Assignment : 4

Modulation Classification

Mohamed Mostafa Badran 18011621
Seif El Din Wael Salem 18010832
Ziad Hassan Mahmoud 18010720

Contents

1. Data format explanation

The data retrieved is formatted as a dictionary and it consists of the following:

- Key
  - A label (one of 10 labels).
  - The SNR of the Channels.
- Value
  - Data of 200 Samples for each Key.
  - Each sample has 2 channels of 128 elements.

In order to train the data, we needed first to visualize the data, since this will emphasize the training parameters. Figure 1 shows the raw data representation and how noise is obvious. Figure 2 shows that the data is nearly similar. The combination used is represented by the equation (1.1).

$$data = data * gradient/Integration \tag{1.1}$$

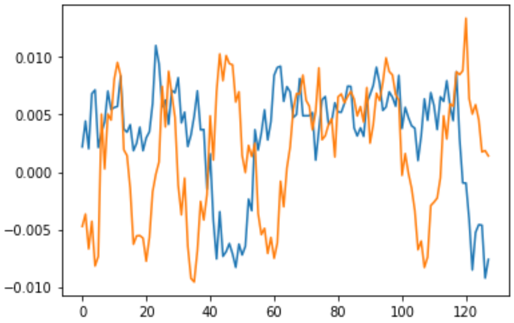


Figure 1.1: A raw sample representation

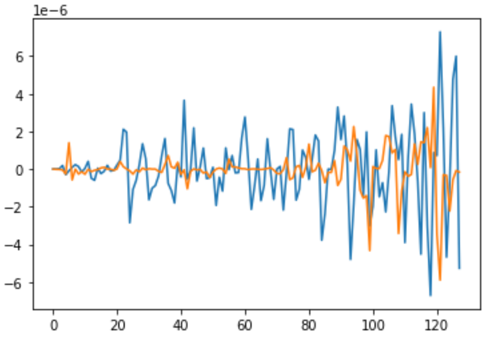
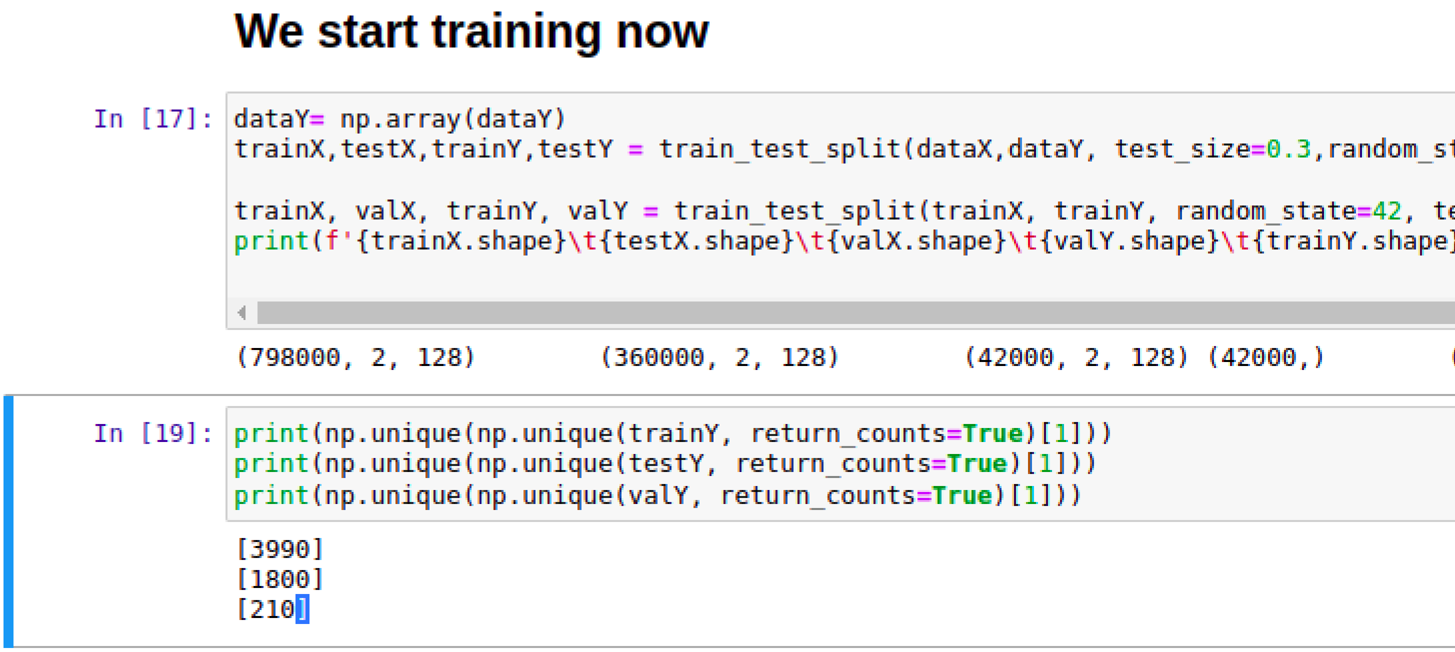


Figure 1.2: After combination with integration and gradient

2. Data preprocessing techniques used

2.0.1 Preparation

In order to start training the model, data was split to test and train data of ratio 30% and 70% respectively. Next, the training data is split to training and validation with ratio 95 to 5. Balance was put into consideration in order to avoid making the data being biased. This can be shown in figure 2.1.



We start training now

```
In [17]: dataY= np.array(dataY)
         trainX,testX,trainY,testY = train_test_split(dataX,dataY, test_size=0.3,random_st

         trainX, valX, trainY, valY = train_test_split(trainX, trainY, random_state=42, te
         print(f'{trainX.shape}\t{testX.shape}\t{valX.shape}\t{valY.shape}\t{trainY.shape}

         (798000, 2, 128)        (360000, 2, 128)        (42000, 2, 128) (42000,)        (
```

```
In [19]: print(np.unique(np.unique(trainY, return_counts=True)[1]))
         print(np.unique(np.unique(testY, return_counts=True)[1]))
         print(np.unique(np.unique(valY, return_counts=True)[1]))

         [3990]
         [1800]
         [210]
```

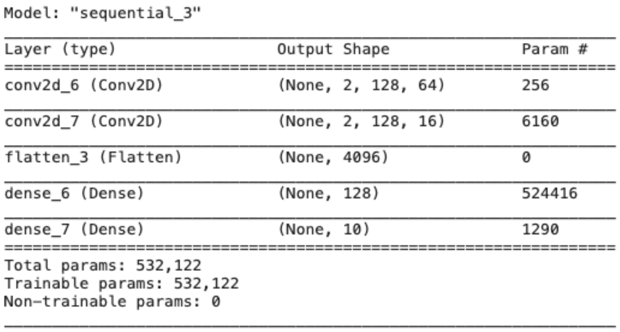
Figure 2.1: Cell 1 includes splitting, cell 2 to check that all labels have same size.

3. Method explanation

3 Models were introduced, a CNN model, Vanilla RNN model and an LSTM model. Each model was used as a classifier for the raw data. In addition the RNN was used also on the gradient and the integral data.

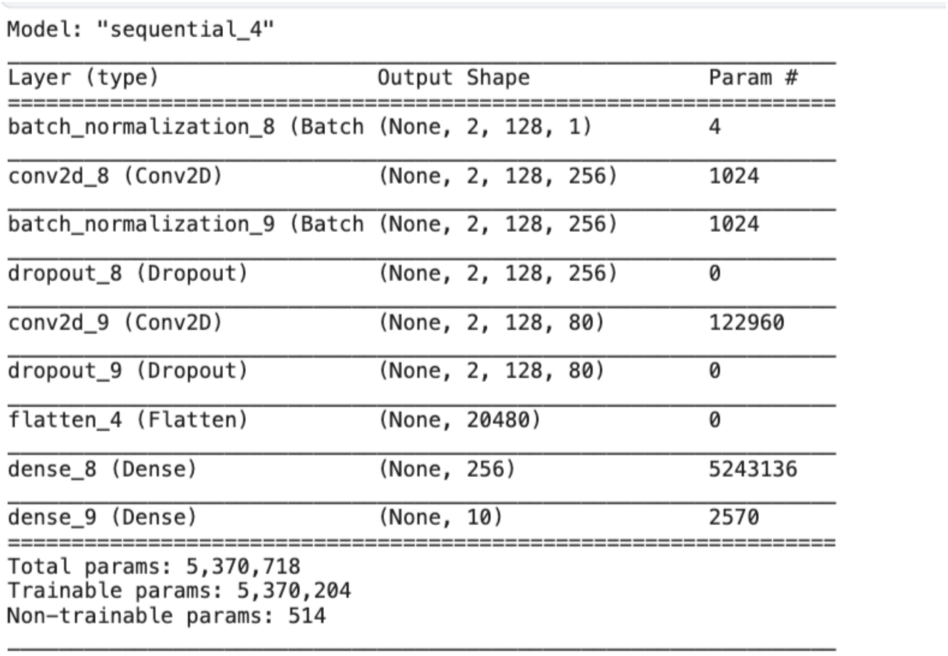
3.1 CNN

3.1.1 the requested CNN architecture



```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_6 (Conv2D)            (None, 2, 128, 64)        256
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 2, 128, 16)        6160
_________________________________________________________________
flatten_3 (Flatten)          (None, 4096)              0
_________________________________________________________________
dense_6 (Dense)              (None, 128)               524416
_________________________________________________________________
dense_7 (Dense)              (None, 10)                1290
=================================================================
Total params: 532,122
Trainable params: 532,122
Non-trainable params: 0
_________________________________________________________________
```

