Vergleich verschiedener Deep-Learning-Modelle zur Klassifizierung von Fledermausrufen

Christian Müller
Gutenbergstraße 20
64342 Seeheim-Jugenheim
chrmue44@gmail.com

1 Einleitung

Ziel der Untersuchung ist es, einen automatischen Klassifizierer für Fledermausrufe zu entwickeln. Die Klassifizierung soll eine möglichst gute Vorhersage-Genauigkeit bei möglichst geringem Rechenaufwand liefern. Als Software-Plattform dienen das AI-Framework tensorflow 2.0[i] und diverse weitere Python-Bibliotheken. Die Klassifizierung in dieser Untersuchung beschränkt sich auf 9 Fledermausarten.

Als Testdaten wurden Aufnahmen vom Naturkundemuseum Berlin[ii], von Rudolf Böhm[iii] und Christian Müller verwendet. Zum Vergleich verschiedener Deep-Learning Modell-Architekturen wurde immer der gleiche Testdatensatz verwendet.

Inhaltsverzeichnis

2 Verwendete Testdaten

Folgende Fledermausarten bzw. Klassen sollen detektiert werden:

Fledermausart	**Quelle**	**Anzahl Rufe**
Mopsfledermaus (Barbastella barbastellus)	Rudolf Böhm	748
Breitflügelfledermaus (Eptesicus serotinus	Tierstimmenarchiv.de	1069
Wasserfledermaus (Myotis daubentonii)	Tierstimmenarchiv.de	3308
Großes Mausohr (Myotis myotis)	Tierstimmenarchiv.de	999
Kleinabendsegler (Nyctalus leisleri)	Tierstimmenarchiv.de	779
Großer Abendsegler (Nyctalus noctula)	Tierstimmenarchiv.de	852
Rauhautfledermaus (Pipistrellus nathusii)	Tierstimmenarchiv.de	1678
Zwergfledermaus (Pipistrellus pipistrellus)	Tierstimmenarchiv.de	3579
Mückenfledermaus (Pipistrellus pygmaeus)	Rudolf, Böhm	4450
Heuschrecken (nachfolgend als Cric bezeichnet)	Christian Müller,	424
Sonstiges (incl. diverser Störgeräusche)	Christian Müller, www.github.org/ paddygoat[iv]	4329

Die Testdaten werden in 3 Klassen aufgeteilt:

Name	**Anzahl Rufe**	**Beschreibung**
train	17676	Daten, mit denen das Modell trainiert wird
Dev	2209	Daten, mit denen das Modell während des Trainings überprüft wird
test	2210	Daten, mit denen das Modell am Ende des Trainings validiert wird

3 Aufbereitung der Daten

Bevor die Tonaufnahmen dem Training bzw. der Klassifizierung zugeführt werden, erfolgt eine Aufbereitung der Daten. Dazu werden die folgenden Schritte durchgeführt:

1. Resampling der WAV-Aufnahmen zu einer Abtastrate von 312500 Hz

2. Erkennung und Isolation von einzelnen Rufen in der jeweiligen Aufnahme. Hierzu wird das biocoustics package[v] verwendet.

3. FFT: Die Aufnahme wird per FFT in den Frequenzbereich transformiert

4. Normierung: Die lauteste festgestellte Amplitude innerhalb eines Rufes wird auf den Wert 1.0 gesetzt. Sämtliche anderen Amplituden werden zu dieser Maximalamplitude ins Verhältnis gesetzt. Mit dieser Maßnahme werden die teilweise erheblichen Lautstärkeunterschiede zwischen den Aufnahmen ausgeglichen.

5. Die FFTs der einzelnen Rufe werden auf ein Format von 129 x 330 Punkten begrenzt. Kürzere Rufe werden mit 0 aufgefüllt, längere Rufe werden am Ende abgeschnitten.

6. Rauschunterdrückung: 2 verschiedene Methoden getestet:
- alle Amplituden im Spektrogramm unterhalb eines Schwellwertes auf 0 gesetzt
- nur die oberen 95% der Energie einer Spektrallinie werden berücksichtigt, der Rest auf 0 gesetzt (Methode 1 funktioniert besser)

4 Ergebnisse

In der unten stehenden Tabelle sind die Ergebnisse für die einzelnen Arten aufgelistet. Berechnet ist jeweils die „precision“[vi] für jede erkannte Klasse bzw. Fledermausart. Generell schneidet das resNet34 am besten ab. Für einzelne Arten liefern andere Modelle jedoch teilweise bessere Ergebnisse.

Das Training der Modelle dauert bei allen Modellen außer dem resNet34 60 … 90 Sekunden pro Epoche. Lediglich das Resnet34 braucht ca. 1,5 Stunden zum Training einer Epoche, ist also wesentlich ressourcenhungriger.

