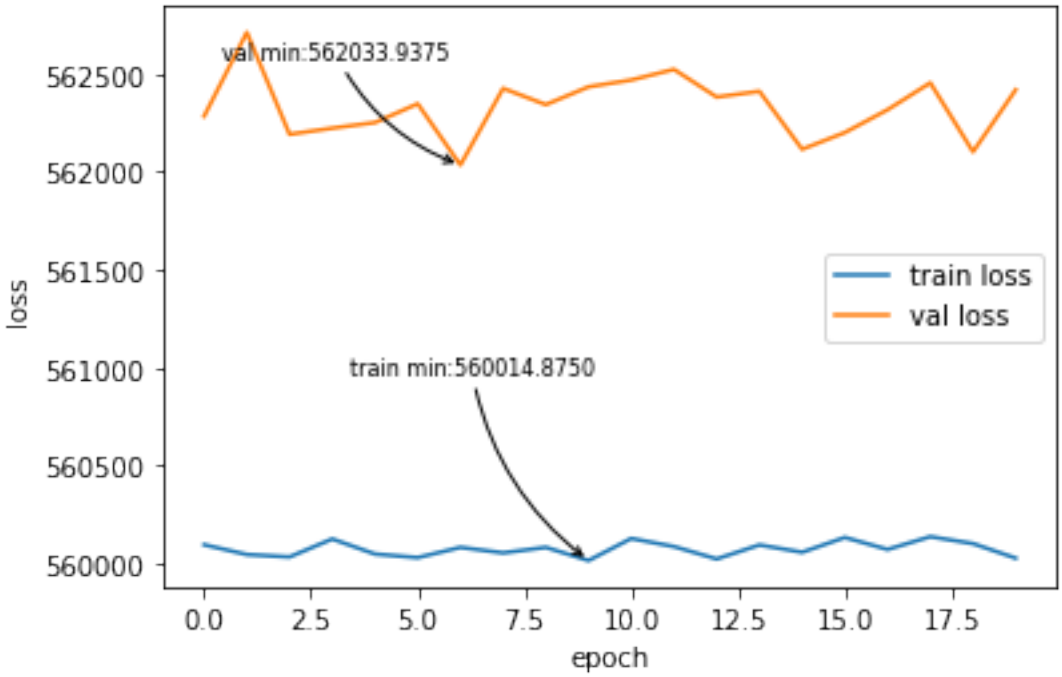
[6]:
```
#Baseline Autoencoder on Ptbxl 52 classes no pca, batch_size 24 new accuracy␣
↪function, simple loss, windowslength 512, latents 256
```

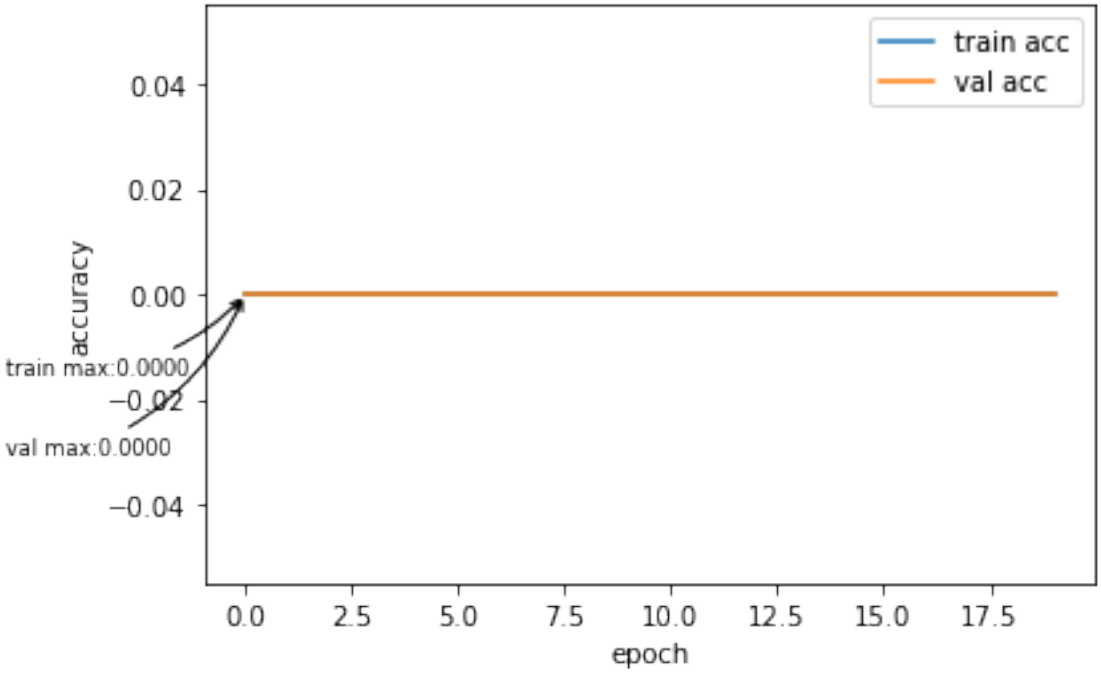
```
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/06_01_21-16')
plot(train_losses, val_losses, train_acc, val_acc)
```





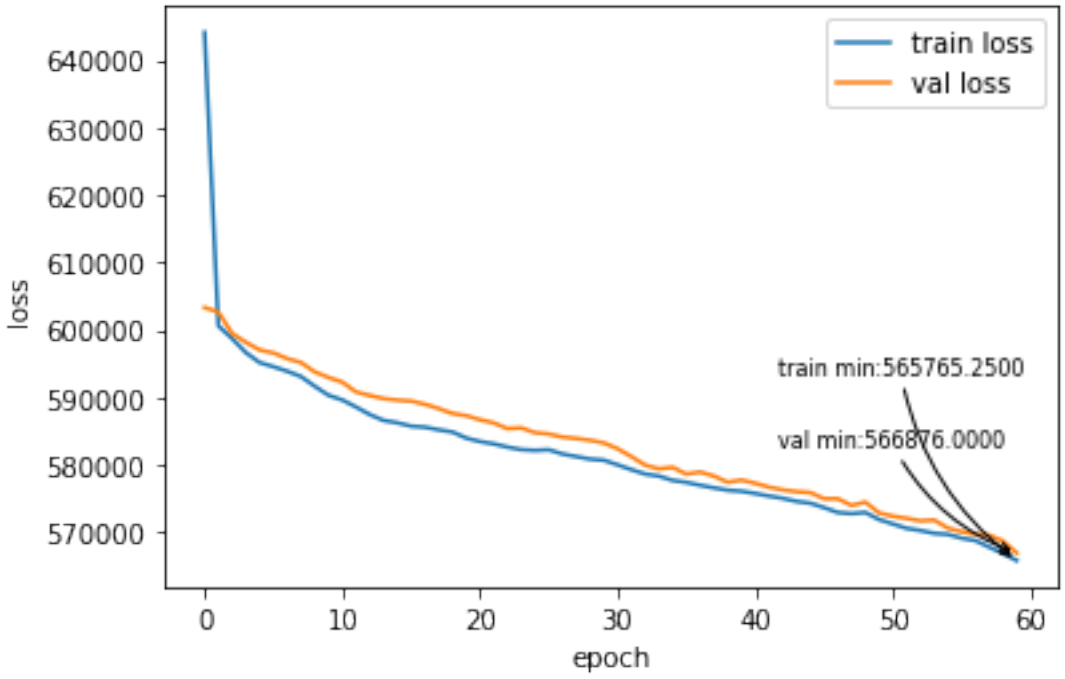
[4]:
```
#Baseline Autoencoder on Ptbxl 52 classes no pca, batch_size 24 new accuracy␣
↪function, simple loss, windowslength 9500
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/06_01_21-13')
plot(train_losses, val_losses, train_acc, val_acc)
```

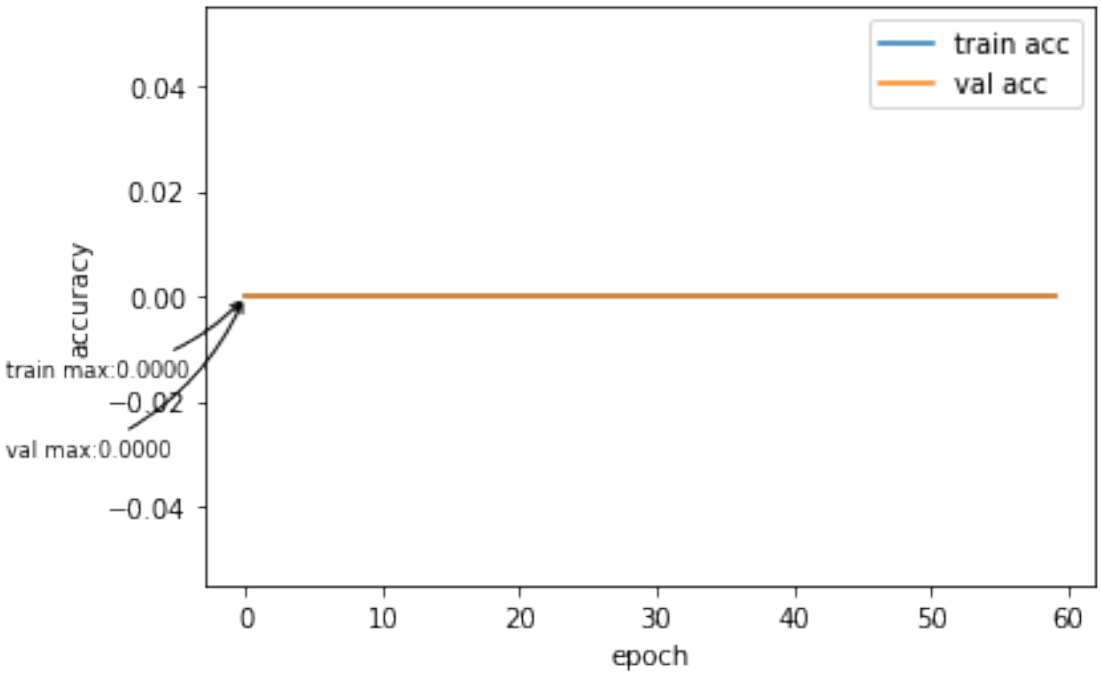




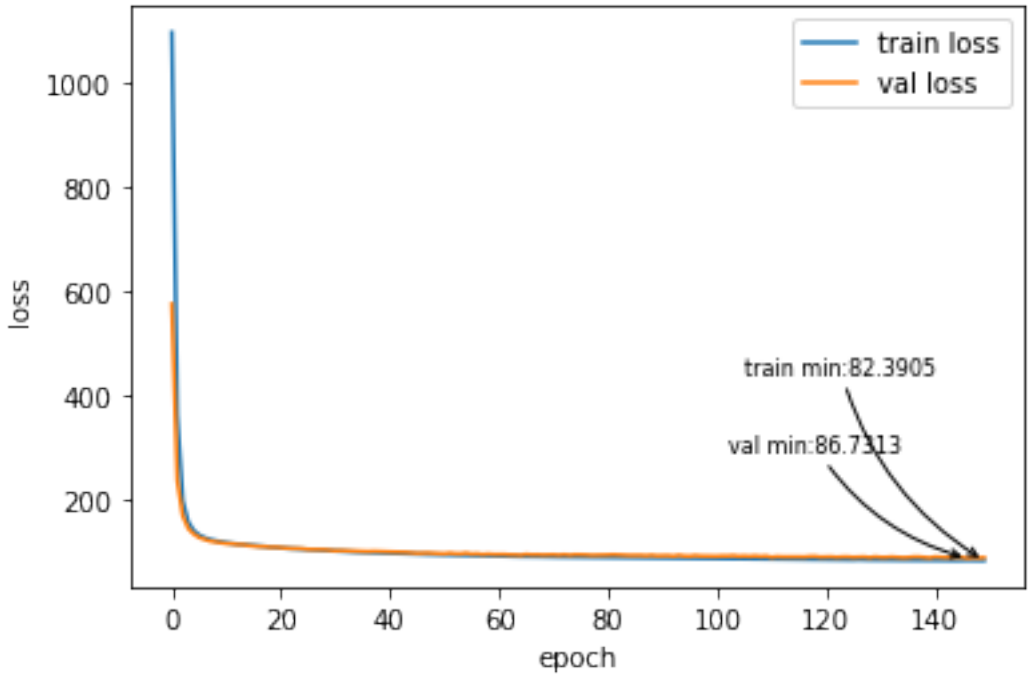
[3]:
```
#Autoencoder on Ptbxl 52 classes no pca, batch_size 24 new accuracy function,␣
↪simple loss
```

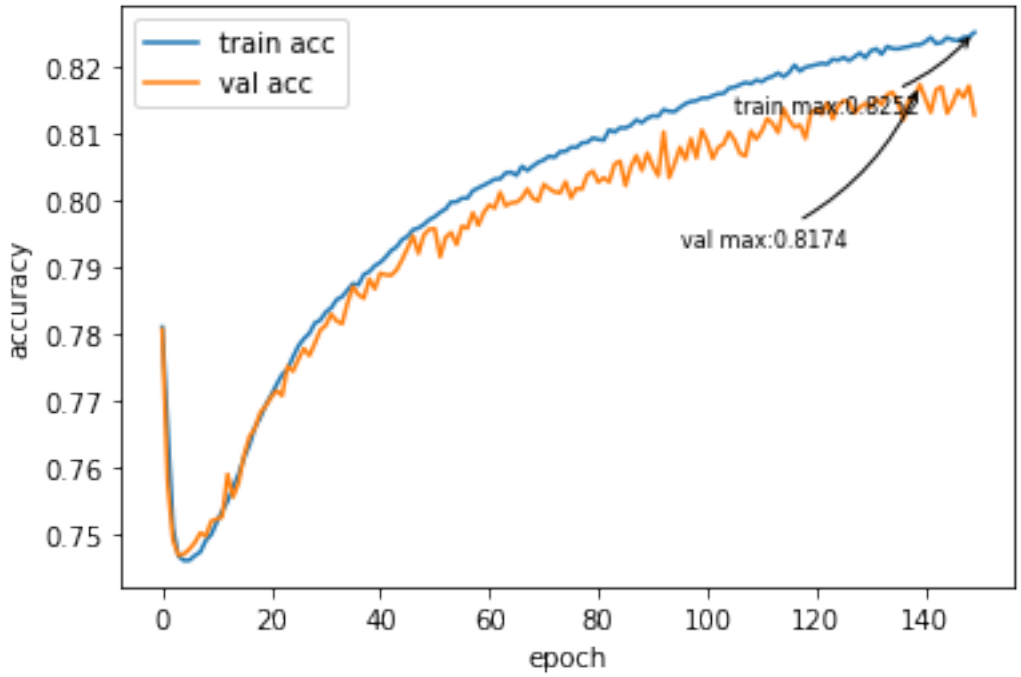
```
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/04_01_21-18')
plot(train_losses, val_losses, train_acc, val_acc)
```