3.1.2 tuned CNN architecture

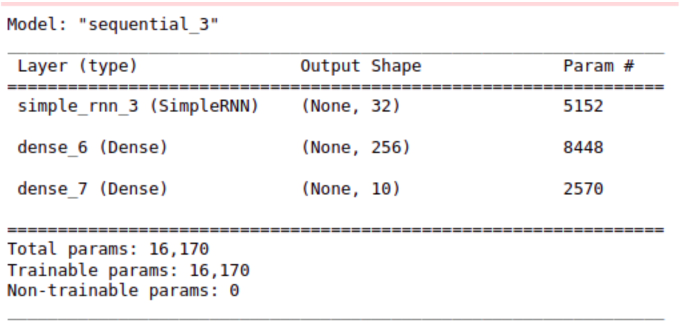


```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
batch_normalization_8 (Batch (None, 2, 128, 1)         4
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 2, 128, 256)       1024
_________________________________________________________________
batch_normalization_9 (Batch (None, 2, 128, 256)       1024
_________________________________________________________________
dropout_8 (Dropout)          (None, 2, 128, 256)       0
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 2, 128, 80)        122960
_________________________________________________________________
dropout_9 (Dropout)          (None, 2, 128, 80)        0
_________________________________________________________________
flatten_4 (Flatten)          (None, 20480)             0
_________________________________________________________________
dense_8 (Dense)              (None, 256)               5243136
_________________________________________________________________
dense_9 (Dense)              (None, 10)                2570
=================================================================
Total params: 5,370,718
Trainable params: 5,370,204
Non-trainable params: 514
_________________________________________________________________
```

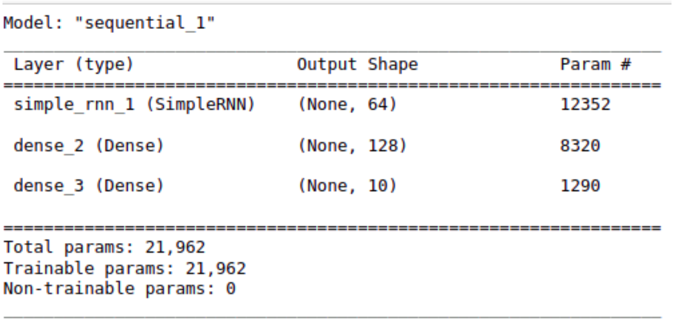
3.2 RNN

3.2.1 Raw Data Model

First we will talk about the model's architecture and then the results will be discussed. Figure 3.1 shows model used for raw data classification. We can see from figure 3.2 that there is no big difference between model used for raw and for the integral.



```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 simple_rnn_3 (SimpleRNN)    (None, 32)                5152

 dense_6 (Dense)             (None, 256)               8448

 dense_7 (Dense)             (None, 10)                2570

=================================================================
Total params: 16,170
Trainable params: 16,170
Non-trainable params: 0
_________________________________________________________________
```

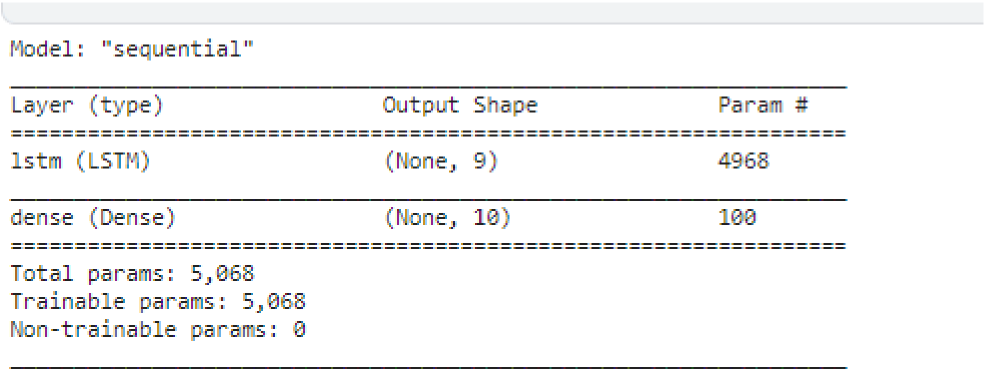


```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 simple_rnn_1 (SimpleRNN)    (None, 64)                12352

 dense_2 (Dense)             (None, 128)               8320

 dense_3 (Dense)             (None, 10)                1290

=================================================================
Total params: 21,962
Trainable params: 21,962
Non-trainable params: 0
_________________________________________________________________
```

3.3 LTSM

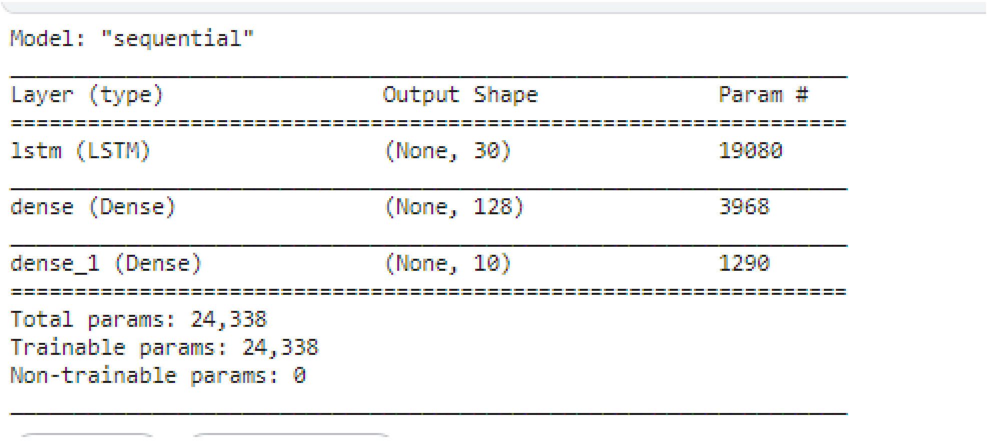
First we will talk about the model's architecture and then the results will be discussed

3.3.1 first-model-Without-Dense-layers



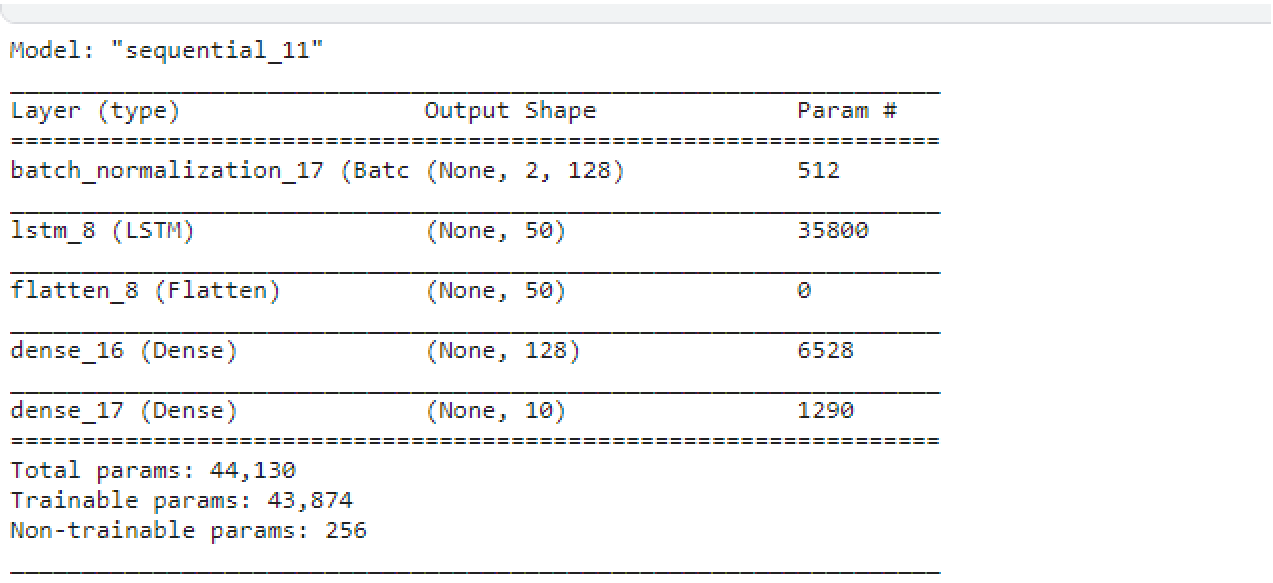
```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 9)                 4968
_________________________________________________________________
dense (Dense)                (None, 10)                100
=================================================================
Total params: 5,068
Trainable params: 5,068
Non-trainable params: 0
_________________________________________________________________
```