Modell Art	Precision (TP / (TP + FP)				
	Resnet34	**rnn5**	**rnn6a**	**flat**	**rnn1a**
----	99.1%	98.8%	99.1%	98.6%	99.1%
Bbar	96.4%	95.2%	91.7%	88.1%	94.0%
Cric	97.5%	97.3%	97.3%	94.7%	100.0%
Eser	80.4%	81.4%	81.4%	76.2%	81.4%
Mdau	98.4%	96.6%	96.9%	91.1%	92.7%
Mmyo	93.5%	80.4%	76.0%	62.0%	64.1%
Nlei	66.7%	74.7%	76%	56.0%	65.3%
Nnoc	78.4%	79.5%	76.1%	76.1%	84.1%
Pnat	96.3%	92.1%	93.3%	91.4%	92.1%
Ppip	98.5%	96.2%	95.0%	94.4%	96.8%
Ppyg	99.8%	99.1%	98.6%	99.5%	99.8%

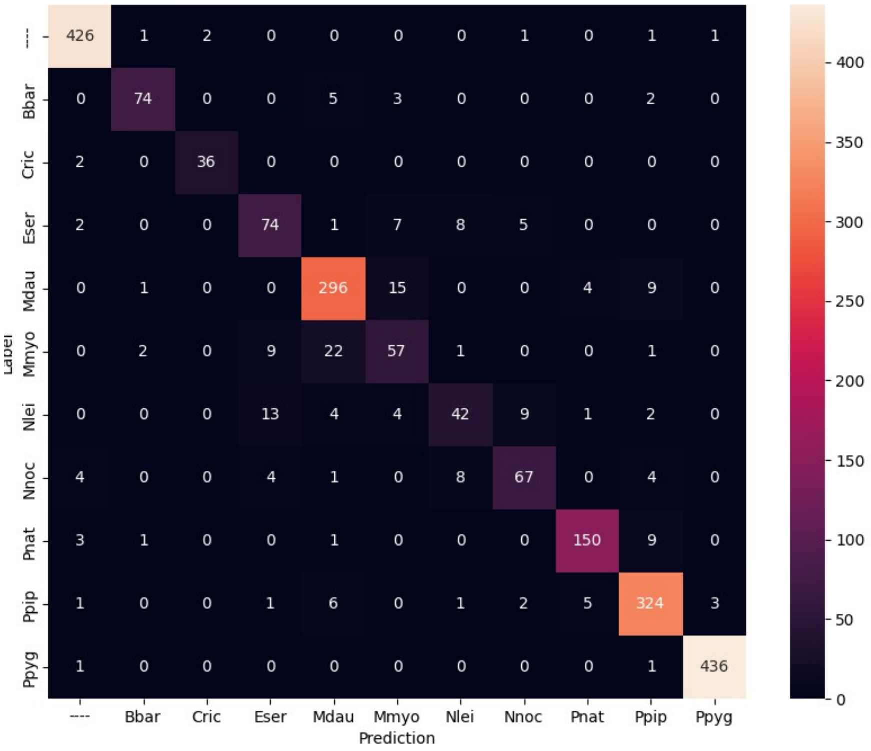
Da die Modelle zwischen 2 und über 20 Millionen Parameter haben, erscheinen die gut 20.000 Trainingsbeispiele etwas knapp bemessen. Hier würden mehr verlässliche Daten sicher noch zu einer Verbesserung des Ergebnisses beitragen.

5 Implementierung der untersuchten Modelle

5.1 FlatModel

Learning rate: 0.00002, accurracy: 0.91

```
Model: "flatModel"
_________________________________________________________________
 Layer (type)              Output Shape              Param #
=================================================================
 flatten (Flatten)         (None, 42570)             0
 dense (Dense)              (None, 256)              10898176
 dense_1 (Dense)             (None, 256)              65792
 dense_2 (Dense)             (None, 128)              32896
 dense_3 (Dense)             (None, 11)              1419
=================================================================
Total params: 10,998,283
Trainable params: 10,998,283
Non-trainable params: 0
_________________________________________________________________
```