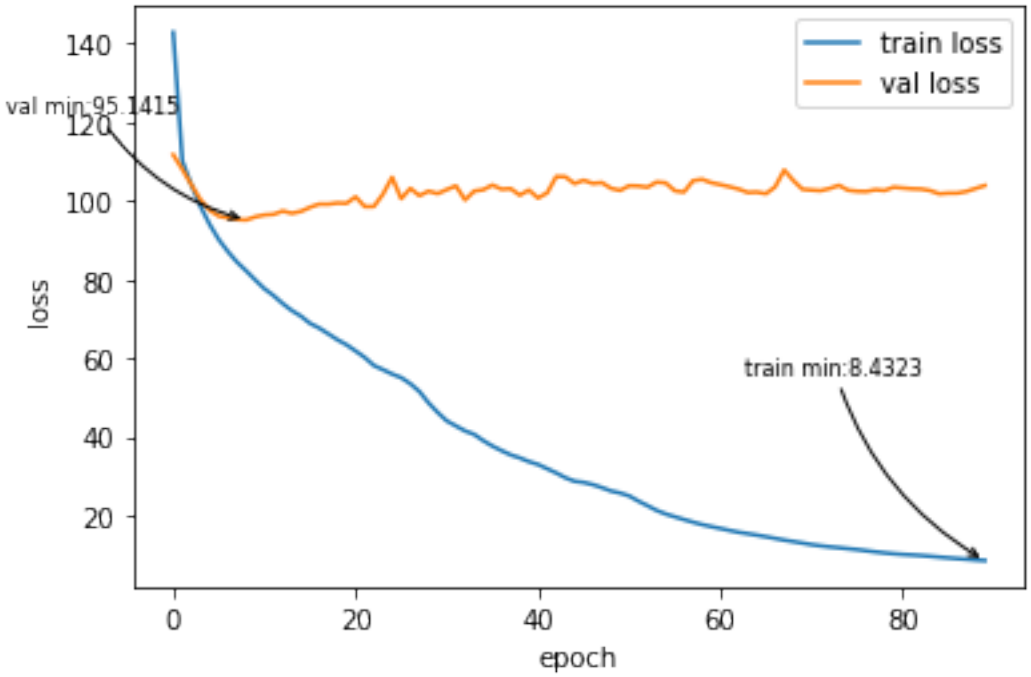


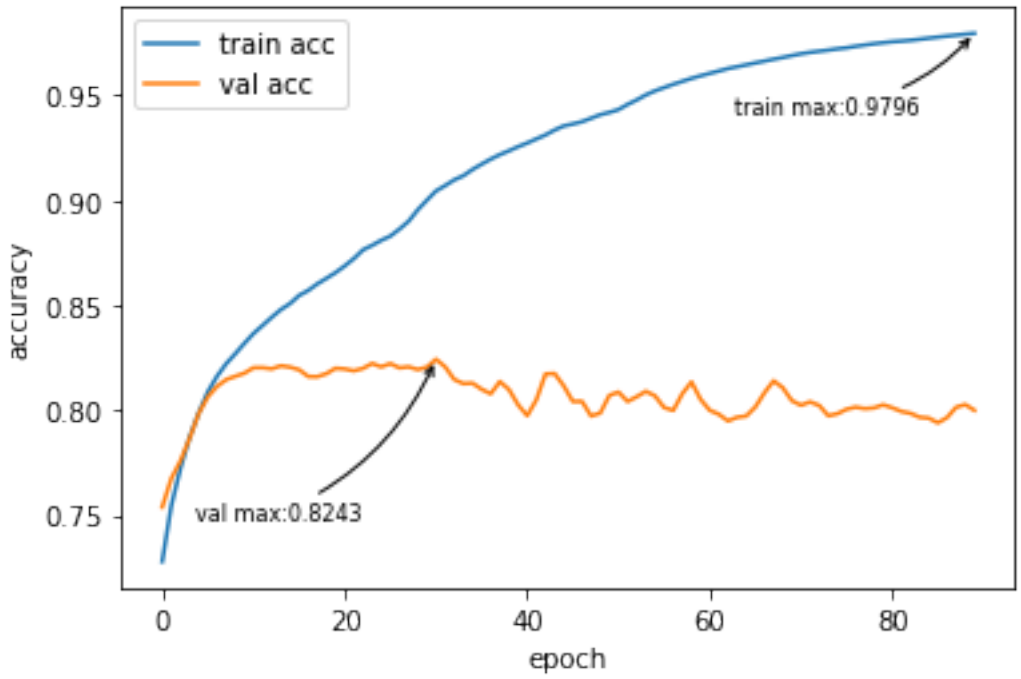
[11]:
```
#Baseline on Ptbxl 52 classes no pca, batch_size 24 new accuracy function, simple loss
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/Downloads/Github/contrastive-predictive-coding/models/03_01_21-19')
plot(train_losses, val_losses, train_acc, val_acc)
```





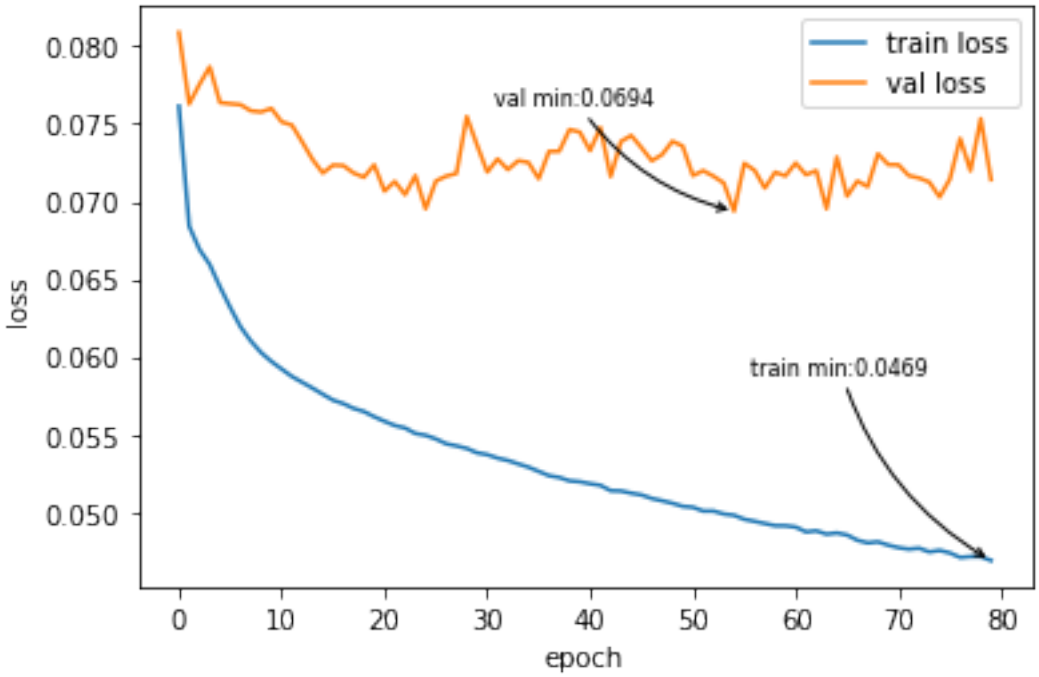
[9]:
```
#Downstream on Ptbxl 52 classes no pca, batch_size 24 new accuracy function, simple loss
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/Downloads/Github/contrastive-predictive-coding/models/03_01_21-18')
plot(train_losses, val_losses, train_acc, val_acc)
```

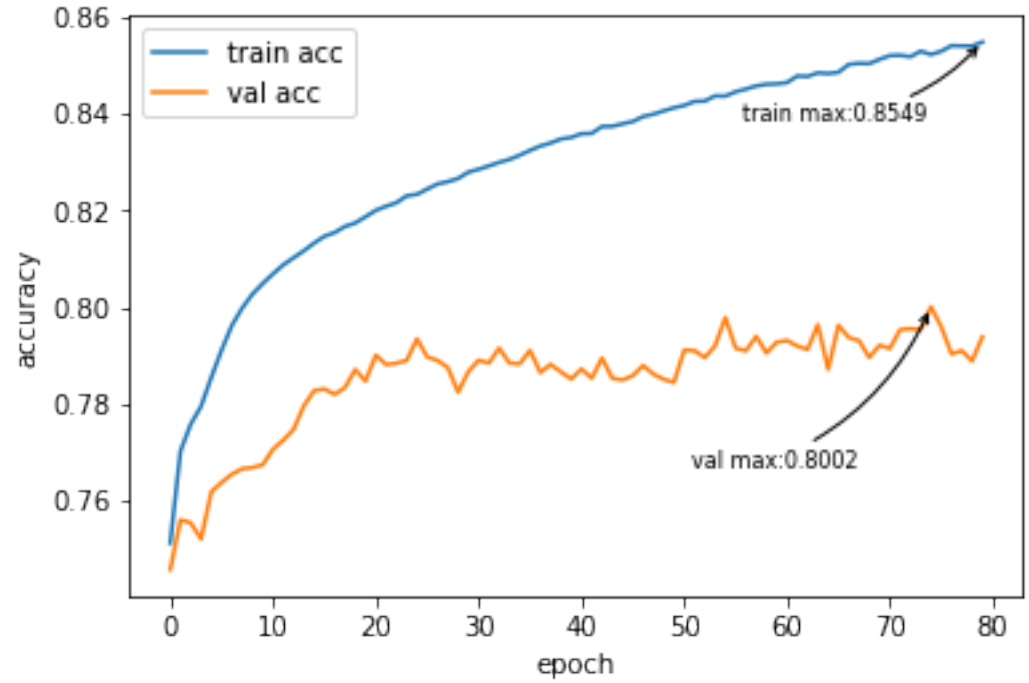




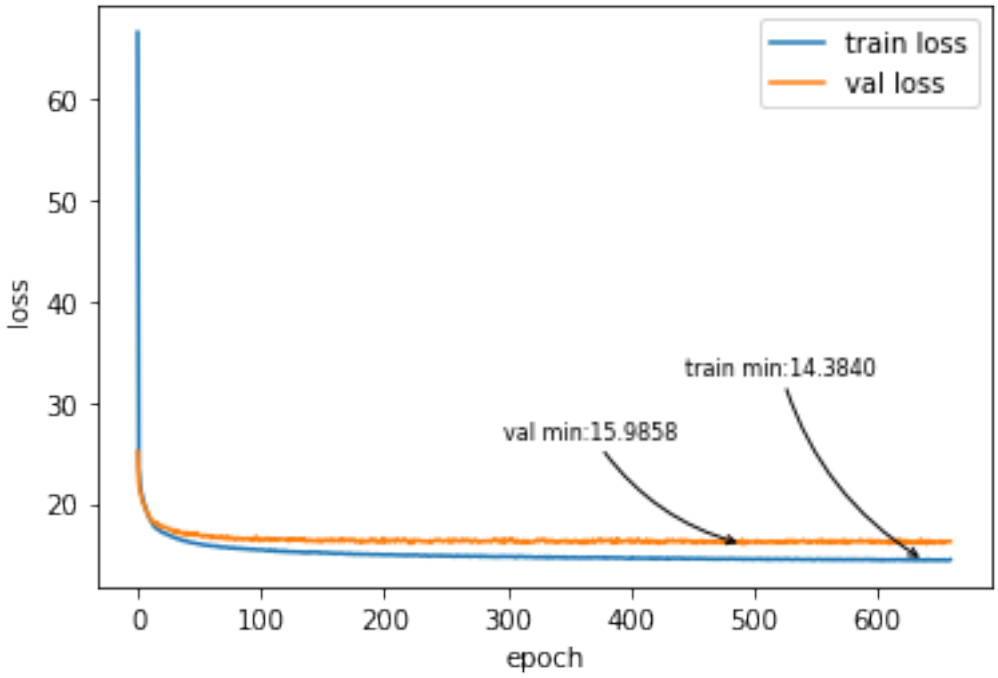
[4]:
```
#Downstream on Ptbxl 52 classes no pca, batch_size 24 new accuracy function,␣
↪BCELoss
```

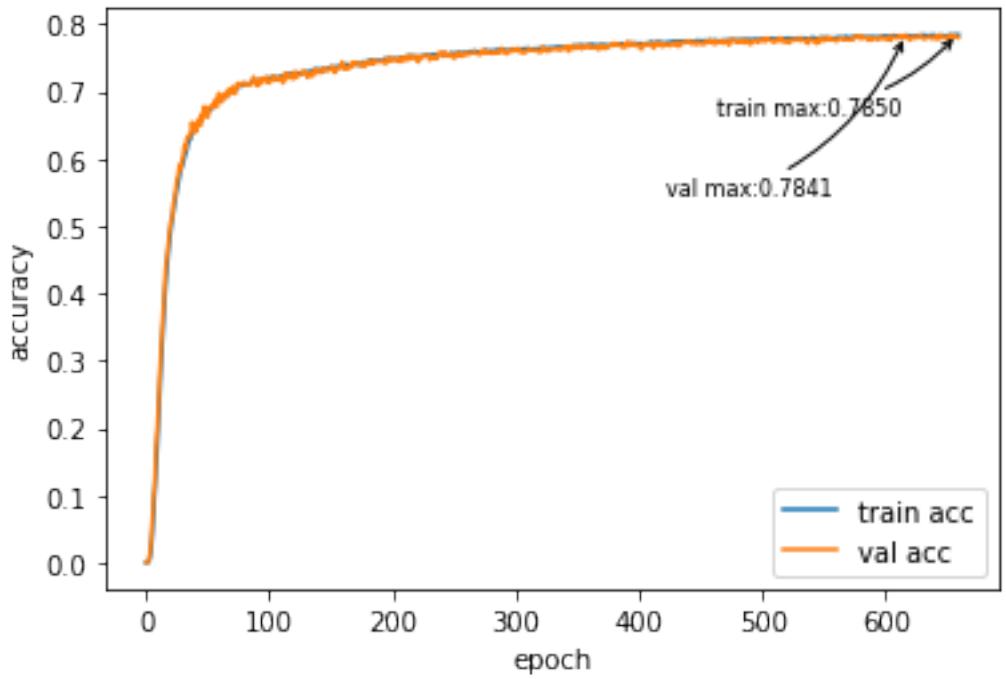
```
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/03_01_21-17')
plot(train_losses, val_losses, train_acc, val_acc)
```