3.3.2 model-before-tunning-the-parameters



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
lstm (LSTM)                  (None, 30)                19080     
_________________________________________________________________
dense (Dense)                (None, 128)               3968      
_________________________________________________________________
dense_1 (Dense)              (None, 10)                1290      
=================================================================
Total params: 24,338
Trainable params: 24,338
Non-trainable params: 0
_________________________________________________________________
```

3.3.3 model-after-tunning-the-parameters

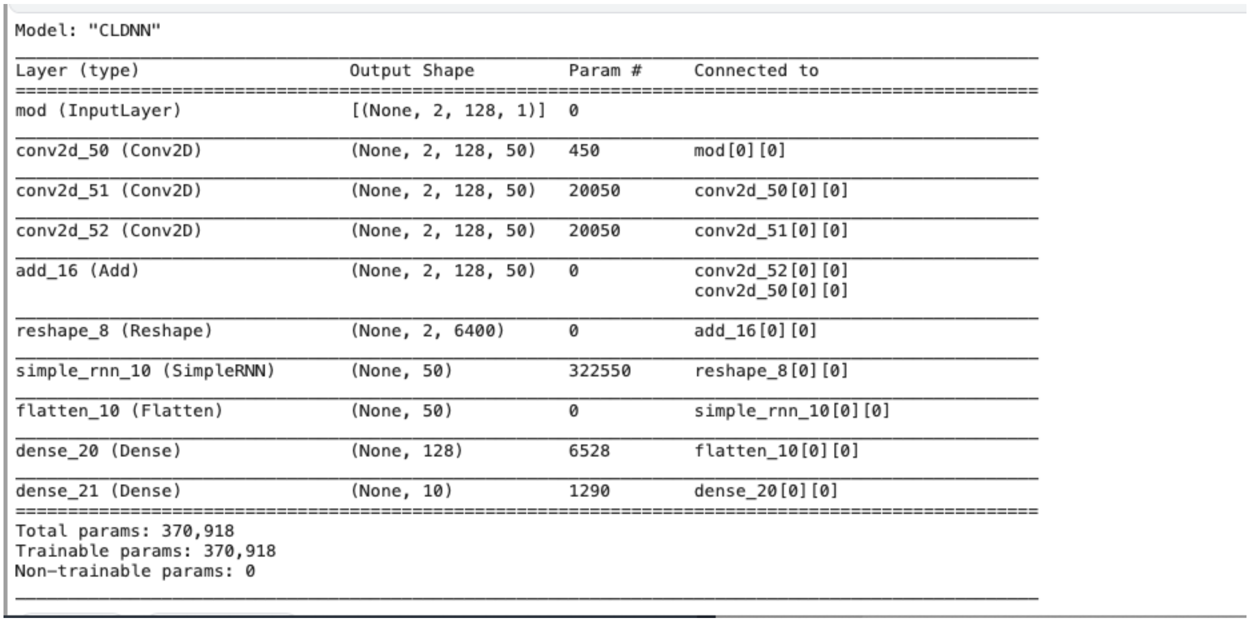


```
Model: "sequential_11"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
batch_normalization_17 (Batc (None, 2, 128)            512       
_________________________________________________________________
lstm_8 (LSTM)                (None, 50)                35800     
_________________________________________________________________
flatten_8 (Flatten)          (None, 50)                0         
_________________________________________________________________
dense_16 (Dense)             (None, 128)               6528      
_________________________________________________________________
dense_17 (Dense)             (None, 10)                1290      
=================================================================
Total params: 44,130
Trainable params: 43,874
Non-trainable params: 256
_________________________________________________________________
```

3.4 Conv-Lstm

first we built a CLDNN model

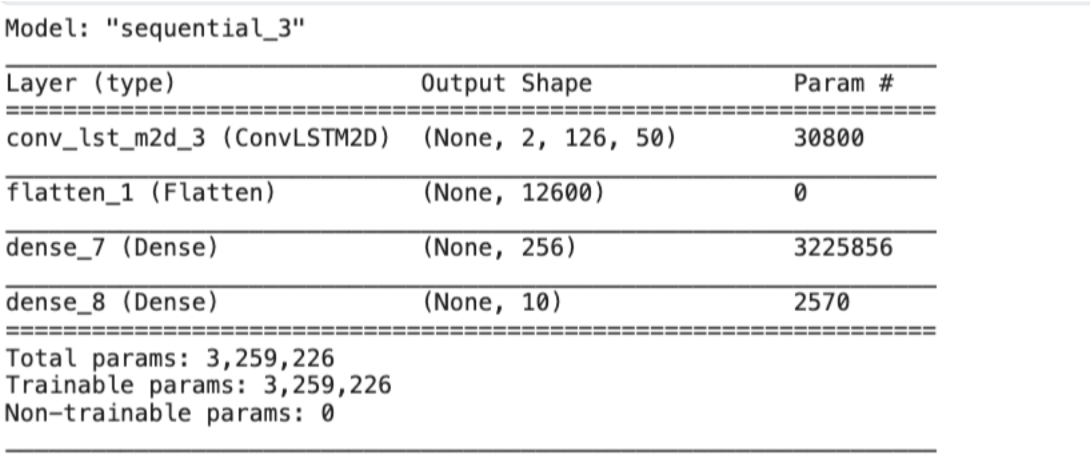
3.4.1 CLDNN model