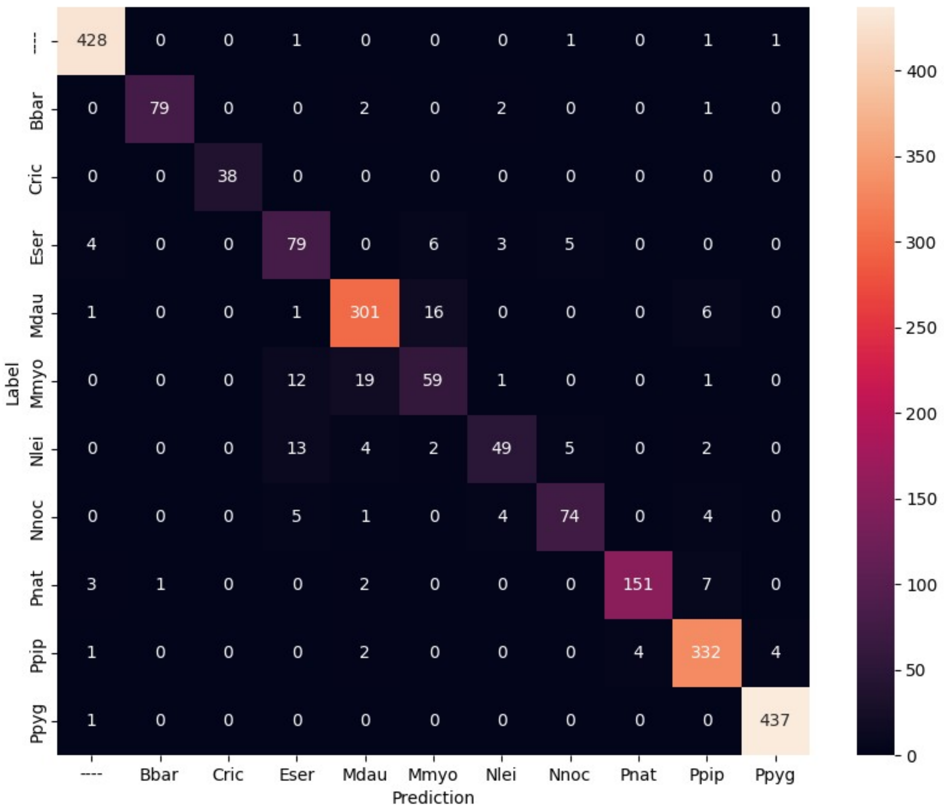
5.2 Rnn1aModel

Learning rate: 0.00002, Optimizer „Adam“, 95 Epochs, dropout 0,5

accuracy 0.93

Model: "rnn1aModel"

```
_________________________________________________________________
 Layer (type)             Output Shape             Param #
=================================================================
 gru (GRU)                 (None, 129, 128)          176640
 flatten (Flatten)         (None, 16512)            0
 dropout (Dropout)              (None, 16512)           0
 dense (Dense)               (None, 128)             2113664
 dense_1 (Dense)               (None, 11)               1419
=================================================================
Total params: 2,291,723
Trainable params: 2,291,723
Non-trainable params: 0
```



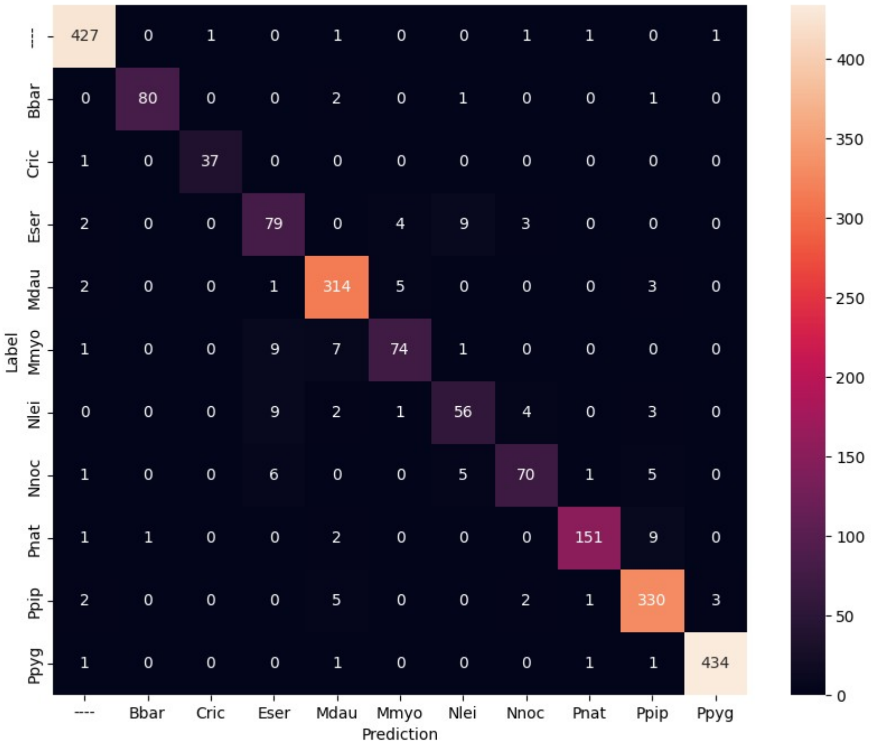
	----	Bbar	Cric	Eser	Mdau	Mmyo	Nlei	Nnoc	Pnat	Ppip	Ppyg
----	428	0	0	1	0	0	0	1	0	1	1
Bbar	0	79	0	0	2	0	2	0	0	1	0
Cric	0	0	38	0	0	0	0	0	0	0	0
Eser	4	0	0	79	0	6	3	5	0	0	0
Mdau	1	0	0	1	301	16	0	0	0	6	0
Mmyo	0	0	0	12	19	59	1	0	0	1	0
Nlei	0	0	0	13	4	2	49	5	0	2	0
Nnoc	0	0	0	5	1	0	4	74	0	4	0
Pnat	3	1	0	0	2	0	0	0	151	7	0
Ppip	1	0	0	0	2	0	0	0	4	332	4
Ppyg	1	0	0	0	0	0	0	0	0	0	437