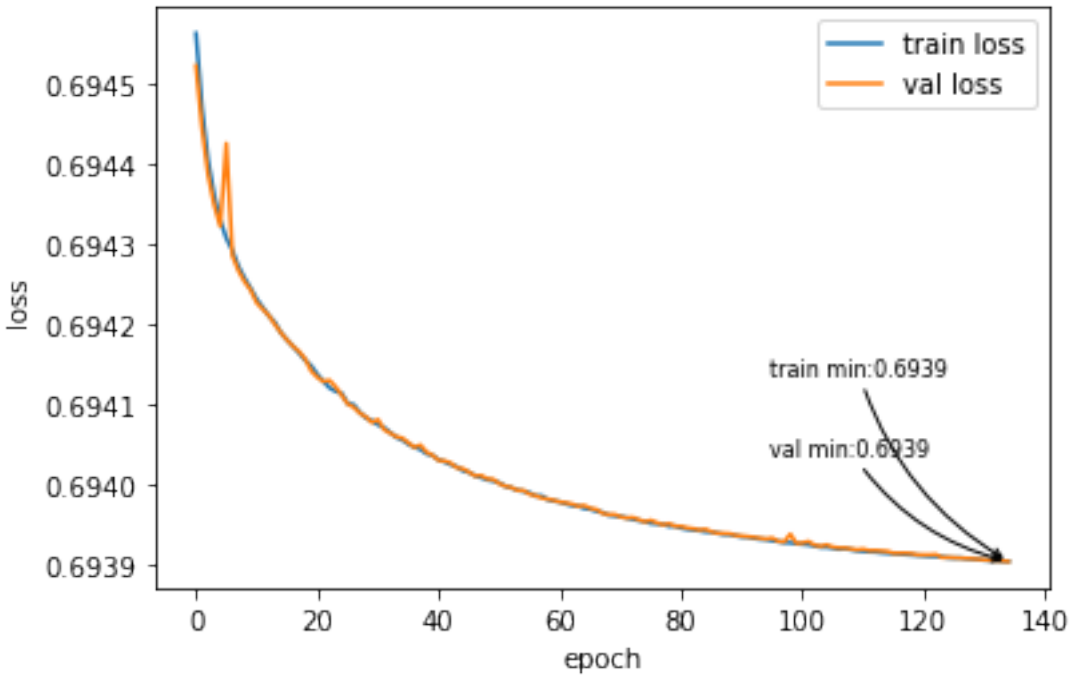


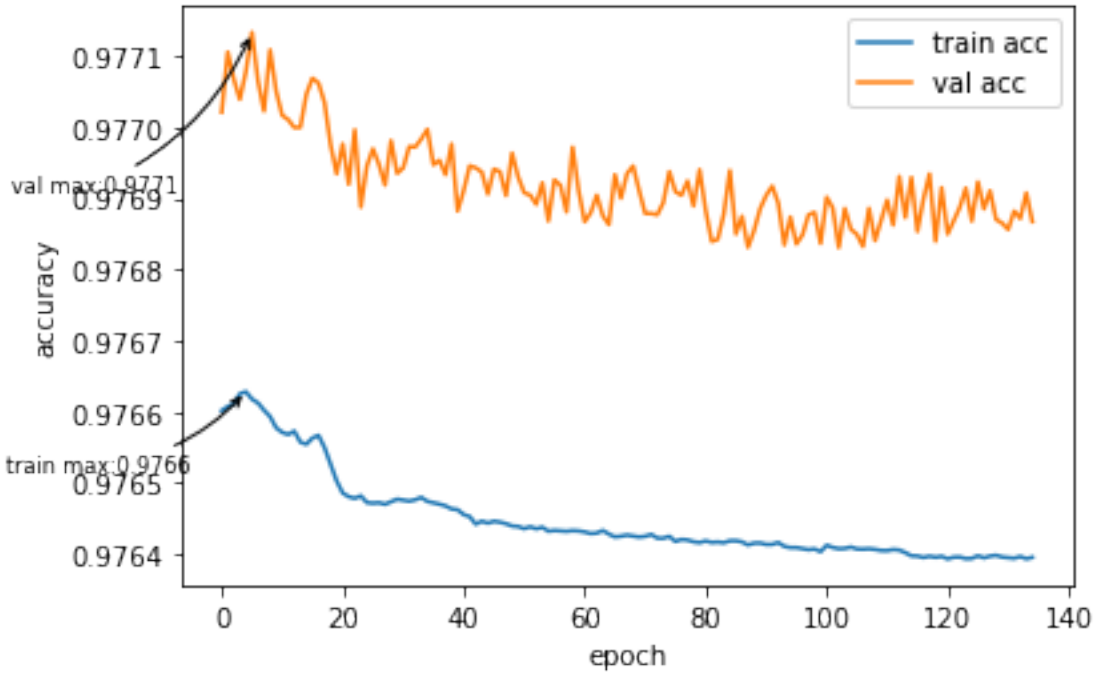
[26]:
```
#BASELINE on Ptbxl 52 classes, 6 Layers of Convs, no pca, batch_size 24 new␣
↪accuracy function, MultiLabelSoftMarginLoss?
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/18_12_20-17')
plot(train_losses, val_losses, train_acc, val_acc)
```



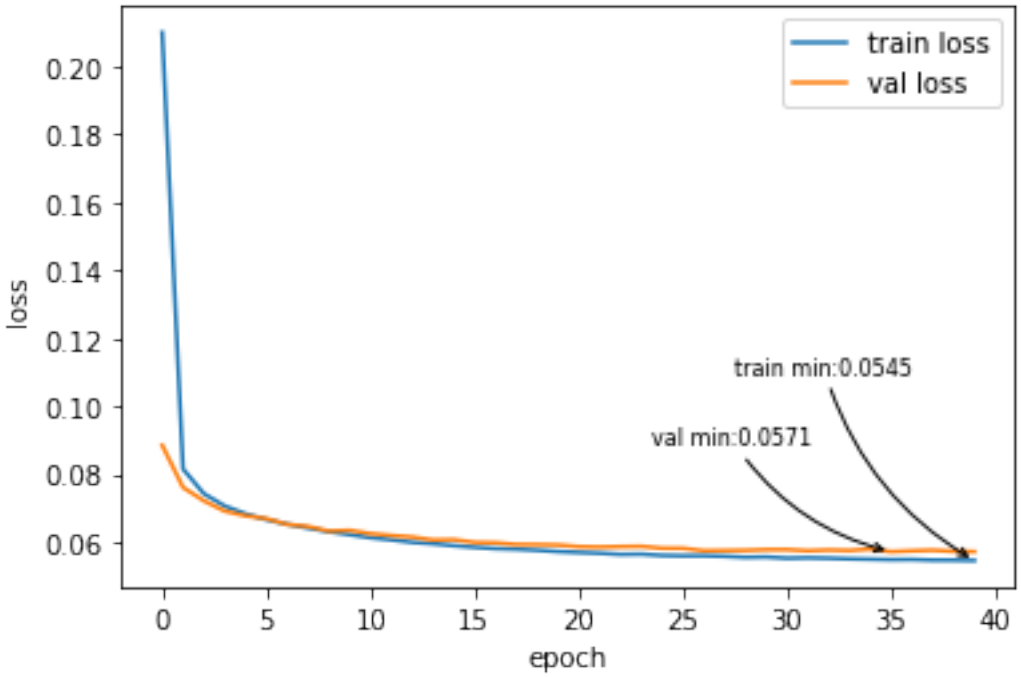


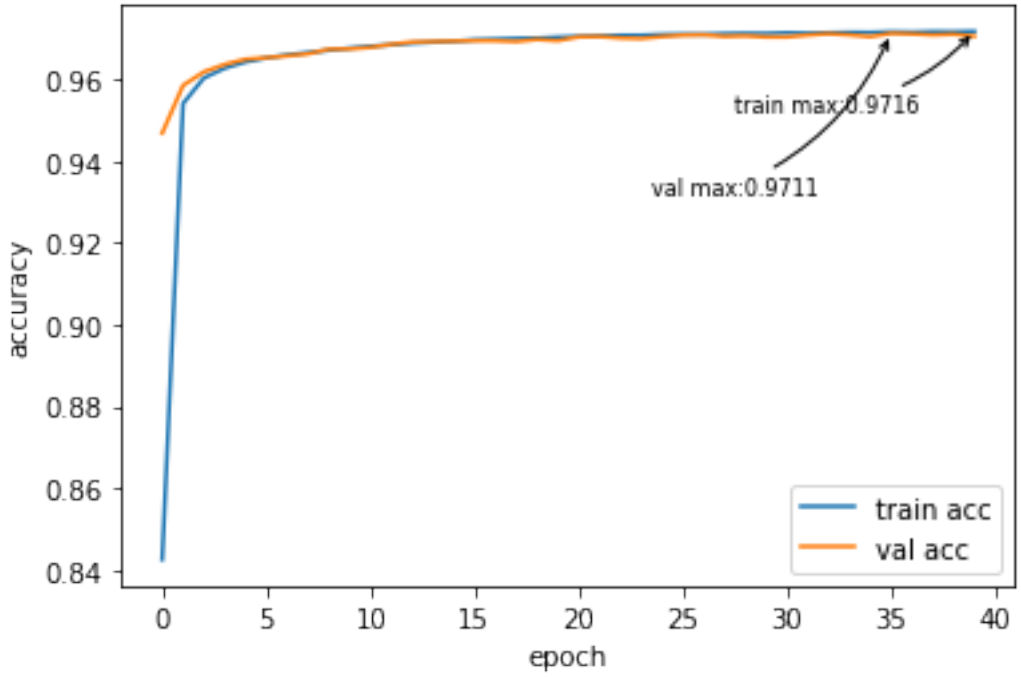
[17]:
```
#BASELINE on Ptbxl 52 classes, 6 Layers of Convs, no pca, batch_size 24
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/18_12_20-15')
plot(train_losses[5:], val_losses[5:], train_acc[5:], val_acc[5:])
```





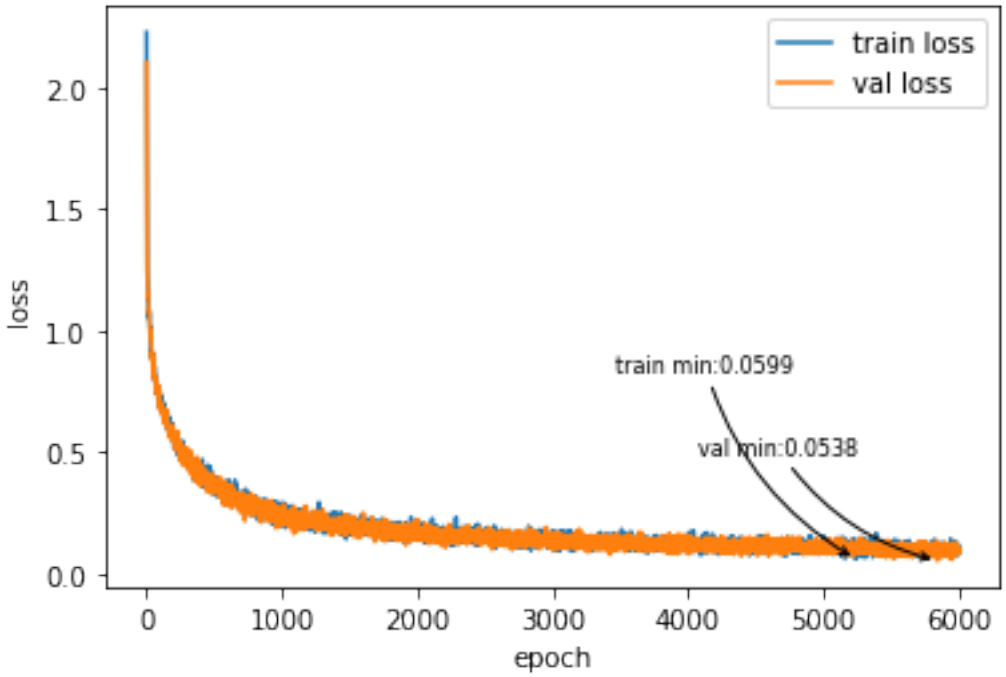
[6]:
```
#BASELINE on Ptbxl 52 classes, 6 Layers of Convs, no pca, batch_size 24
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/18_12_20-14')
plot(train_losses, val_losses, train_acc, val_acc)
```