```
Model: "CLDNN"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
mod (InputLayer)                [(None, 2, 128, 1)]  0
__________________________________________________________________________________________________
conv2d_50 (Conv2D)              (None, 2, 128, 50)   450         mod[0][0]
__________________________________________________________________________________________________
conv2d_51 (Conv2D)              (None, 2, 128, 50)   20050       conv2d_50[0][0]
__________________________________________________________________________________________________
conv2d_52 (Conv2D)              (None, 2, 128, 50)   20050       conv2d_51[0][0]
__________________________________________________________________________________________________
add_16 (Add)                    (None, 2, 128, 50)   0           conv2d_52[0][0]
                                                                 conv2d_50[0][0]
__________________________________________________________________________________________________
reshape_8 (Reshape)             (None, 2, 6400)      0           add_16[0][0]
__________________________________________________________________________________________________
simple_rnn_10 (SimpleRNN)       (None, 50)           322550      reshape_8[0][0]
__________________________________________________________________________________________________
flatten_10 (Flatten)            (None, 50)           0           simple_rnn_10[0][0]
__________________________________________________________________________________________________
dense_20 (Dense)                (None, 128)          6528        flatten_10[0][0]
__________________________________________________________________________________________________
dense_21 (Dense)                (None, 10)           1290        dense_20[0][0]
==================================================================================================
Total params: 370,918
Trainable params: 370,918
Non-trainable params: 0
__________________________________________________________________________________________________
```

second we built a model with lstm2dcnn module

3.4.2 lstm2dcnn-module

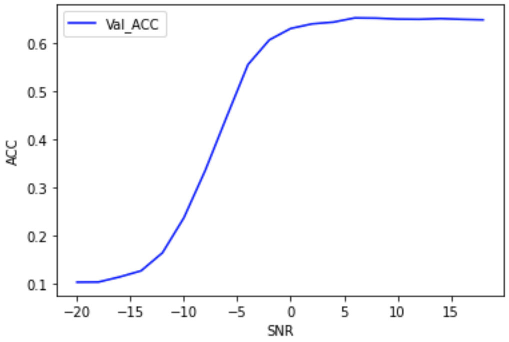


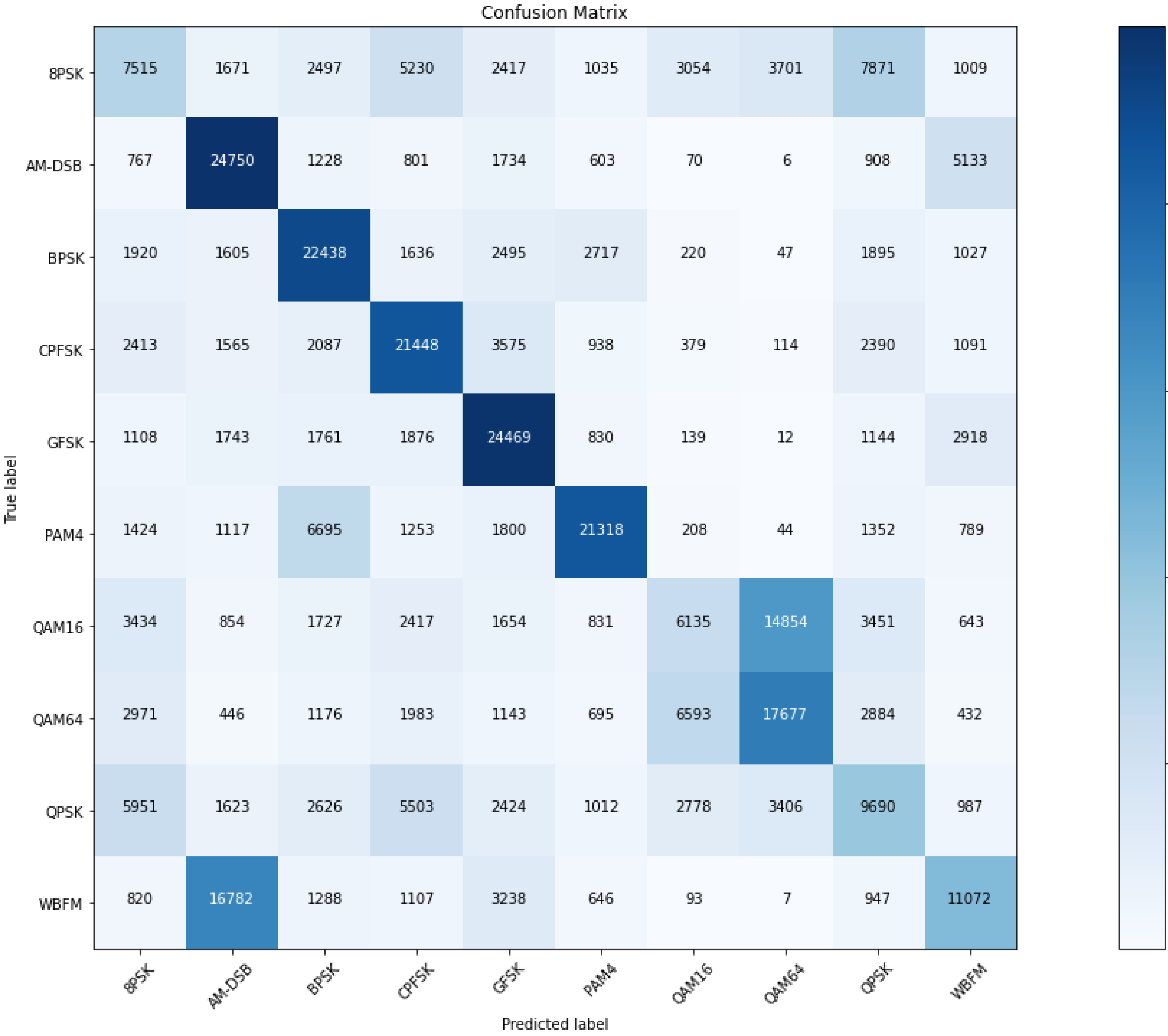
```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv_lst_m2d_3 (ConvLSTM2D)  (None, 2, 126, 50)        30800