5.3 Rnn5Model

Learning rate: 0.00002, Optimizer „Adam“, 41 Epochs, dropout 0,5

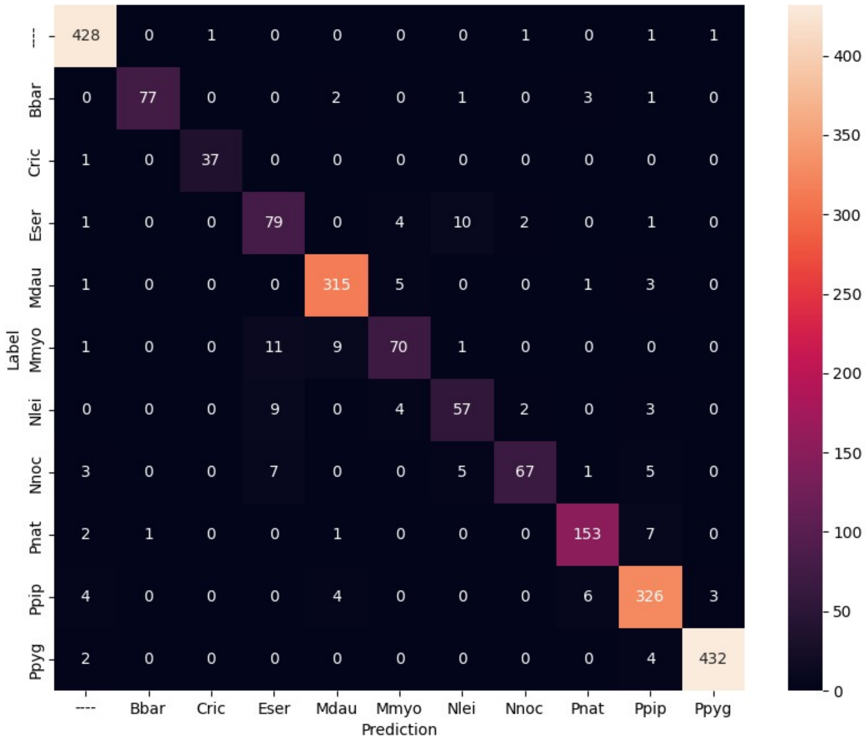
accurracy 0.943

```
Model: "rnn5Model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d (Conv1D)             (None, 29, 196)           970396
 batch_normalization (BatchNormalization)  (None, 29, 196)       784
 activation (Activation)     (None, 29, 196)           0
 dropout (Dropout)           (None, 29, 196)           0
 gru (GRU)                   (None, 29, 128)           125184
 dropout_1 (Dropout)         (None, 29, 128)           0
 batch_normalization_1 (BatchNormalization)  (None, 29, 128)       512
  gru_1 (GRU)                (None, 29, 128)           99072
 dropout_2 (Dropout)         (None, 29, 128)           0
 flatten (Flatten)           (None, 3712)              0
 dense (Dense)               (None, 128)               475264
 dense_1 (Dense)             (None, 11)                1419
=================================================================
Total params: 1,672,631
Trainable params: 1,671,983
Non-trainable params: 648
```