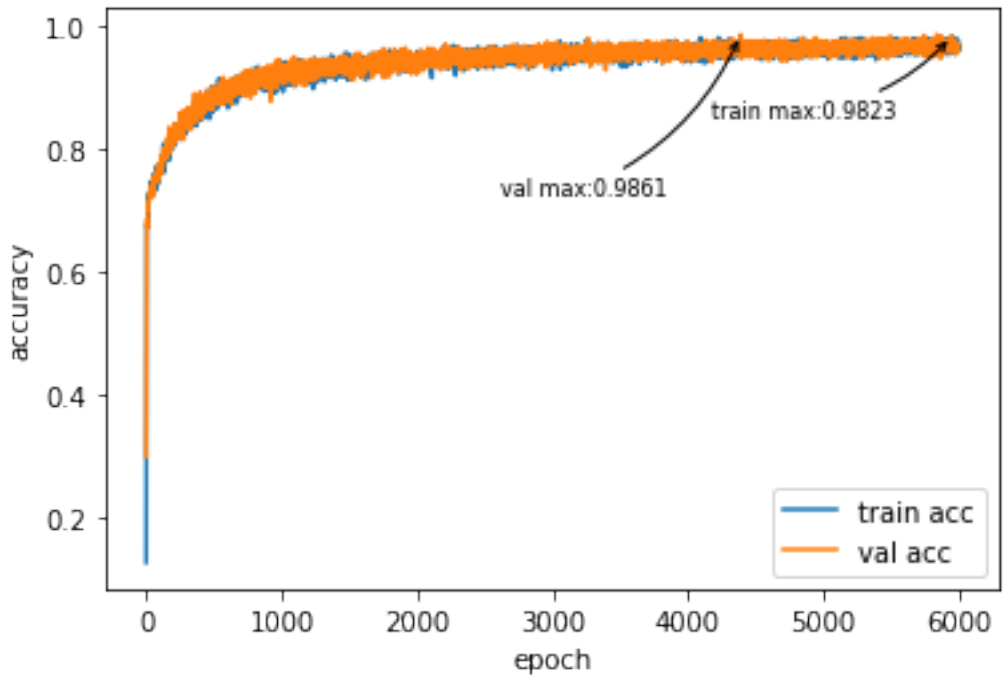

train loss
val loss
loss
epoch
train min:0.0545
val min:0.0571


train acc
val acc
accuracy
epoch
train max:0.9716
val max:0.9711

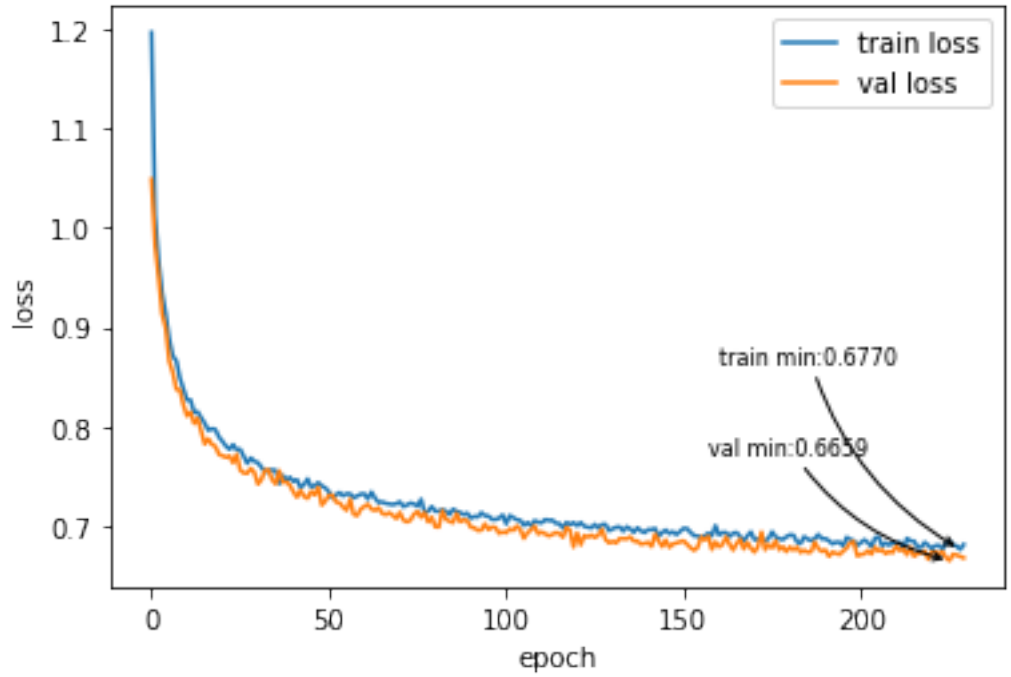
0.2 Losses for single target

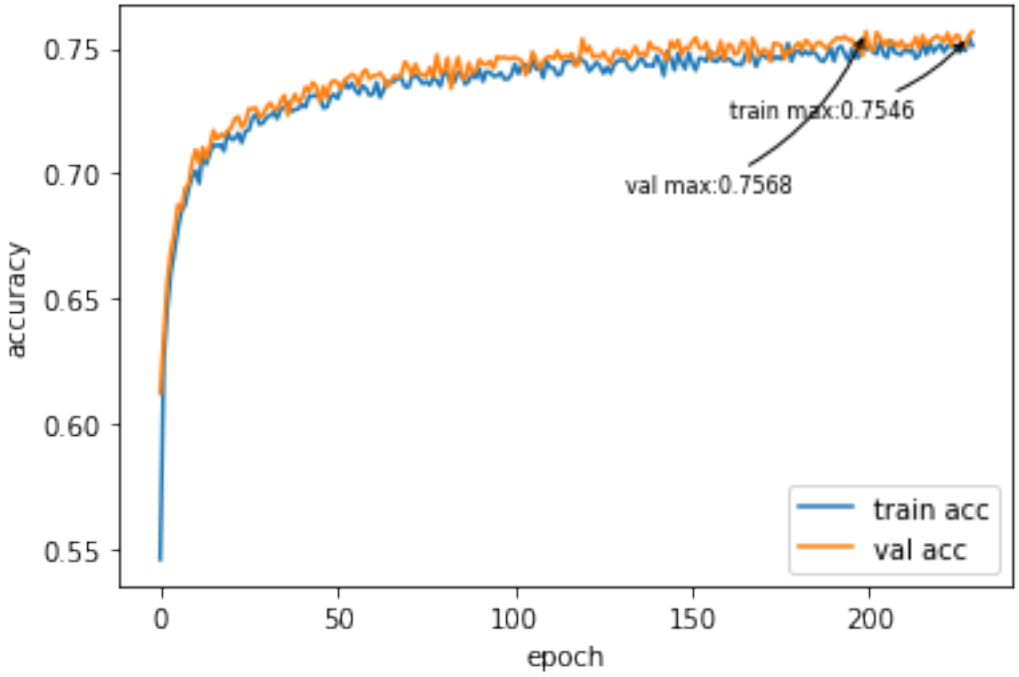
[41]:
```
#BASELINE on Ptb, 6 Layers of Convs, no pca, batch_size 24 #WRONG!!!␣
↪train=valset
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/15_12_20-14')
plot(train_losses, val_losses, train_acc, val_acc)
```





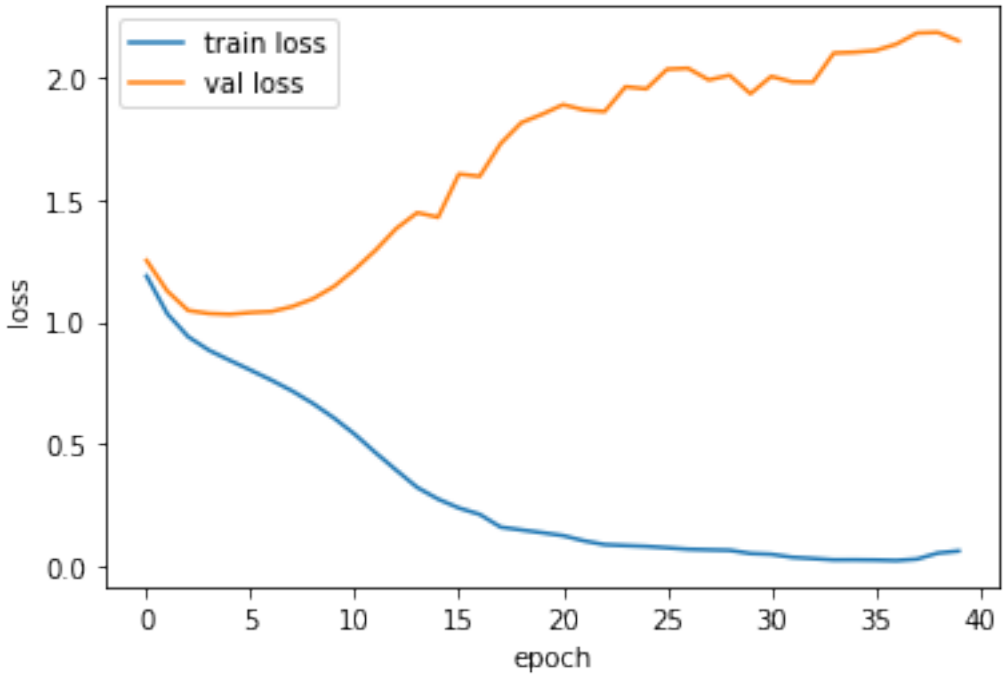
[33]:
```
#BASELINE on Ptbxl (correct split), 6 Layers of Convs, no pca, batch_size 24 ␣
↪#WRONG!!! train=valset
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/14_12_20-19')
plot(train_losses, val_losses, train_acc, val_acc)
```

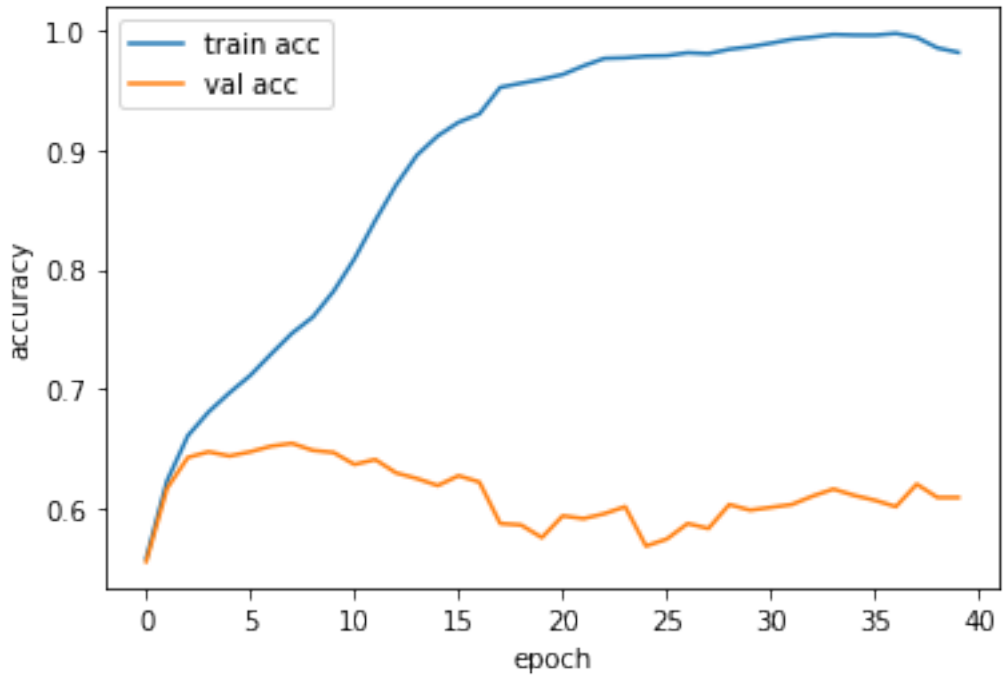




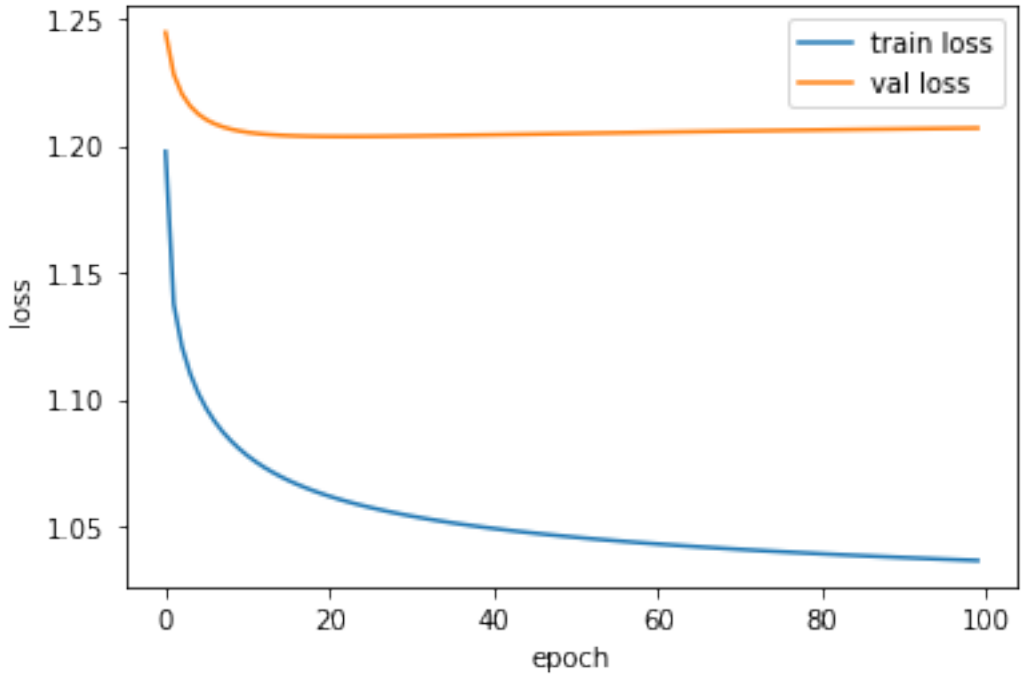
[13]:
```
#Downstream on PTBxl (correct split), ResNet, less windows 6-6, bigger context␣
↪512, no pca (12 channels), layers not? frozen, batch_size 12
```

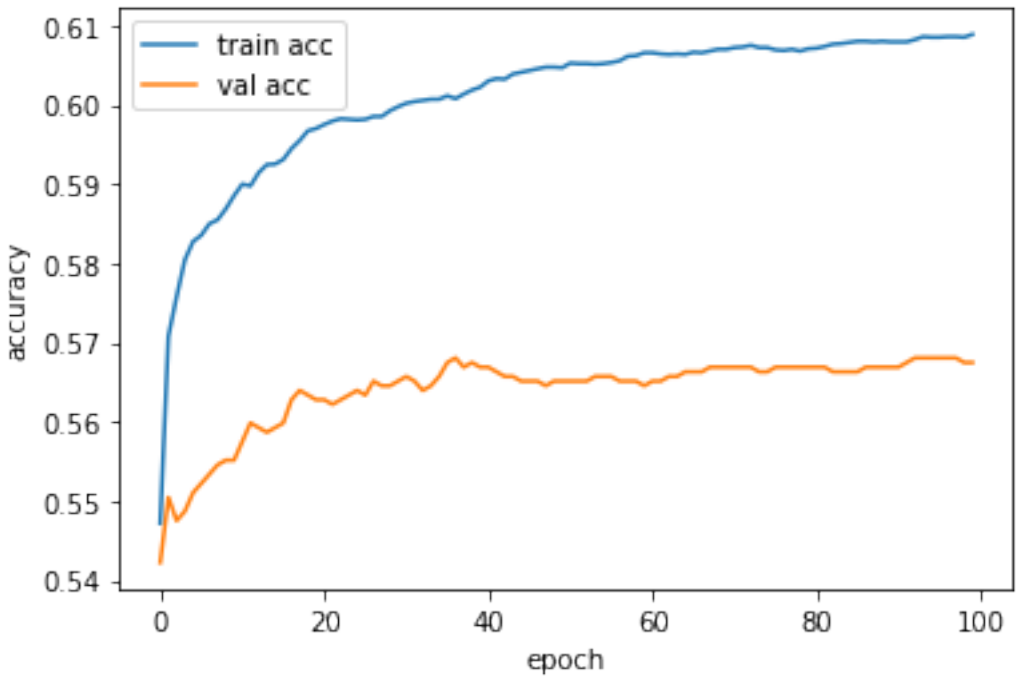
```
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/10_12_20-22')
plot(train_losses, val_losses, train_acc, val_acc)
```