_________________________________________________________________
flatten_1 (Flatten)          (None, 12600)             0
_________________________________________________________________
dense_7 (Dense)              (None, 256)               3225856
_________________________________________________________________
dense_8 (Dense)              (None, 10)                2570
=================================================================
Total params: 3,259,226
Trainable params: 3,259,226
Non-trainable params: 0
_________________________________________________________________
```

4. Results

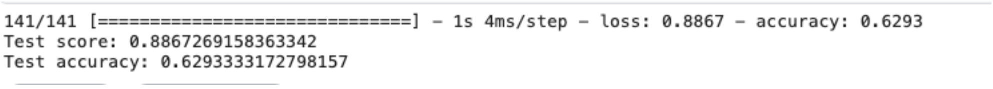
4.1 CNN

First the required CNN architecture



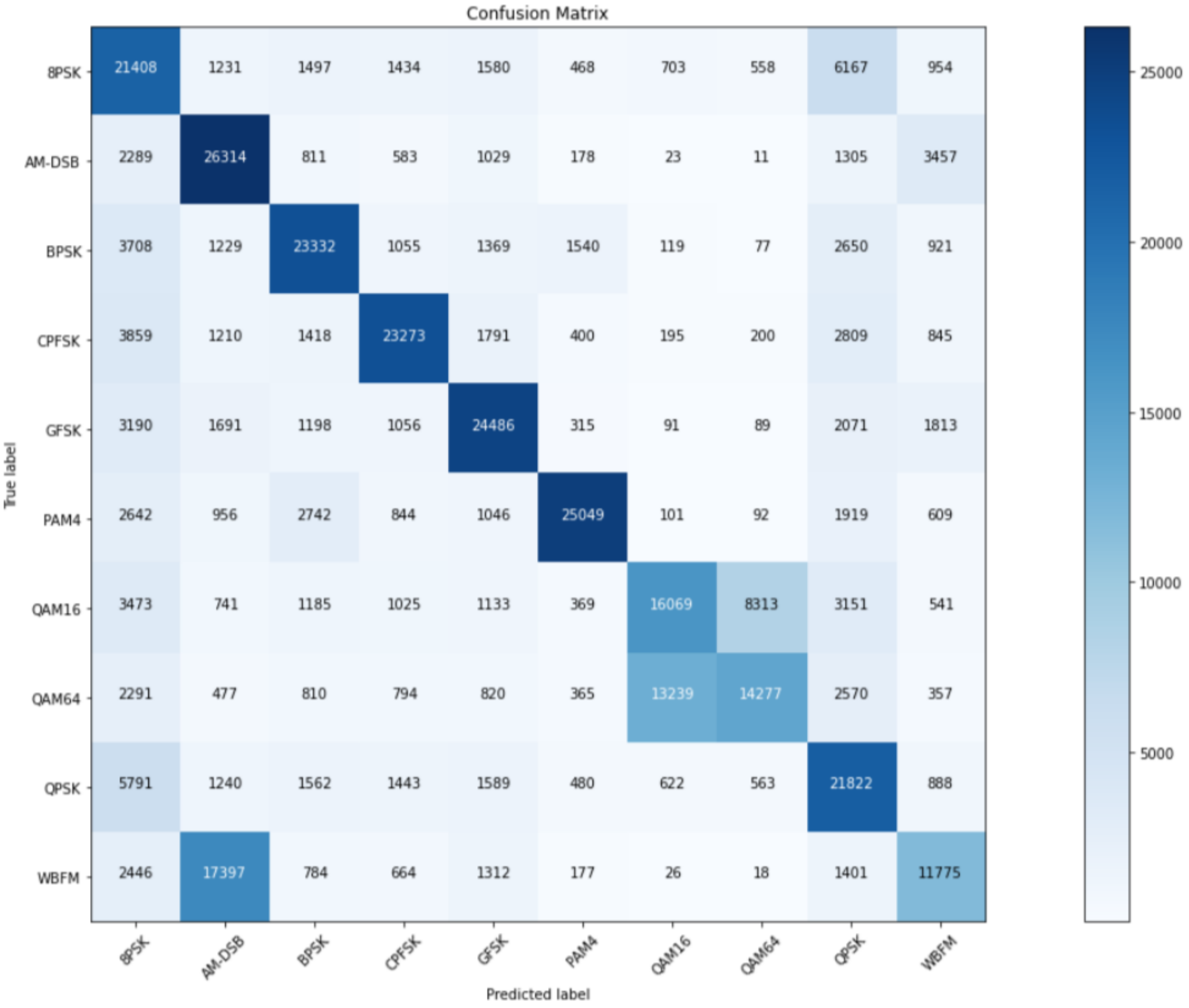


	8PSK	AM-DSB	BPSK	CPFSK	GFSK	PAM4	QAM16	QAM64	QPSK	WBFM
8PSK	7515	1671	2497	5230	2417	1035	3054	3701	7871	1009
AM-DSB	767	24750	1228	801	1734	603	70	6	908	5133
BPSK	1920	1605	22438	1636	2495	2717	220	47	1895	1027
CPFSK	2413	1565	2087	21448	3575	938	379	114	2390	1091
GFSK	1108	1743	1761	1876	24469	830	139	12	1144	2918
PAM4	1424	1117	6695	1253	1800	21318	208	44	1352	789
QAM16	3434	854	1727	2417	1654	831	6135	14854	3451	643
QAM64	2971	446	1176	1983	1143	695	6593	17677	2884	432
QPSK	5951	1623	2626	5503	2424	1012	2778	3406	9690	987
WBFM	820	16782	1288	1107	3238	646	93	7	947	11072

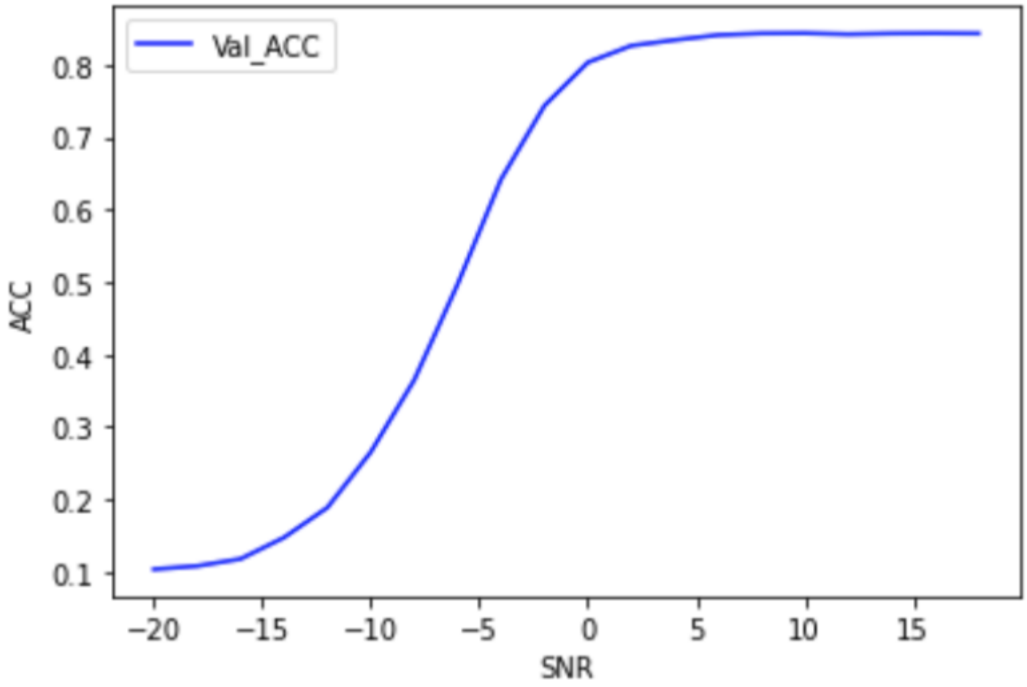


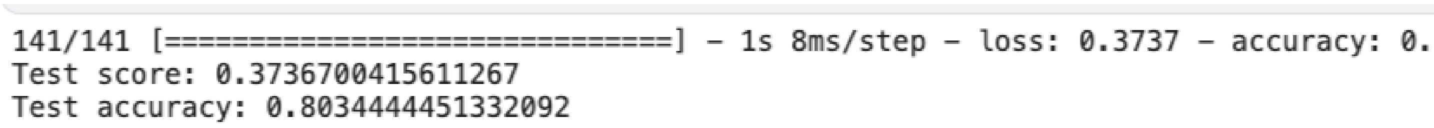
```
141/141 [==============================] - 1s 4ms/step - loss: 0.8867 - accuracy: 0.6293
Test score: 0.8867269158363342
Test accuracy: 0.6293333172798157
```

Second the tuned CNN architecture



True label \ Predicted label	8PSK	AM-DSB	BPSK	CPFSK	GFSK	PAM4	QAM16	QAM64	QPSK	WBFM
8PSK	21408	1231	1497	1434	1580	468	703	558	6167	954
AM-DSB	2289	26314	811	583	1029	178	23	11	1305	3457
BPSK	3708	1229	23332	1055	1369	1540	119	77	2650	921
CPFSK	3859	1210	1418	23273	1791	400	195	200	2809	845
GFSK	3190	1691	1198	1056	24486	315	91	89	2071	1813
PAM4	2642	956	2742	844	1046	25049	101	92	1919	609
QAM16	3473	741	1185	1025	1133	369	16069	8313	3151	541
QAM64	2291	477	810	794	820	365	13239	14277	2570	357
QPSK	5791	1240	1562	1443	1589	480	622	563	21822	888
WBFM	2446	17397	784	664	1312	177	26	18	1401	11775

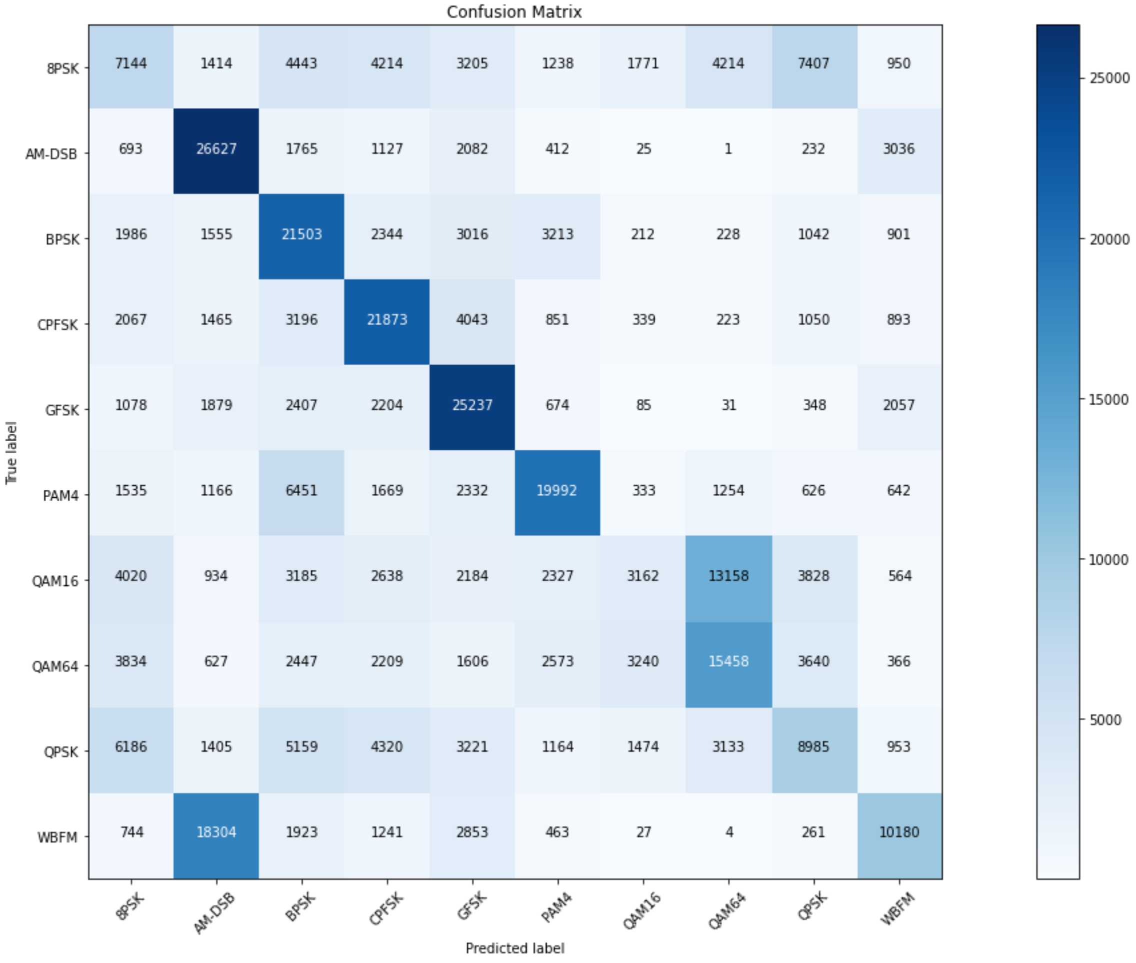




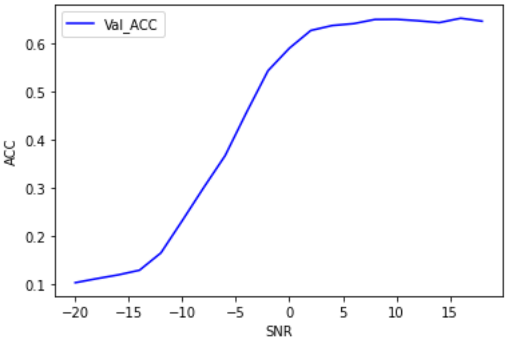