5.4 rnn6aModel

Learning rate: 0.00002, Epochs: 45, dropout 0.5, accuracy 93.8%



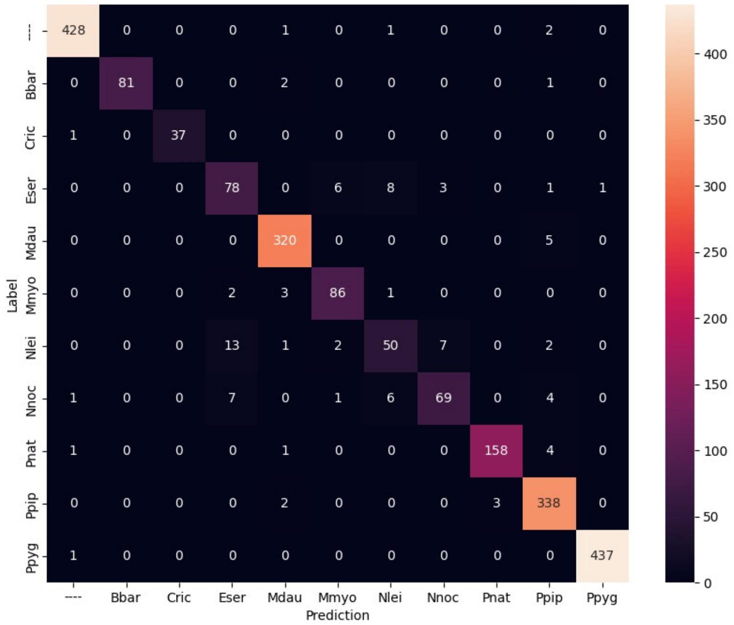
Label \ Prediction	----	Bbar	Cric	Eser	Mdau	Mmyo	Nlei	Nnoc	Pnat	Ppip	Ppyg
----	428	0	1	0	0	0	0	1	0	1	1
Bbar	0	77	0	0	2	0	1	0	3	1	0
Cric	1	0	37	0	0	0	0	0	0	0	0
Eser	1	0	0	79	0	4	10	2	0	1	0
Mdau	1	0	0	0	315	5	0	0	1	3	0
Mmyo	1	0	0	11	9	70	1	0	0	0	0
Nlei	0	0	0	9	0	4	57	2	0	3	0
Nnoc	3	0	0	7	0	0	5	67	1	5	0
Pnat	2	1	0	0	1	0	0	0	153	7	0
Ppip	4	0	0	0	4	0	0	0	6	326	3
Ppyg	2	0	0	0	0	0	0	0	0	4	432

```
Model: "rnn6aModel"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 conv1d (Conv1D)              (None, 29, 196)           970396
 batch_normalization (BatchNormalization)  (None, 29, 196)        784
 activation (Activation)      (None, 29, 196)           0
 dropout (Dropout)            (None, 29, 196)           0
 gru (GRU)                    (None, 29, 128)           125184
 dropout_1 (Dropout)          (None, 29, 128)           0
 batch_normalization_1 (BatchNormalization)  (None, 29, 128)        512
  gru_1 (GRU)                 (None, 29, 128)           99072
 dropout_2 (Dropout)          (None, 29, 128)           0
 batch_normalization_2 (BatchNormalization)  (None, 29, 128)        512
 gru_2 (GRU)                  (None, 29, 128)           99072
 dropout_3 (Dropout)          (None, 29, 128)           0
 flatten (Flatten)            (None, 3712)              0
 dense (Dense)                (None, 128)               475264
 dense_1 (Dense)              (None, 11)                1419
=================================================================
Total params: 1,772,215
Trainable params: 1,771,311
Non-trainable params: 904
```