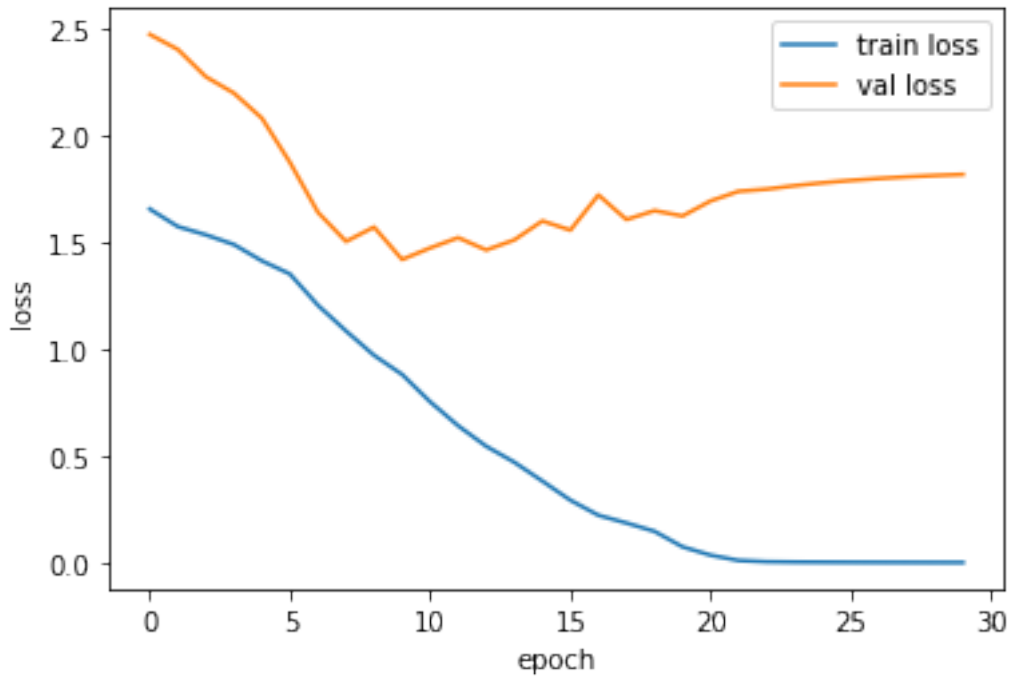


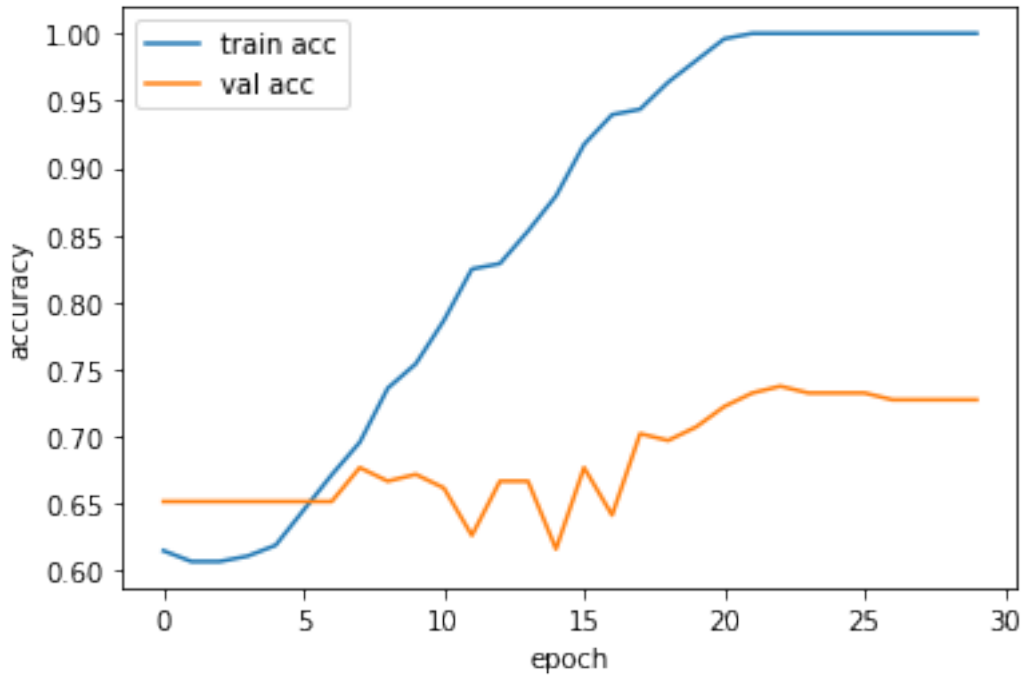
[11]:
```
#Downstream on PTBxl (correct split), ResNet, less windows 6-6, bigger context␣
↪512, no pca (12 channels), layers frozen, batch_size 12
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/10_12_20-21')
plot(train_losses, val_losses, train_acc, val_acc)
```





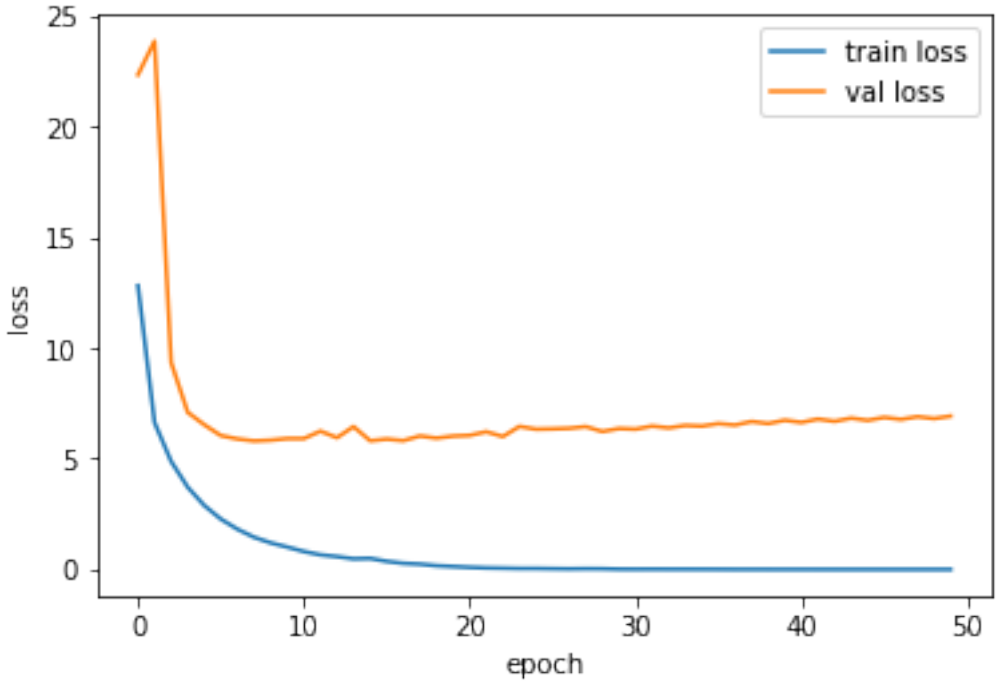
[9]:
```
#Downstream on PTB, ResNet, less windows 6-6, bigger context 512, no pca (12 channels), batch_size 1, layers NOT frozen
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/Downloads/Github/contrastive-predictive-coding/models/10_12_20-15')
plot(train_losses, val_losses, train_acc, val_acc)
```

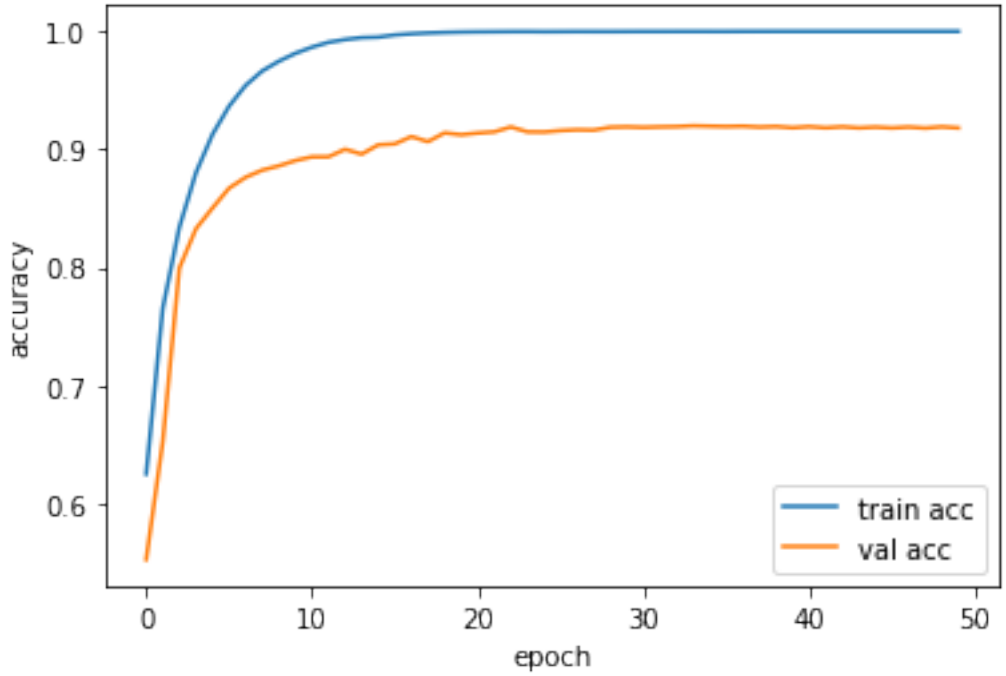




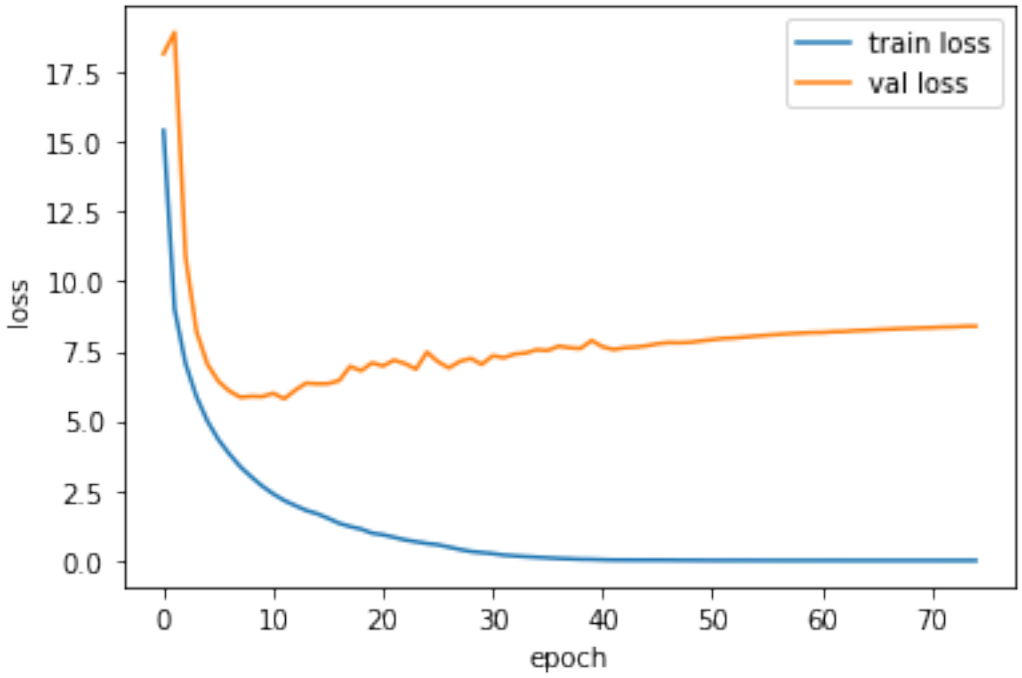
[5]:
```
#CPC on PTB, PTBxl (correct split), ResNet, less windows 6-6, bigger context␣
↪512, no pca (12 channels)
```

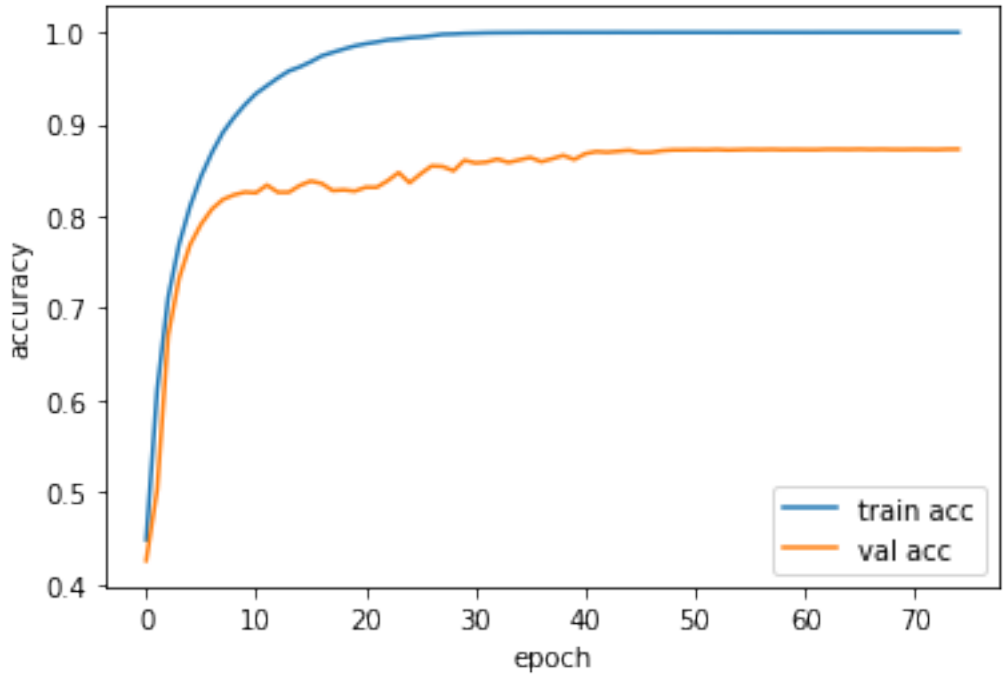
```
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/10_12_20-12')
plot(train_losses, val_losses, train_acc, val_acc)
```