```
141/141 [==============================] - 1s 8ms/step - loss: 0.3737 - accuracy: 0.
Test score: 0.3736700415611267
Test accuracy: 0.8034444451332092
```

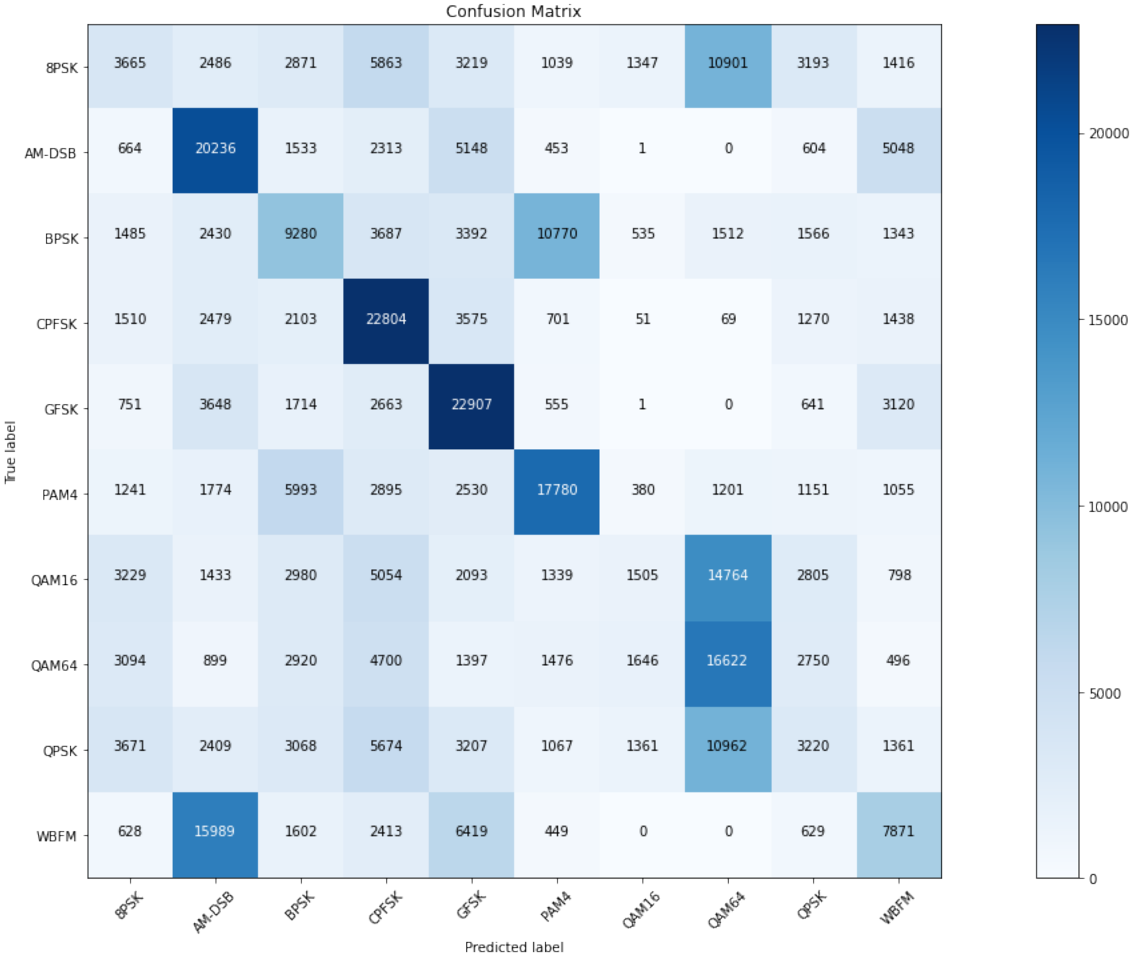
4.2 RNN

We will list the results with captions carrying the conclusion.



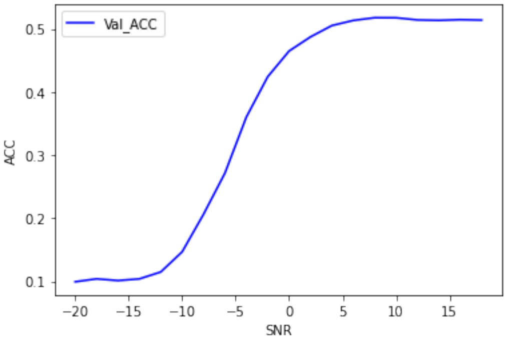
	8PSK	AM-DSB	BPSK	CPFSK	GFSK	PAM4	QAM16	QAM64	QPSK	WBFM
8PSK	7144	1414	4443	4214	3205	1238	1771	4214	7407	950
AM-DSB	693	26627	1765	1127	2082	412	25	1	232	3036
BPSK	1986	1555	21503	2344	3016	3213	212	228	1042	901
CPFSK	2067	1465	3196	21873	4043	851	339	223	1050	893
GFSK	1078	1879	2407	2204	25237	674	85	31	348	2057
PAM4	1535	1166	6451	1669	2332	19992	333	1254	626	642
QAM16	4020	934	3185	2638	2184	2327	3162	13158	3828	564
QAM64	3834	627	2447	2209	1606	2573	3240	15458	3640	366
QPSK	6186	1405	5159	4320	3221	1164	1474	3133	8985	953
WBFM	744	18304	1923	1241	2853	463	27	4	261	10180

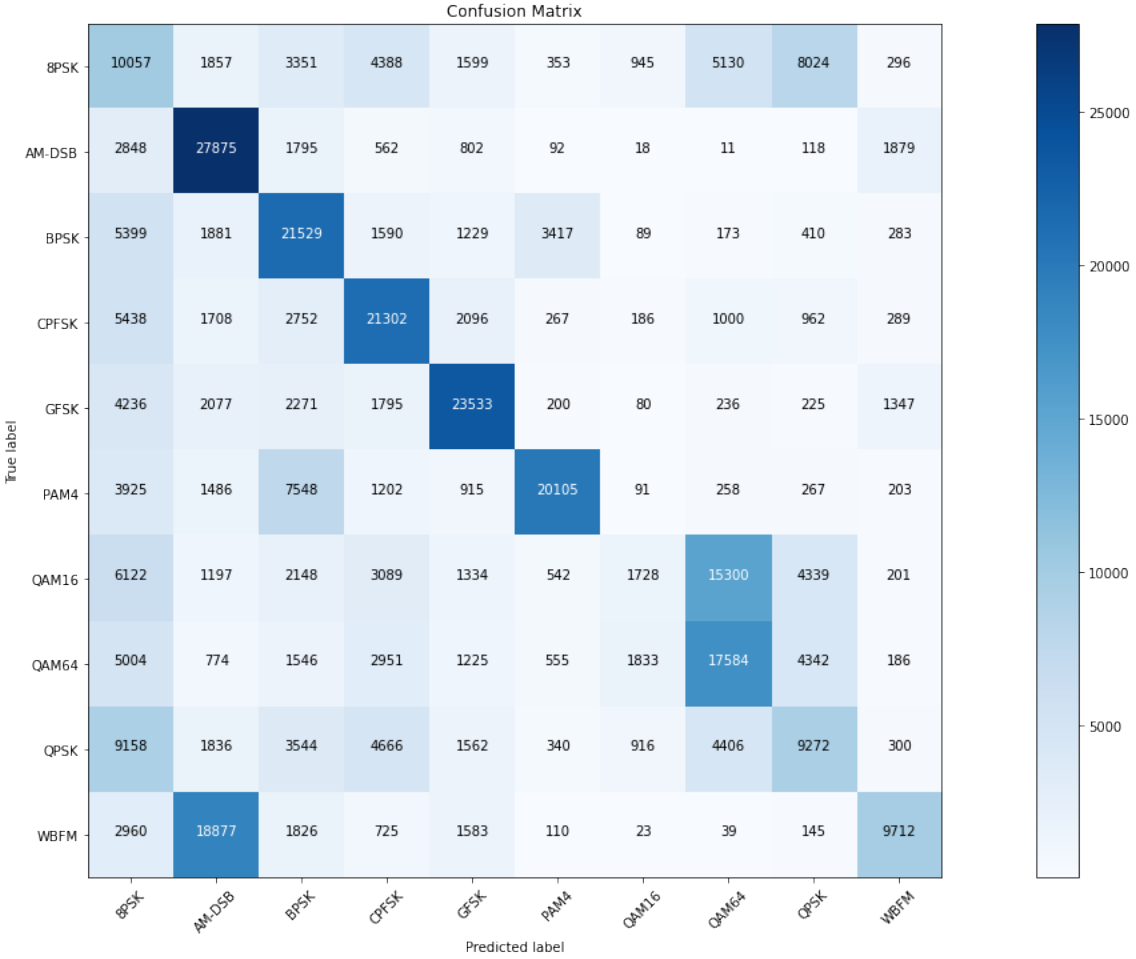




Confusion Matrix

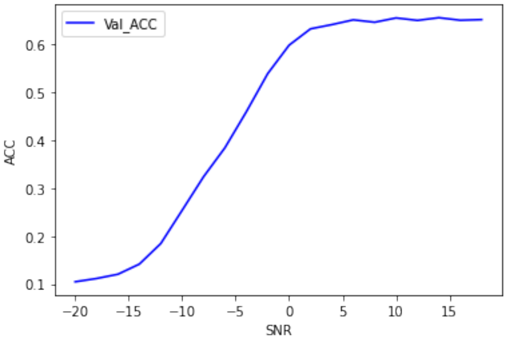
True label \ Predicted label	8PSK	AM-DSB	BPSK	CPFSK	GFSK	PAM4	QAM16	QAM64	QPSK	WBFM
8PSK	3665	2486	2871	5863	3219	1039	1347	10901	3193	1416
AM-DSB	664	20236	1533	2313	5148	453	1	0	604	5048
BPSK	1485	2430	9280	3687	3392	10770	535	1512	1566	1343
CPFSK	1510	2479	2103	22804	3575	701	51	69	1270	1438
GFSK	751	3648	1714	2663	22907	555	1	0	641	3120
PAM4	1241	1774	5993	2895	2530	17780	380	1201	1151	1055
QAM16	3229	1433	2980	5054	2093	1339	1505	14764	2805	798
QAM64	3094	899	2920	4700	1397	1476	1646	16622	2750	496
QPSK	3671	2409	3068	5674	3207	1067	1361	10962	3220	1361
WBFM	628	15989	1602	2413	6419	449	0	0	629	7871

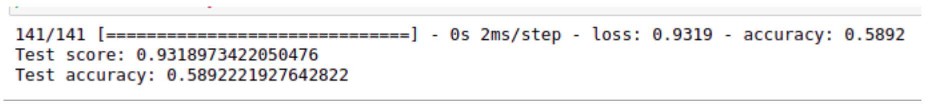




Confusion Matrix

True label \ Predicted label	8PSK	AM-DSB	BPSK	CPFSK	GFSK	PAM4	QAM16	QAM64	QPSK	WBFM
8PSK	10057	1857	3351	4388	1599	353	945	5130	8024	296
AM-DSB	2848	27875	1795	562	802	92	18	11	118	1879
BPSK	5399	1881	21529	1590	1229	3417	89	173	410	283
CPFSK	5438	1708	2752	21302	2096	267	186	1000	962	289
GFSK	4236	2077	2271	1795	23533	200	80	236	225	1347
PAM4	3925	1486	7548	1202	915	20105	91	258	267	203
QAM16	6122	1197	2148	3089	1334	542	1728	15300	4339	201
QAM64	5004	774	1546	2951	1225	555	1833	17584	4342	186
QPSK	9158	1836	3544	4666	1562	340	916	4406	9272	300
WBFM	2960	18877	1826	725	1583	110	23	39	145	9712

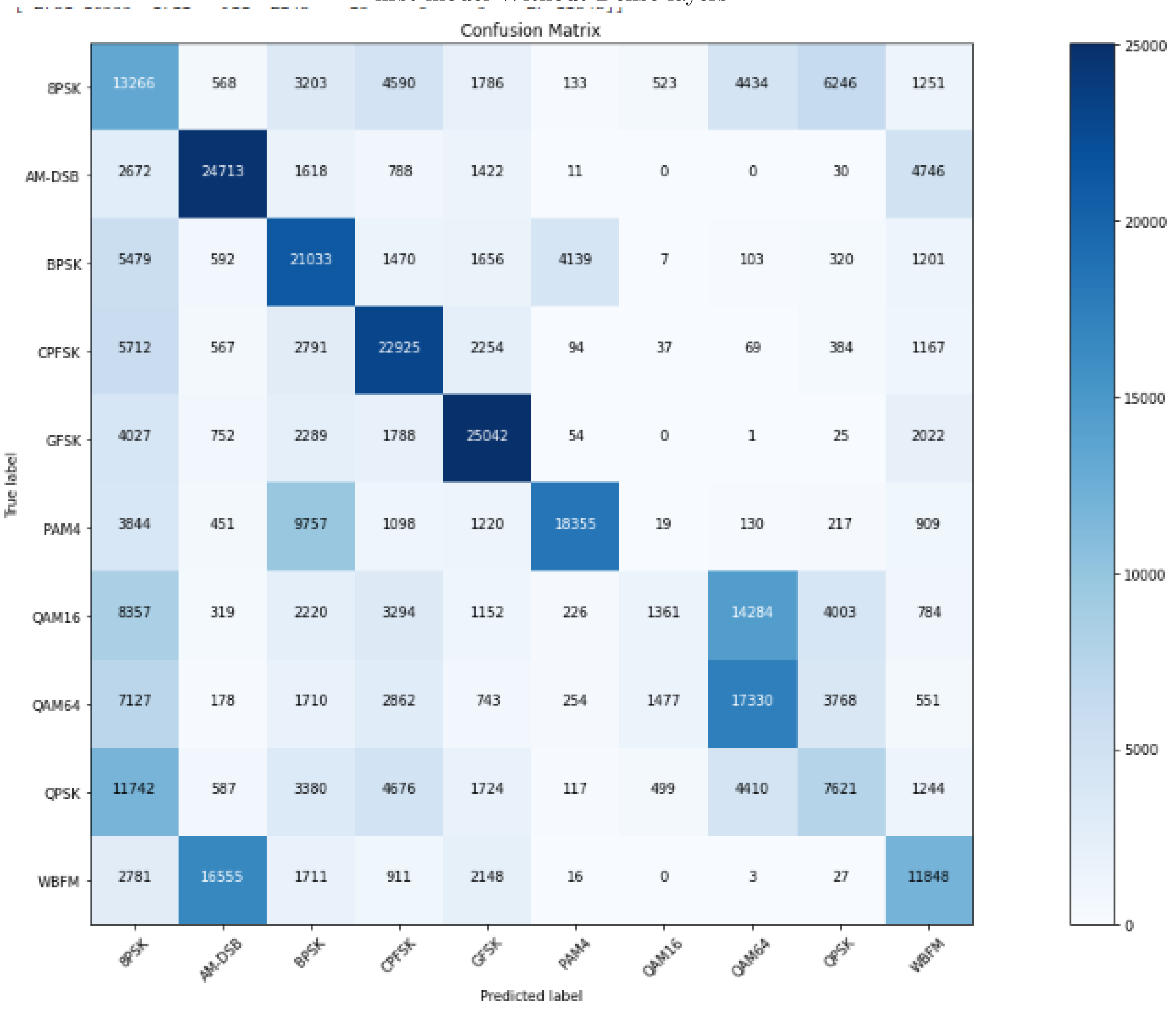