5.5 ResNet34

rows, timeSteps, classes: 129 330 11, 5 epochs, learning rate 0.00002

accurracy 0.956



Model: "resNet34Model"

```
 Layer (type)                Output Shape         Param #     Connected to
=====================================================================================================
 input_1 (InputLayer)         [(None, 129, 330)]   0           []
 reshape (Reshape)            (None, 129, 330, 1)  0           ['input_1[0][0]']
 zero_padding2d (ZeroPadding2D)  (None, 135, 336, 1)  0          ['reshape[0][0]']
 conv2d (Conv2D)              (None, 68, 168, 64)  3200        ['zero_padding2d[0][0]']
 batch_normalization (BatchNorm alization) (None, 68, 168, 64)  256        ['conv2d[0][0]']
activation (Activation)       (None, 68, 168, 64)  0           ['batch_normalization[0][0]']
 max_pooling2d (MaxPooling2D)   (None, 34, 84, 64)  0           ['activation[0][0]']
 conv2d_1 (Conv2D)              (None, 34, 84, 64)  36928      ['max_pooling2d[0][0]']
 batch_normalization_1 (BatchNo rmalization)  (None, 34, 84, 64)  256        ['conv2d_1[0][0]']
 activation_1 (Activation)      (None, 34, 84, 64)  0           ['batch_normalization_1[0][0]']
 conv2d_2 (Conv2D)              (None, 34, 84, 64)  36928      ['activation_1[0][0]']
 batch_normalization_2 (BatchNormalization)  (None, 34, 84, 64)  256        ['conv2d_2[0][0]']
 add (Add)                    (None, 34, 84, 64)   0           ['batch_normalization_2[0][0]',
                                                                'max_pooling2d[0][0]']
 activation_2 (Activation)      (None, 34, 84, 64)  0           ['add[0][0]']
 conv2d_3 (Conv2D)              (None, 34, 84, 64)  36928      ['activation_2[0][0]']
 batch_normalization_3 (BatchNormalization)  (None, 34, 84, 64)  256        ['conv2d_3[0][0]']
 activation_3 (Activation)      (None, 34, 84, 64)  0           ['batch_normalization_3[0][0]']
 conv2d_4 (Conv2D)              (None, 34, 84, 64)  36928      ['activation_3[0][0]']
 batch_normalization_4 (BatchNormalization)  (None, 34, 84, 64)  256        ['conv2d_4[0][0]']
 add_1 (Add)                  (None, 34, 84, 64)   0           ['batch_normalization_4[0][0]',
                                                                'activation_2[0][0]']
 activation_4 (Activation)      (None, 34, 84, 64)  0           ['add_1[0][0]']
 conv2d_5 (Conv2D)              (None, 34, 84, 64)  36928      ['activation_4[0][0]']
 batch_normalization_5 (BatchNormalization)  (None, 34, 84, 64)  256        ['conv2d_5[0][0]']
 activation_5 (Activation)      (None, 34, 84, 64)  0           ['batch_normalization_5[0][0]']
 conv2d_6 (Conv2D)              (None, 34, 84, 64)  36928      ['activation_5[0][0]']
 batch_normalization_6 (BatchNo rmalization)  (None, 34, 84, 64)  256        ['conv2d_6[0][0]']
```

```
add_2 (Add)                  (None, 34, 84, 64)   0          ['batch_normalization_6[0][0]',
                                                              'activation_4[0][0]']
activation_6 (Activation)      (None, 34, 84, 64)   0          ['add_2[0][0]']
conv2d_7 (Conv2D)              (None, 17, 42, 128)  73856      ['activation_6[0][0]']
batch_normalization_7 (BatchNormalization)  (None, 17, 42, 128)  512        ['conv2d_7[0][0]']
activation_7 (Activation)      (None, 17, 42, 128)  0          ['batch_normalization_7[0][0]']
conv2d_8 (Conv2D)              (None, 17, 42, 128)  147584     ['activation_7[0][0]']
batch_normalization_8 (BatchNormalization)  (None, 17, 42, 128)  512        ['conv2d_8[0][0]']
conv2d_9 (Conv2D)              (None, 17, 42, 128)  8320       ['activation_6[0][0]']
add_3 (Add)                  (None, 17, 42, 128)  0          ['batch_normalization_8[0][0]',
                                                              'conv2d_9[0][0]']
activation_8 (Activation)      (None, 17, 42, 128)  0          ['add_3[0][0]']
conv2d_10 (Conv2D)             (None, 17, 42, 128)  147584     ['activation_8[0][0]']
batch_normalization_9 (BatchNormalization)  (None, 17, 42, 128)  512        ['conv2d_10[0][0]']
activation_9 (Activation)      (None, 17, 42, 128)  0          ['batch_normalization_9[0][0]']
conv2d_11 (Conv2D)             (None, 17, 42, 128)  147584     ['activation_9[0][0]']
batch_normalization_10 (BatchNormalization)  (None, 17, 42, 128)  512        ['conv2d_11[0][0]']
add_4 (Add)                  (None, 17, 42, 128)  0          ['batch_normalization_10[0][0]',
                                                              'activation_8[0][0]']
activation_10 (Activation)     (None, 17, 42, 128)  0          ['add_4[0][0]']
conv2d_12 (Conv2D)             (None, 17, 42, 128)  147584     ['activation_10[0][0]']
batch_normalization_11 (BatchN  (None, 17, 42, 128)  512        ['conv2d_12[0][0]']