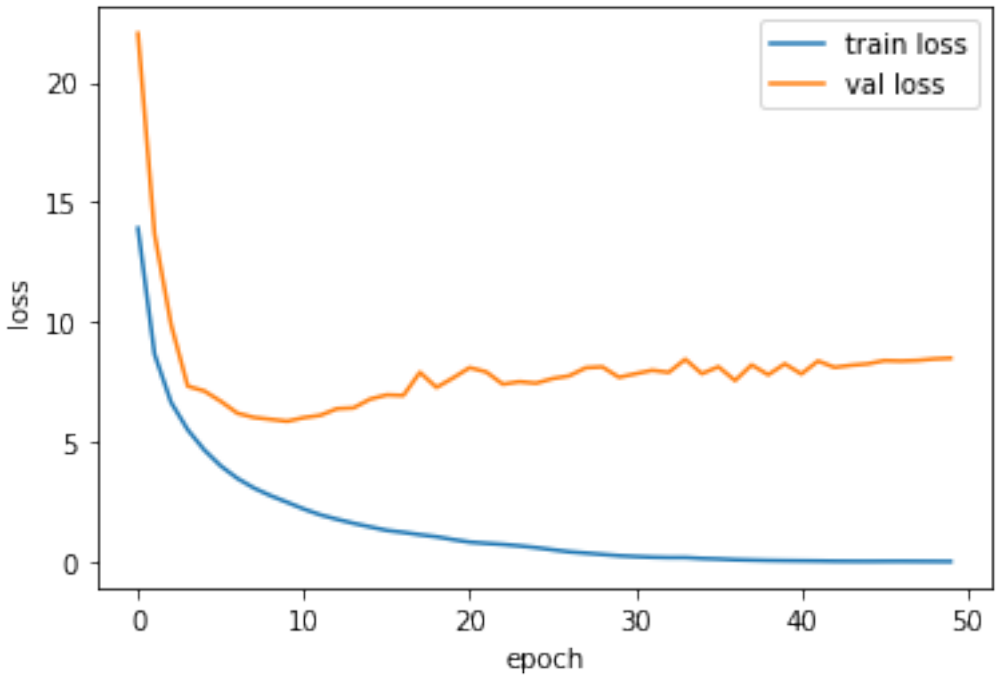


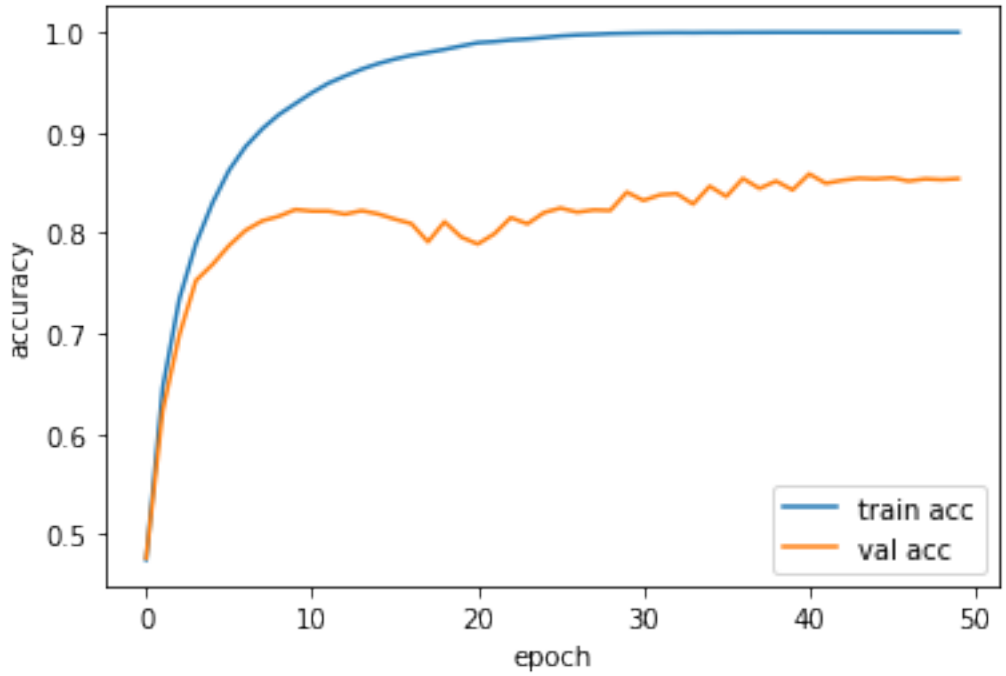
[9]:
```
#CPC on PTB, PTBxl (correct split), ResNet, less windows 6-6, bigger context␣
↪512, pca 2 channels
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/09_12_20-15')
plot(train_losses, val_losses, train_acc, val_acc)
```



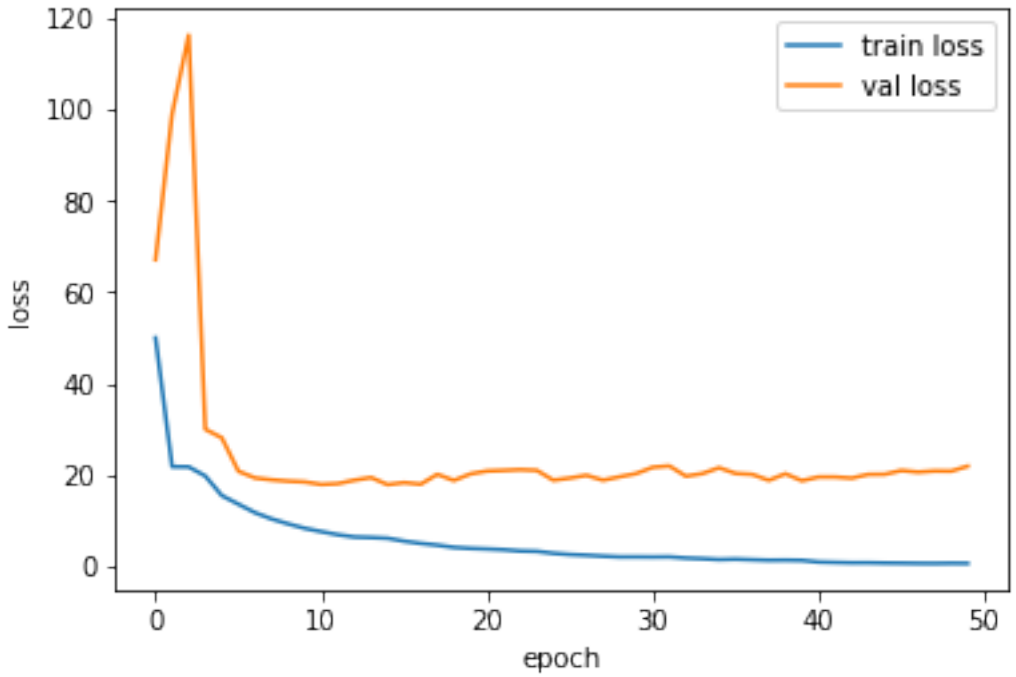


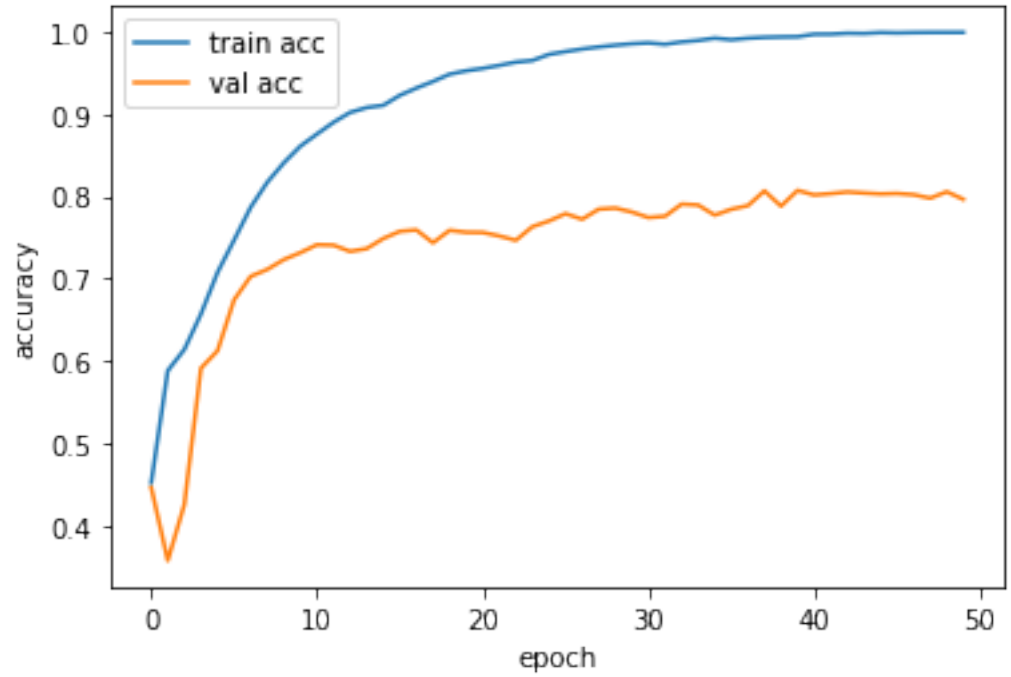
[5]:
```
#CPC on PTB, PTBxl (correct split), ResNet, less windows 6-6, pca 2 channels
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/09_12_20-14')
plot(train_losses, val_losses, train_acc, val_acc)
```





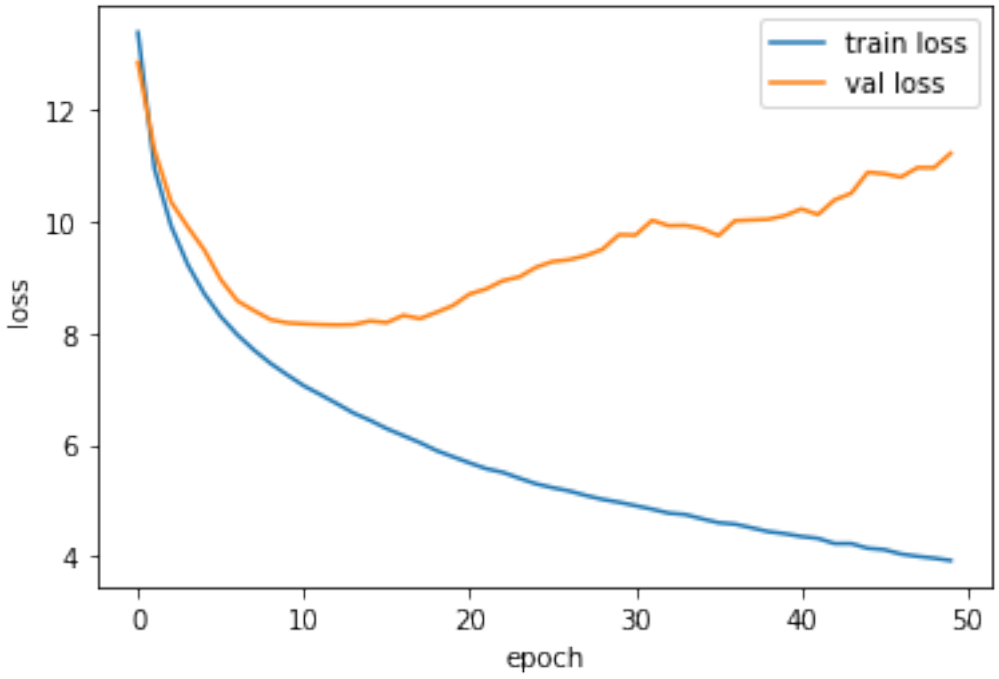
[4]:
```
#CPC on PTB, PTBxl (correct split), ResNet, many windows 12-12, pca 2 channels
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/09_12_20-13')
plot(train_losses, val_losses, train_acc, val_acc)
```

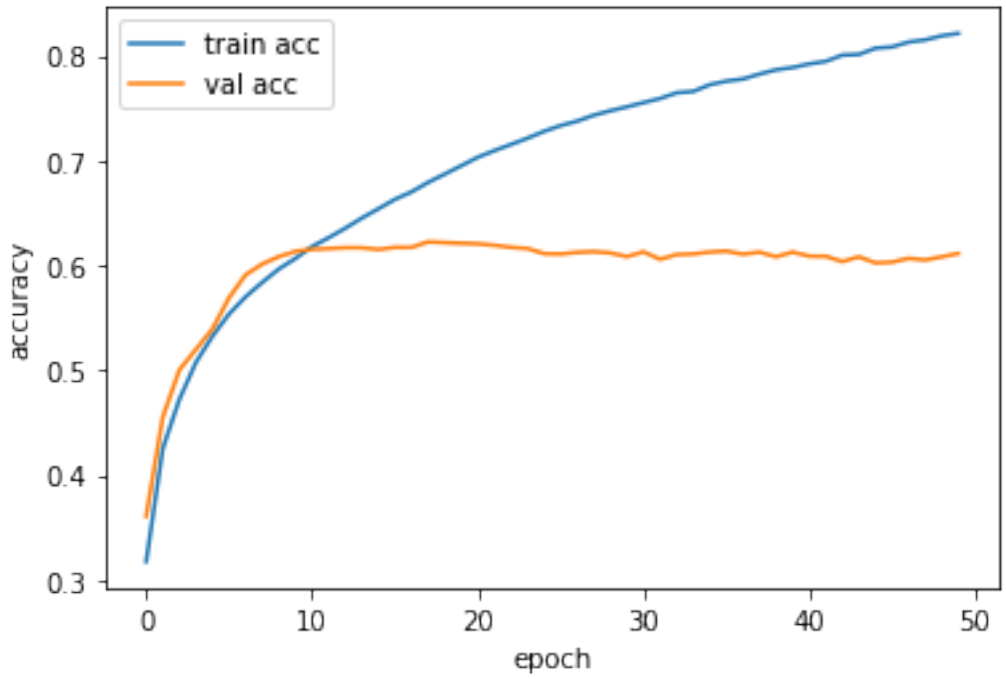




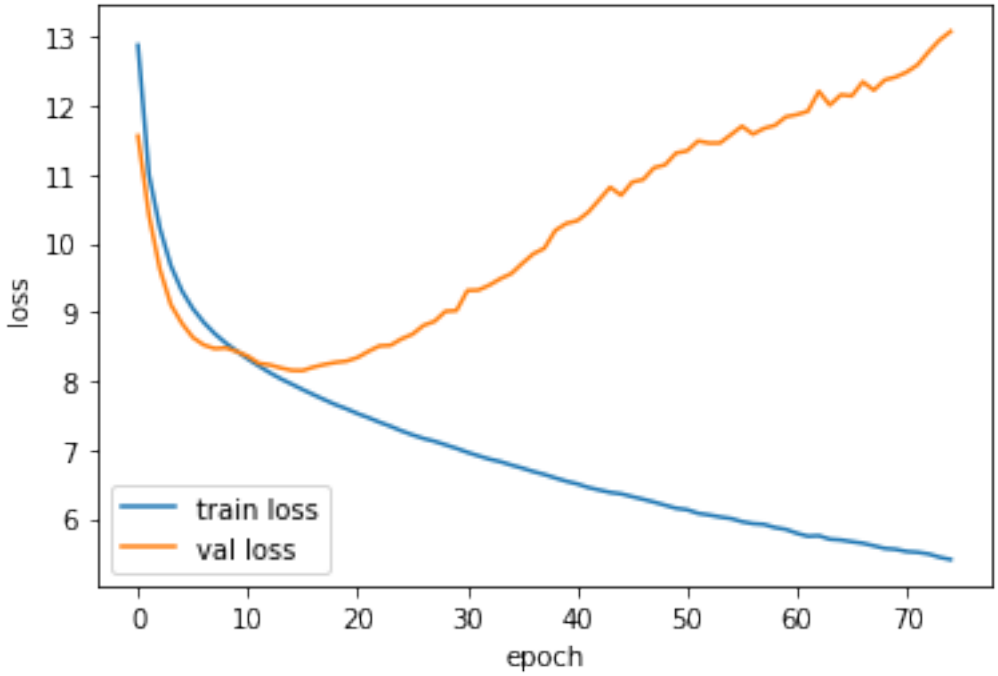
[6]:
```
#CPC on PTB, PTBxl (correct split), small model (slightly bigger), pca 2␣
↪channels
```