```
141/141 [==============================] - 0s 2ms/step - loss: 0.9319 - accuracy: 0.5892
Test score: 0.9318973422050476
Test accuracy: 0.5892221927642822
```

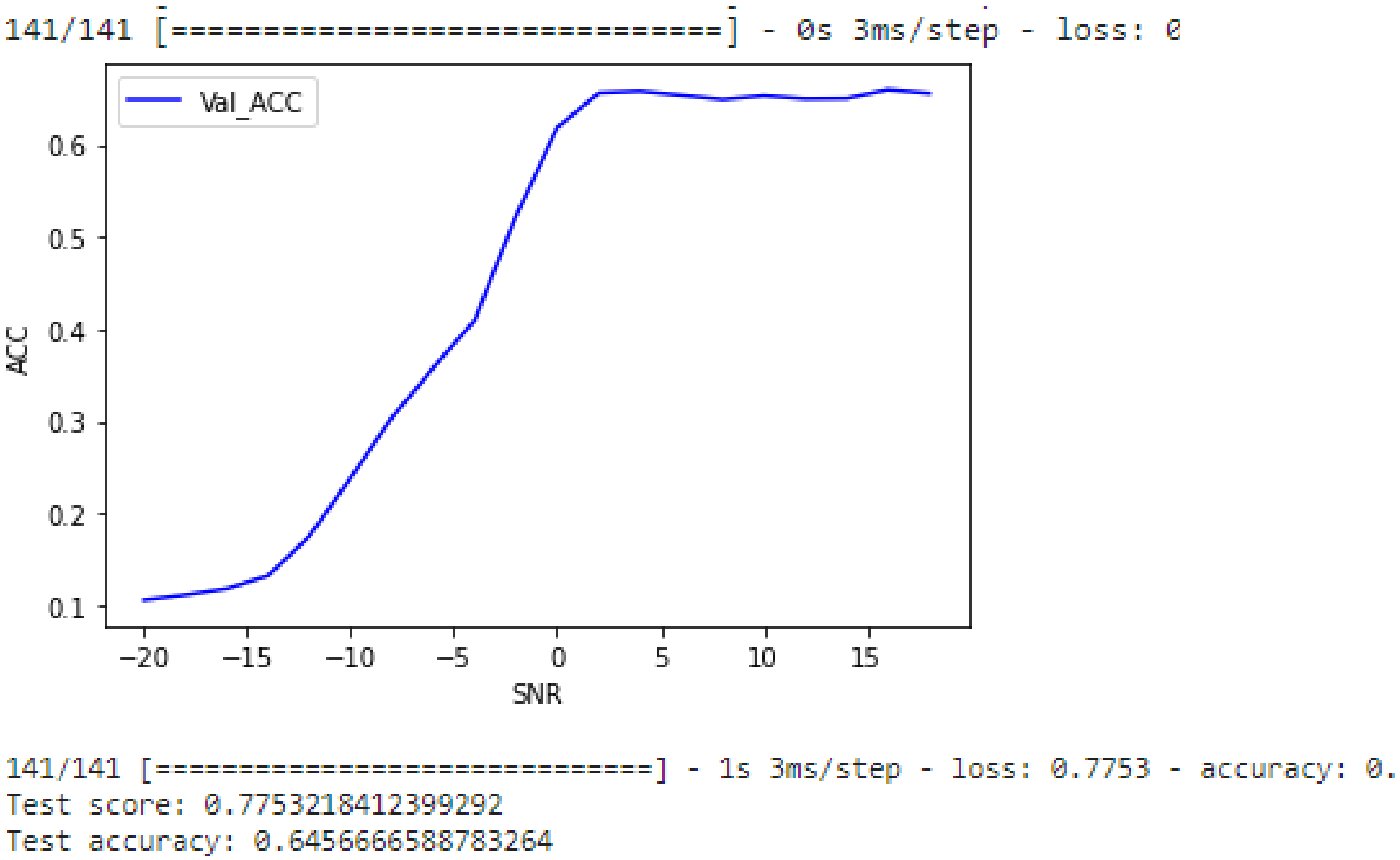
4.3 LTSM

We will list the results with captions carrying the conclusion. of two models we trie first-model-Without-Dense-layers



Confusion Matrix

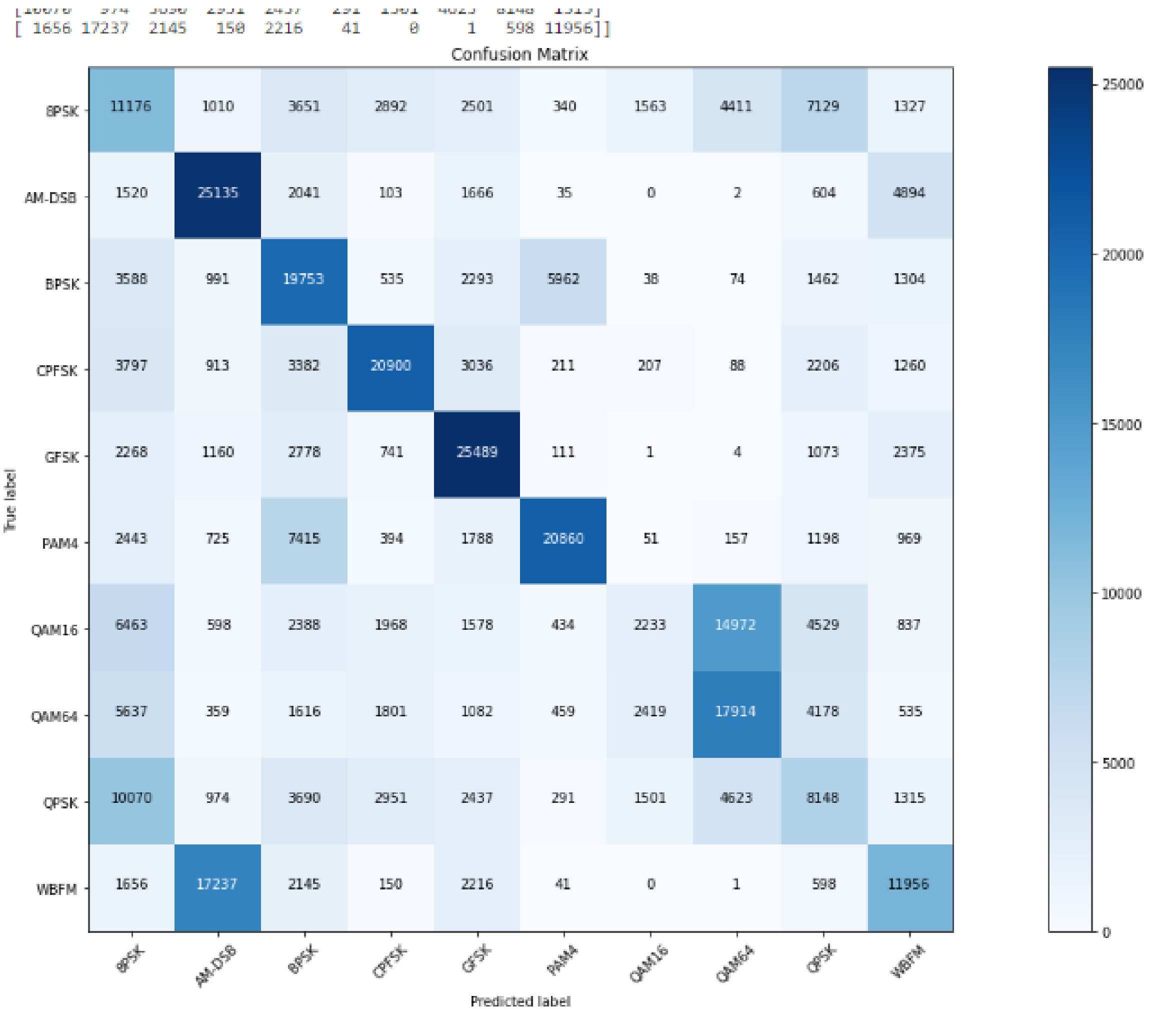
True label \ Predicted label	8PSK	AM-DSB	BPSK	CPFSK	GFSK	PAM4	QAM16	QAM64	QPSK	WBFM
8PSK	13266	568	3203	4590	1786	133	523	4434	6246	1251
AM-DSB	2672	24713	1618	788	1422	11	0	0	30	4746
BPSK	5479	592	21033	1470	1656	4139	7	103	320	1201
CPFSK	5712	567	2791	22925	2254	94	37	69	384	1167
GFSK	4027	752	2289	1788	25042	54	0	1	25	2022
PAM4	3844	451	9757	1098	1220	18355	19	130	217	909
QAM16	8357	319	2220	3294	1152	226	1361	14284	4003	784
QAM64	7127	178	1710	2862	743	254	1477	17330	3768	551
QPSK	11742	587	3380	4676	1724	117	499	4410	7621	1244
WBFM	2781	16555	1711	911	2148	16	0	3	27	11848