ormalization)
activation_11 (Activation)     (None, 17, 42, 128)  0          ['batch_normalization_11[0][0]']
conv2d_13 (Conv2D)             (None, 17, 42, 128)  147584     ['activation_11[0][0]']
batch_normalization_12 (BatchNormalization)  (None, 17, 42, 128)  512        ['conv2d_13[0][0]']
add_5 (Add)                  (None, 17, 42, 128)  0          ['batch_normalization_12[0][0]',
                                                              'activation_10[0][0]']
activation_12 (Activation)     (None, 17, 42, 128)  0          ['add_5[0][0]']
conv2d_14 (Conv2D)             (None, 17, 42, 128)  147584     ['activation_12[0][0]']
batch_normalization_13 (BatchN ormalization)  (None, 17, 42, 128)  512        ['conv2d_14[0][0]']
activation_13 (Activation)     (None, 17, 42, 128)  0          ['batch_normalization_13[0][0]']
conv2d_15 (Conv2D)             (None, 17, 42, 128)  147584     ['activation_13[0][0]']
batch_normalization_14 (BatchNormalization)  (None, 17, 42, 128)  512        ['conv2d_15[0][0]']
add_6 (Add)                  (None, 17, 42, 128)  0          ['batch_normalization_14[0][0]',
                                                              'activation_12[0][0]']
activation_14 (Activation)     (None, 17, 42, 128)  0          ['add_6[0][0]']
conv2d_16 (Conv2D)             (None, 9, 21, 256)   295168     ['activation_14[0][0]']
batch_normalization_15 (BatchN ormalization)  (None, 9, 21, 256)  1024       ['conv2d_16[0][0]']
activation_15 (Activation)     (None, 9, 21, 256)   0          ['batch_normalization_15[0][0]']
conv2d_17 (Conv2D)             (None, 9, 21, 256)   590080     ['activation_15[0][0]']
batch_normalization_16 (BatchNormalization)  (None, 9, 21, 256)  1024       ['conv2d_17[0][0]']
conv2d_18 (Conv2D)             (None, 9, 21, 256)   33024      ['activation_14[0][0]']
add_7 (Add)                  (None, 9, 21, 256)   0          ['batch_normalization_16[0][0]',
                                                              'conv2d_18[0][0]']
activation_16 (Activation)     (None, 9, 21, 256)   0          ['add_7[0][0]']
conv2d_19 (Conv2D)             (None, 9, 21, 256)   590080     ['activation_16[0][0]']
batch_normalization_17 (BatchNormalization)  (None, 9, 21, 256)  1024       ['conv2d_19[0][0]']
activation_17 (Activation)     (None, 9, 21, 256)   0          ['batch_normalization_17[0][0]']
conv2d_20 (Conv2D)             (None, 9, 21, 256)   590080     ['activation_17[0][0]']
batch_normalization_18 (BatchNormalization)  (None, 9, 21, 256)  1024       ['conv2d_20[0][0]']
add_8 (Add)                  (None, 9, 21, 256)   0          ['batch_normalization_18[0][0]',
                                                              'activation_16[0][0]']
activation_18 (Activation)     (None, 9, 21, 256)   0          ['add_8[0][0]']
conv2d_21 (Conv2D)             (None, 9, 21, 256)   590080     ['activation_18[0][0]']
batch_normalization_19 (BatchNormalization)  (None, 9, 21, 256)  1024       ['conv2d_21[0][0]']
activation_19 (Activation)     (None, 9, 21, 256)   0          ['batch_normalization_19[0][0]']
conv2d_22 (Conv2D)             (None, 9, 21, 256)   590080     ['activation_19[0][0]']
batch_normalization_20 (BatchNormalization)  (None, 9, 21, 256)  1024       ['conv2d_22[0][0]']
add_9 (Add)                  (None, 9, 21, 256)   0          ['batch_normalization_20[0][0]',
                                                              'activation_18[0][0]']
activation_20 (Activation)     (None, 9, 21, 256)   0          ['add_9[0][0]']
conv2d_23 (Conv2D)             (None, 9, 21, 256)   590080     ['activation_20[0][0]']
batch_normalization_21 (BatchNormalization)  (None, 9, 21, 256)  1024       ['conv2d_23[0][0]']
activation_21 (Activation)     (None, 9, 21, 256)   0          ['batch_normalization_21[0][0]']
conv2d_24 (Conv2D)             (None, 9, 21, 256)   590080     ['activation_21[0][0]']
batch_normalization_22 (BatchNormalization)  (None, 9, 21, 256)  1024       ['conv2d_24[0][0]']
add_10 (Add)                  (None, 9, 21, 256)   0          ['batch_normalization_22[0][0]',
                                                   'activation_20[0][0]']
 activation_22 (Activation)     (None, 9, 21, 256)   0          ['add_10[0][0]']
 conv2d_25 (Conv2D)             (None, 9, 21, 256)   590080      ['activation_22[0][0]']
 batch_normalization_23 (BatchNormalization)  (None, 9, 21, 256)  1024        ['conv2d_25[0][0]']
 activation_23 (Activation)     (None, 9, 21, 256)   0          ['batch_normalization_23[0][0]']
 conv2d_26 (Conv2D)             (None, 9, 21, 256)   590080      ['activation_23[0][0]']
 batch_normalization_24 (BatchNormalization)  (None, 9, 21, 256)  1024        ['conv2d_26[0][0]']
 add_11 (Add)                   (None, 9, 21, 256)   0          ['batch_normalization_24[0][0]',