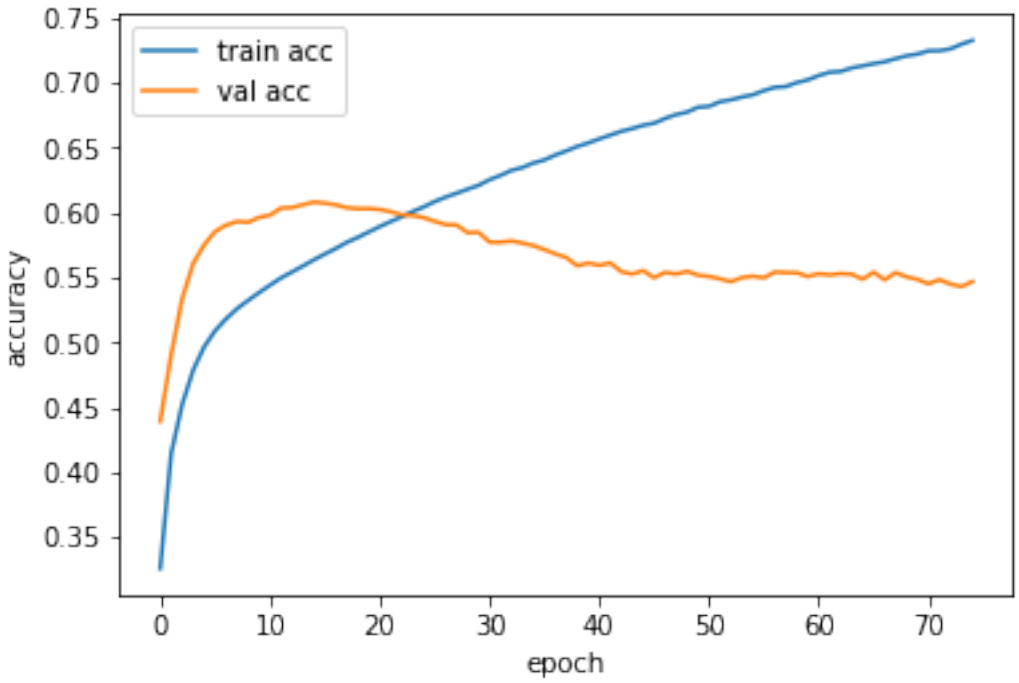
```
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/08_12_20-19')
plot(train_losses, val_losses, train_acc, val_acc)
```



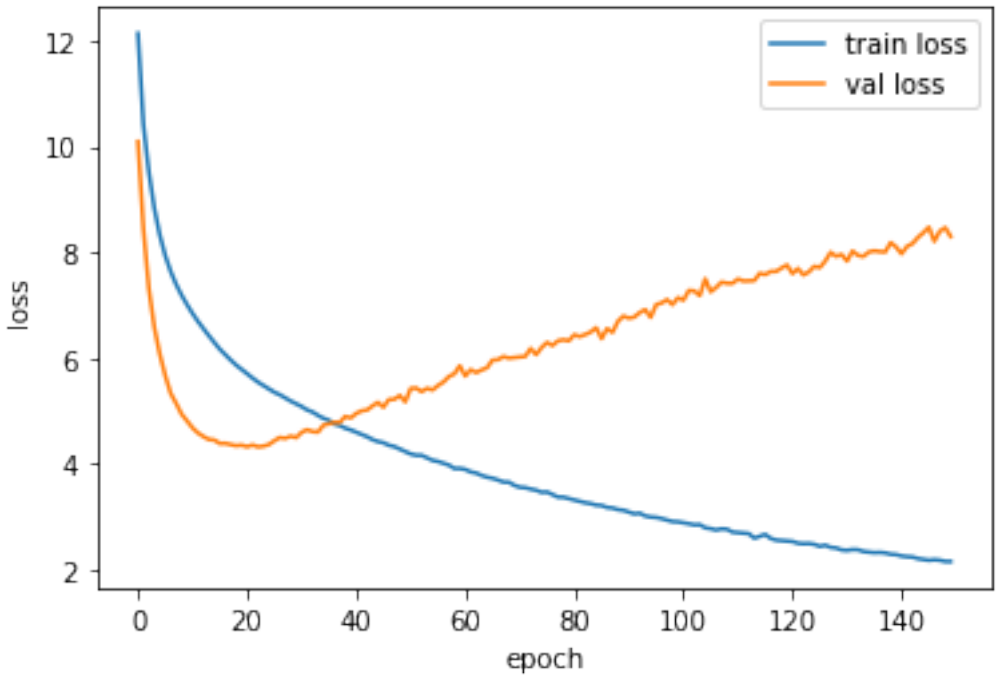


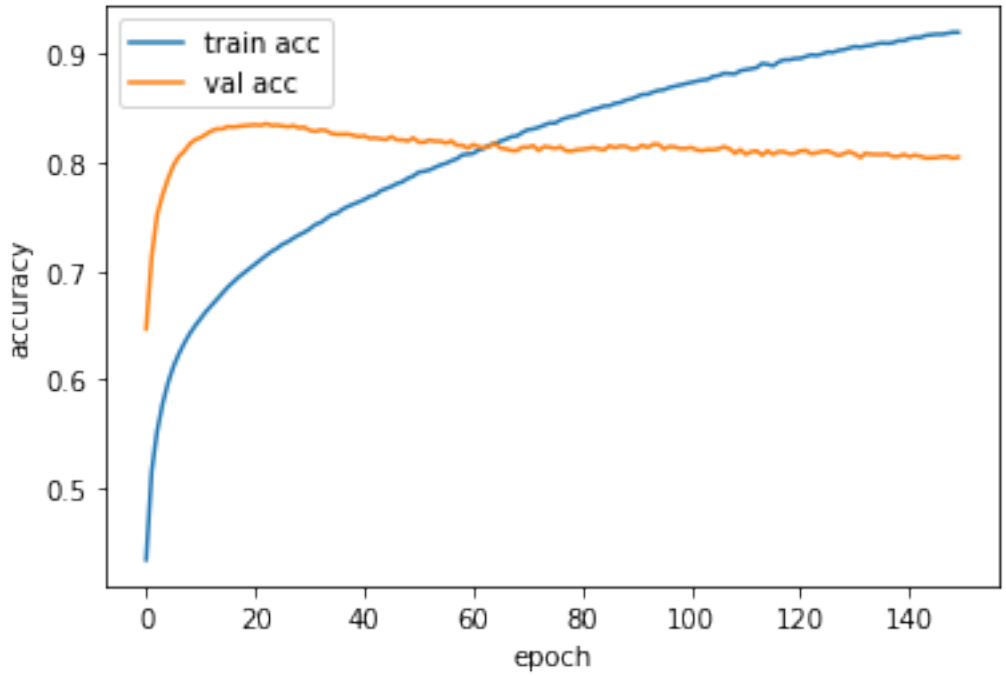
[4]:
```
#CPC on PTB, PTBxl (correct split), small model, pca 2 channels
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/08_12_20-16')
plot(train_losses, val_losses, train_acc, val_acc)
```



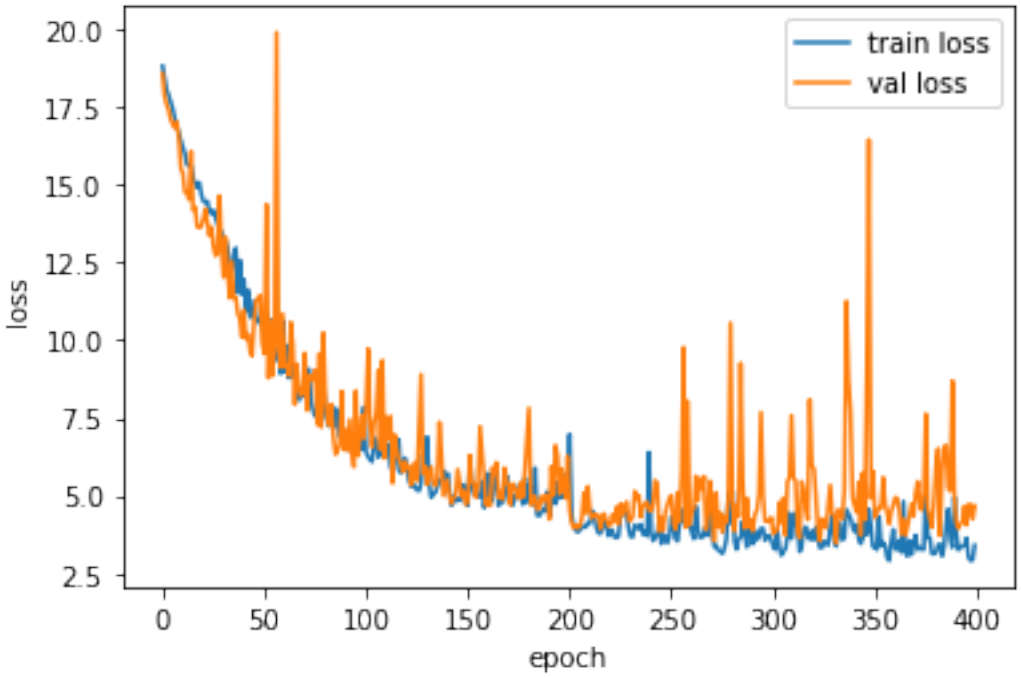


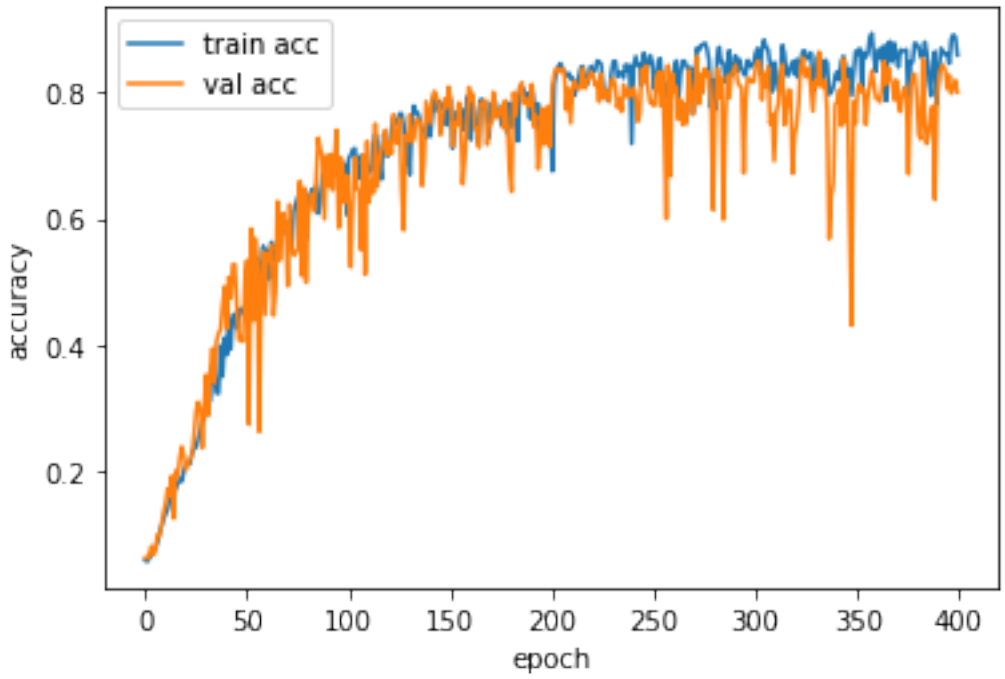
[13]:
```
#CPC on PTB, PTBxl (correct split), small model, no pca
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/07_12_20-18')
plot(train_losses, val_losses, train_acc, val_acc)
```





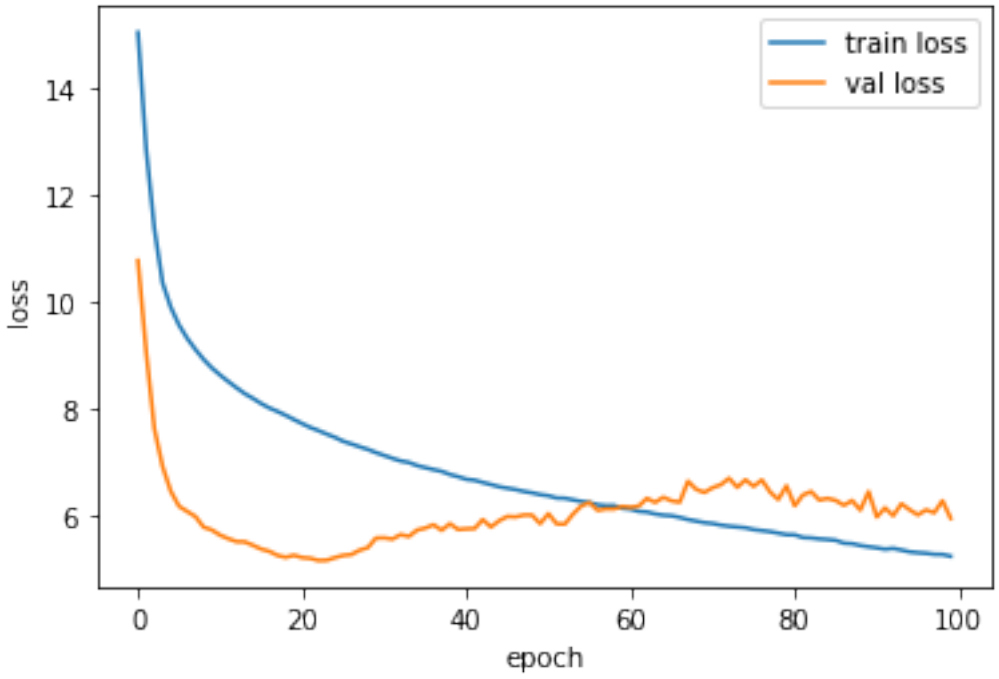
[9]:
```
#CPC on sinus data, small model, no pca
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/07_12_20-17')
plot(train_losses, val_losses, train_acc, val_acc) #Sinus data too similar for␣
↪contrastive loss
```

