

Gruppe 4
Aufgabe 11

Modell	# DNN-Parameter (Epoch_lr_Train_Evaluation)	DNN-Accuracy(Test)	Test-WER
Baseline	<u>13 0.001 0.7004 0.6619</u> "lr": 0.001, "batch_size": 1, "epochs": 50, "window_size": 25e-3, "hop_size": 10e-3, "feature_type": "MFCC_D_DD", "n_filters": 40, "fbank_fmin": 0, "fbank_fmax": 8000, "num_ceps": 13, "left_context": 10, "right_context": 10,	0.6485967298113164	2.924137931034483
Best	<u>9 0.000001 0.8392 0.7920</u>	0.79522086562379	0.896551724137931

```

uebung10 x
C:\Users\yfdon\anaconda3\envs\ASE39\python.exe C:/Project/ase-gruppe-4/uebung10.py --sourcedatadir ./dataset/ --savedir ./results/
Arguments:
sourcedatadir ./dataset/
savedir ./results/
Given posteriori OUT: ['seven', 'oh', 'one', 'seven', 'oh', 'four', 'nine']
OUT: ['seven', 'oh', 'one', 'seven', 'oh', 'four', 'nine']
-----
Total WER: 2.924137931034483
进程已结束,退出代码0

```

```

uebung10 x
C:\Users\yfdon\anaconda3\envs\ASE39\python.exe C:/Project/ase-gruppe-4/uebung10.py
Arguments:
sourcedatadir ./dataset/
savedir ./trained/
Given posteriori OUT: ['SEVEN', 'OH', 'ONE', 'SEVEN', 'OH', 'FOUR', 'NINE']
Ev|Te: 0%|          | 0/1 [00:00<?, ?it/s]C:\Users\yfdon\anaconda3\envs\ASE39\lib\site-packages\torch\nn\modules\rnn.
 chunk of memory. This means they need to be compacted at every call, possibly greatly increasing memory usage. To comp
C:\actions-runner\_work\pytorch\pytorch\builder\windows\pytorch\aten\src\ATen\native\cudnn\RNN.cpp:1225.)
result = _VF.lstm(input, hx, self._flat_weights, self.bias, self.num_layers,
Ev|Te: 0%|          | 0/1 [00:04<?, ?it/s]
Ev|Te: 0%|          | 0/2195 [00:00<?, ?it/s]OUT: ['SEVEN', 'OH', 'ONE', 'SEVEN', 'OH', 'FOUR', 'NINE']
Best model: 9_0.000001_0.8392_0.7920.pkl | Best train acc: 0.792
Ev|Te: 100%|██████████| 2195/2195 [00:38<00:00, 57.06it/s]
WER calculation:: 0%|          | 0/2195 [00:00<?, ?it/s]
DNN Test Acc: 0.79522086562379
WER calculation:: 100%|██████████| 2195/2195 [03:19<00:00, 10.99it/s]
-----
Total WER: 0.896551724137931

```

Best:

1. BLSTM Layer nach den FC-Layers
2. Learning Rate: ein StepLR scheduler wurde implementiert mit gamma=0.1 und step=4 bei 10 Epochen, also dass die learning rate bei 0.0001 startet und dann nach der 4. Epoche auf 0.00001 und nach der 8. auf 0.000001 sinkt.

```
x = self.fc1(x)
x = self.relu(x)
x = self.fc2(x)
x = self.relu(x)
x = self.fc3(x)
x = self.relu(x)
# print(x.shape)

x, _ = self.blstm(x)
# print(x.shape)

x = self.fc4(x)
# batch_size, sequence_length = x.shape
# x = x.view(batch_size, sequence_length, x.shape[-1])
```