                                                   'activation_22[0][0]']
 activation_24 (Activation)     (None, 9, 21, 256)   0          ['add_11[0][0]']
 conv2d_27 (Conv2D)             (None, 9, 21, 256)   590080      ['activation_24[0][0]']
 batch_normalization_25 (BatchNormalization)  (None, 9, 21, 256)  1024        ['conv2d_27[0][0]']
 activation_25 (Activation)     (None, 9, 21, 256)   0          ['batch_normalization_25[0][0]']
 conv2d_28 (Conv2D)             (None, 9, 21, 256)   590080      ['activation_25[0][0]']
 batch_normalization_26 (BatchNormalization)  (None, 9, 21, 256)  1024        ['conv2d_28[0][0]']
 add_12 (Add)                   (None, 9, 21, 256)   0          ['batch_normalization_26[0][0]',
                                                   'activation_24[0][0]']
 activation_26 (Activation)     (None, 9, 21, 256)   0          ['add_12[0][0]']
 conv2d_29 (Conv2D)             (None, 5, 11, 512)   1180160     ['activation_26[0][0]']
 batch_normalization_27 (BatchNormalization)  (None, 5, 11, 512)  2048        ['conv2d_29[0][0]']
  activation_27 (Activation)     (None, 5, 11, 512)   0          ['batch_normalization_27[0][0]']

 conv2d_30 (Conv2D)             (None, 5, 11, 512)   2359808     ['activation_27[0][0]']
 batch_normalization_28 (BatchNormalization)  (None, 5, 11, 512)  2048        ['conv2d_30[0][0]']
 conv2d_31 (Conv2D)             (None, 5, 11, 512)   131584      ['activation_26[0][0]']
 add_13 (Add)                   (None, 5, 11, 512)   0          ['batch_normalization_28[0][0]',
                                                   'conv2d_31[0][0]']
 activation_28 (Activation)     (None, 5, 11, 512)   0          ['add_13[0][0]']
 conv2d_32 (Conv2D)             (None, 5, 11, 512)   2359808     ['activation_28[0][0]']
 batch_normalization_29 (BatchNormalization)  (None, 5, 11, 512)  2048        ['conv2d_32[0][0]']
 activation_29 (Activation)     (None, 5, 11, 512)   0          ['batch_normalization_29[0][0]']
 conv2d_33 (Conv2D)             (None, 5, 11, 512)   2359808     ['activation_29[0][0]']
 batch_normalization_30 (BatchNormalization)  (None, 5, 11, 512)  2048        ['conv2d_33[0][0]']
 add_14 (Add)                   (None, 5, 11, 512)   0          ['batch_normalization_30[0][0]',
                                                   'activation_28[0][0]']
 activation_30 (Activation)     (None, 5, 11, 512)   0          ['add_14[0][0]']
 conv2d_34 (Conv2D)             (None, 5, 11, 512)   2359808     ['activation_30[0][0]']
 batch_normalization_31 (BatchNormalization)  (None, 5, 11, 512)  2048        ['conv2d_34[0][0]']
 activation_31 (Activation)     (None, 5, 11, 512)   0          ['batch_normalization_31[0][0]']
 conv2d_35 (Conv2D)             (None, 5, 11, 512)   2359808     ['activation_31[0][0]']
 batch_normalization_32 (BatchNormalization)  (None, 5, 11, 512)  2048        ['conv2d_35[0][0]']
 add_15 (Add)                   (None, 5, 11, 512)   0          ['batch_normalization_32[0][0]',
                                                   'activation_30[0][0]']
 activation_32 (Activation)     (None, 5, 11, 512)   0          ['add_15[0][0]']
 average_pooling2d (AveragePooling2D)  (None, 3, 6, 512)   0          ['activation_32[0][0]']
 flatten (Flatten)              (None, 9216)         0          ['average_pooling2d[0][0]']
 dense (Dense)                  (None, 512)          4719104    ['flatten[0][0]']
 dense_1 (Dense)                (None, 11)           5643       ['dense[0][0]']
==================================================================================================
Total params: 26,025,099
Trainable params: 26,009,867
Non-trainable params: 15,232
__________________________________
```

6 Quellen

www.tierstimmenarchiv.de

https://cran.r-project.org/web/packages/bioacoustics/index.html

https://www.tensorflow.org/

https://www.coursera.org/learn/neural-networks-deep-learning/home/

https://www.analyticsvidhya.com/blog/2021/08/how-to-code-your-resnet-from-scratch-in-tensorflow/

Gulli Anonio, Kapoor Amita, Oal Sujit; Deep Leraning with Tensorflow 2.0 and Keras; 2019; Birmingham

Rudolf Böhm, Alsbach

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun in their paper – “Deep Residual Learning for Image Recognition”.

https://github.com/paddygoat/bioacoustics/blob/master/Train_and_deploy_bats/readme

https://en.wikipedia.org/wiki/Precision_and_recall

i Www.tensorflow.org

ii www.tierstimmenarchiv.de

iii Rudolf Böhm, Alsbach-Hähnlein

iv https://github.com/paddygoat/bioacoustics/blob/master/Train_and_deploy_bats/readme (house_keys, Junk_A … Junk_F)

v https://cran.r-project.org/web/packages/bioacoustics/index.html

vi https://en.wikipedia.org/wiki/Precision_and_recall