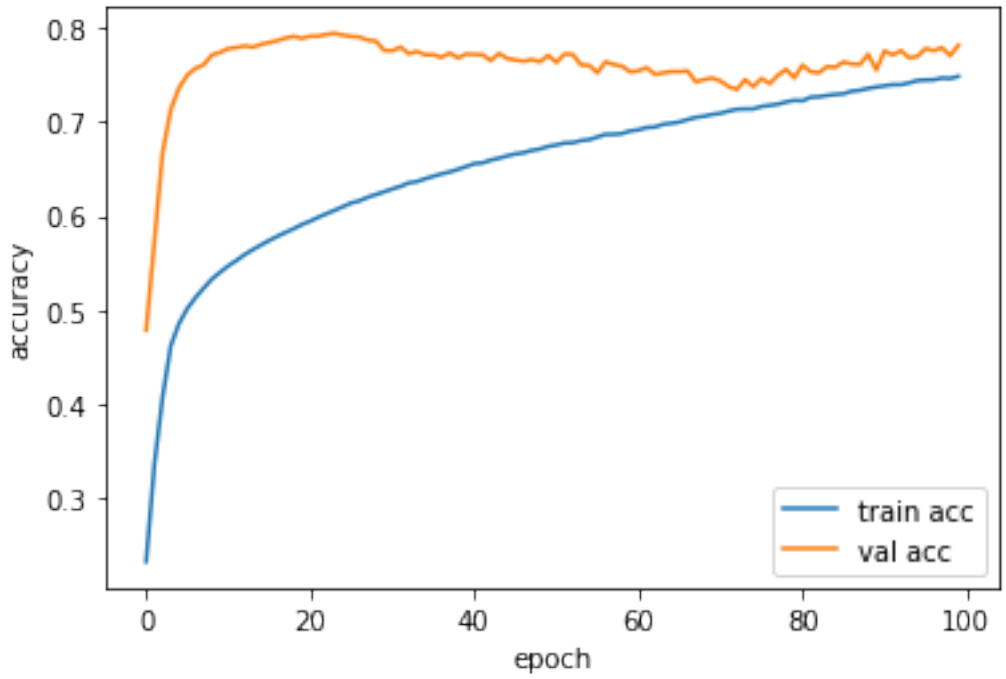




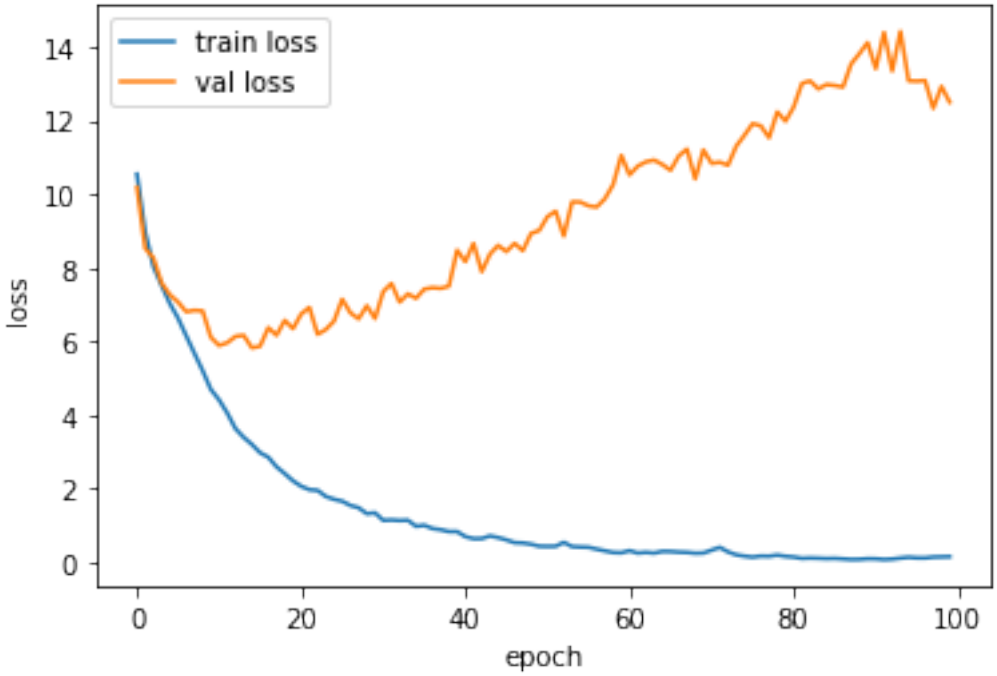
[11]:
```
#CPC with PCA and 2 components (PTB + PTBXL), also small model
```

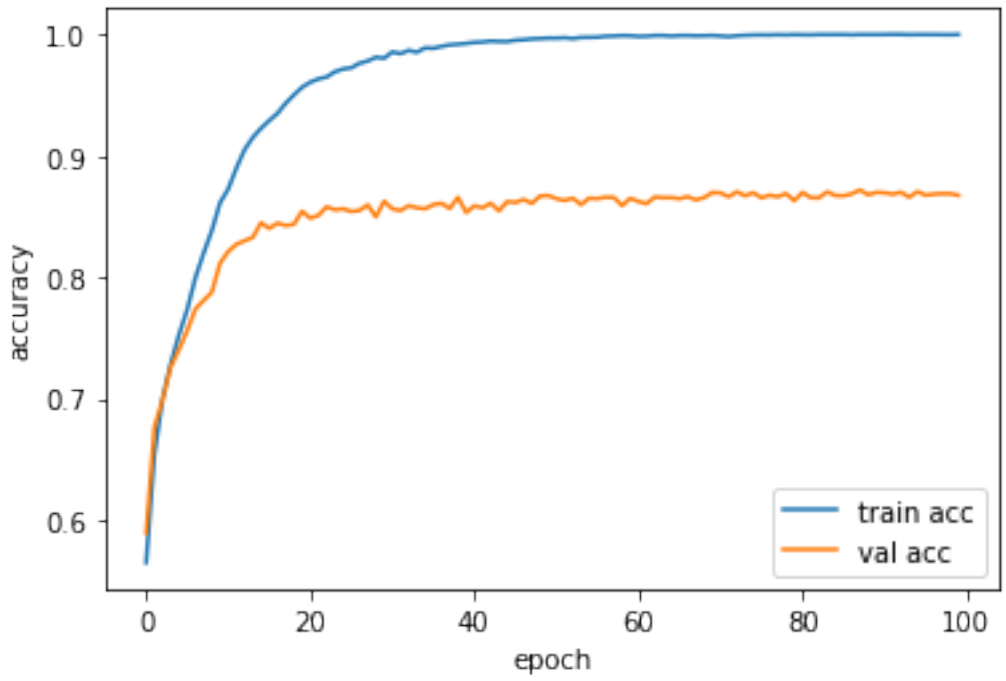
```
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/30_11_20-16')
plot(train_losses, val_losses, train_acc, val_acc)
```



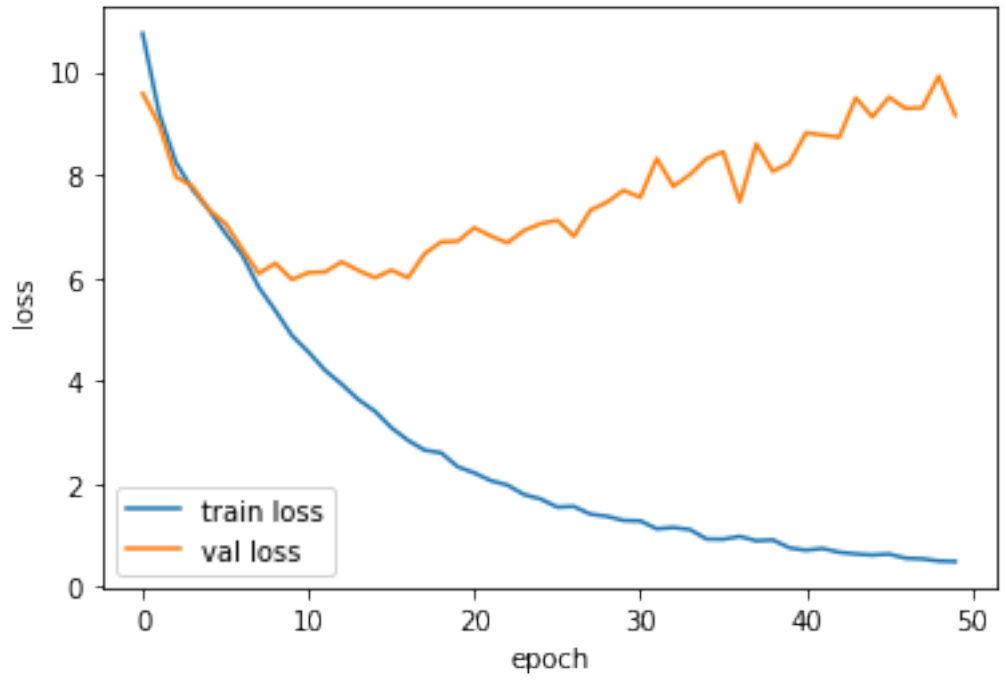


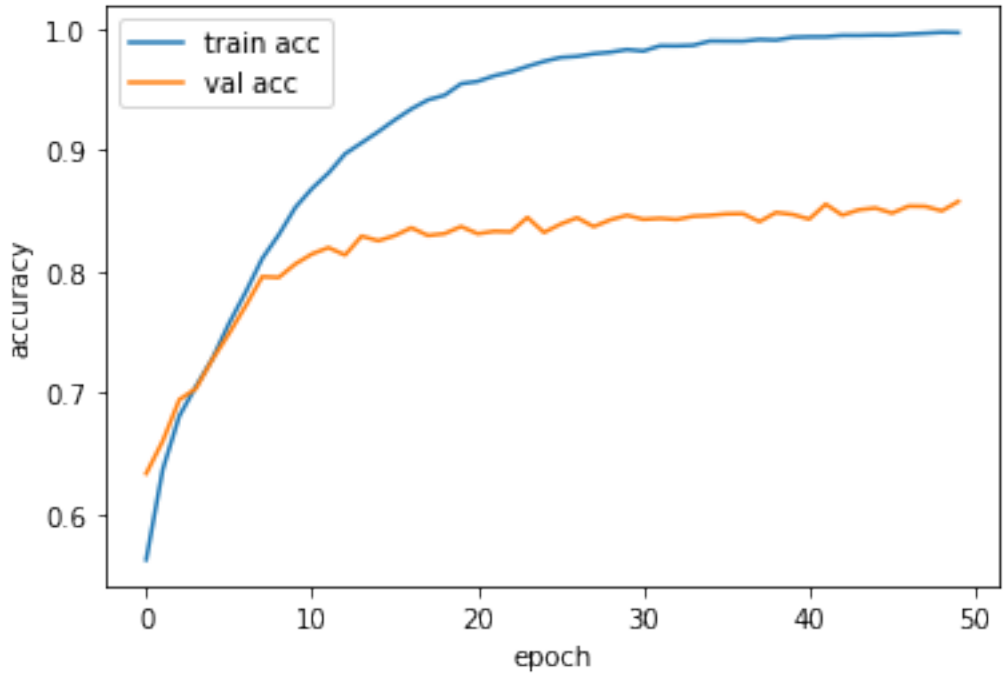
[8]:
```
#CPC with PCA (PTB + PTBXL) and 2 components
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/30_11_20-14')
plot(train_losses, val_losses, train_acc, val_acc)
```



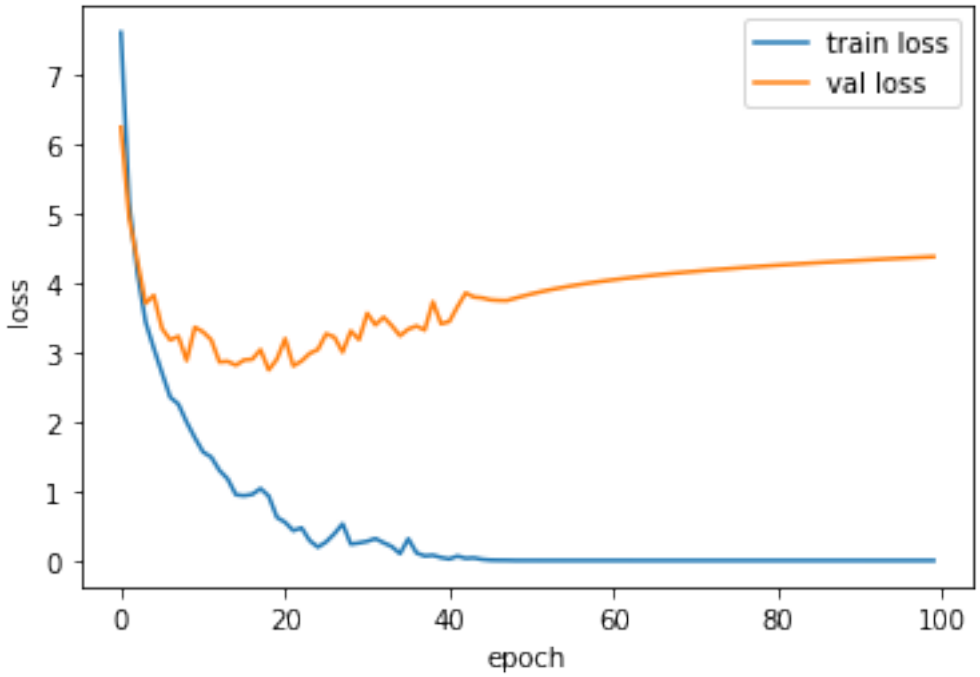


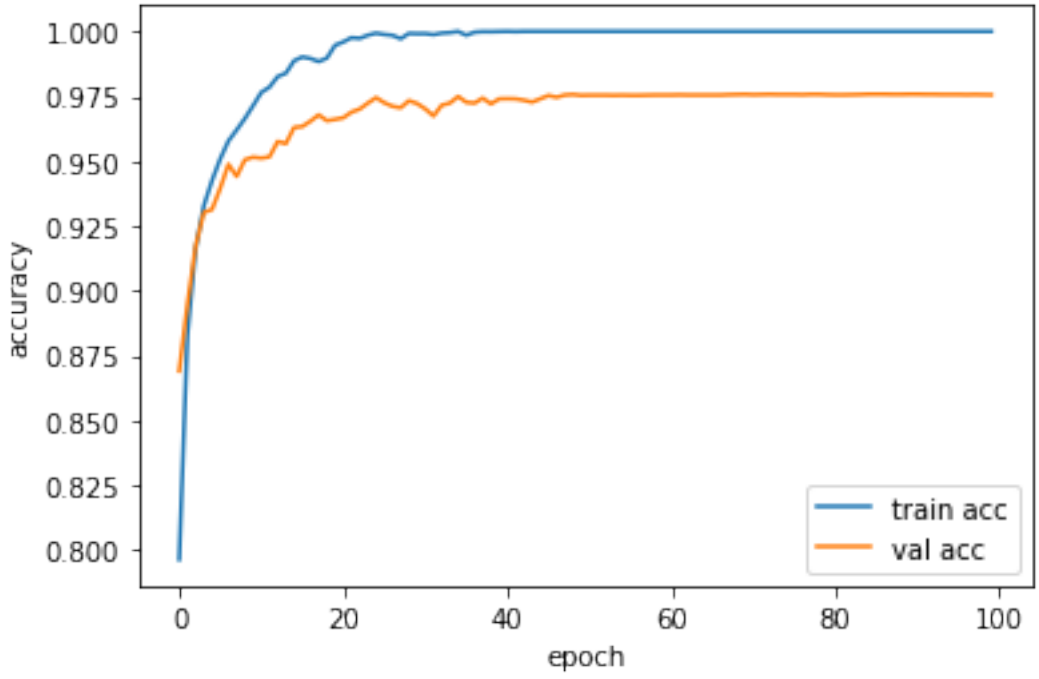
[4]:
```
#CPC with PCA (only PTB data) and 2 components
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/30_11_20-13')
plot(train_losses, val_losses, train_acc, val_acc)
```





[27]:
```
#CPC Got rid of batchnorm
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
 ↪Downloads/Github/contrastive-predictive-coding/models/27_11_20-14')
plot(train_losses, val_losses, train_acc, val_acc)
```





[21]:
```
train_losses, val_losses, train_acc, val_acc = load_pickles('/home/julian/
↪Downloads/Github/contrastive-predictive-coding/models/27_11_20-10')
```

```
plot(train_losses, val_losses, train_acc, val_acc)
```