```
141/141 [==============================] - 0s 3ms/step - loss: 0
```

```
141/141 [==============================] - 1s 3ms/step - loss: 0.7753 - accuracy: 0.
Test score: 0.7753218412399292
Test accuracy: 0.6456666588783264
```

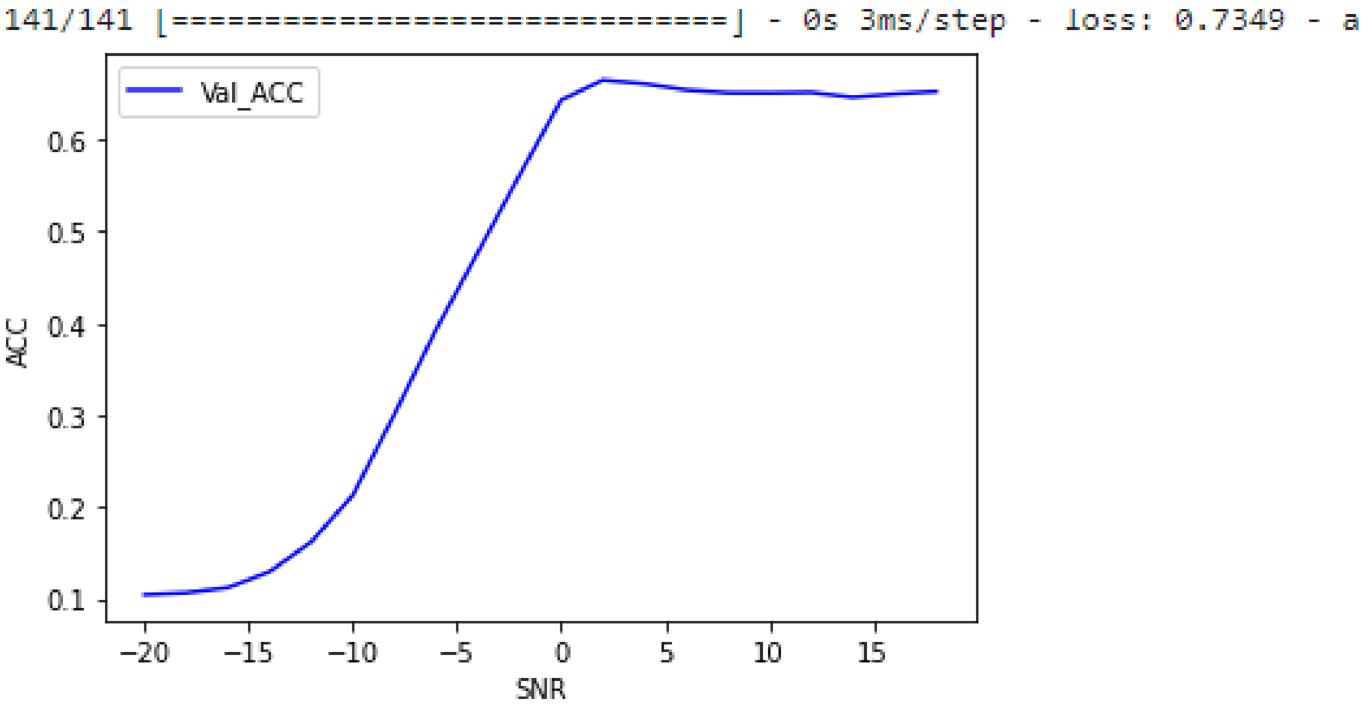
model-before-tunning-the-parameters



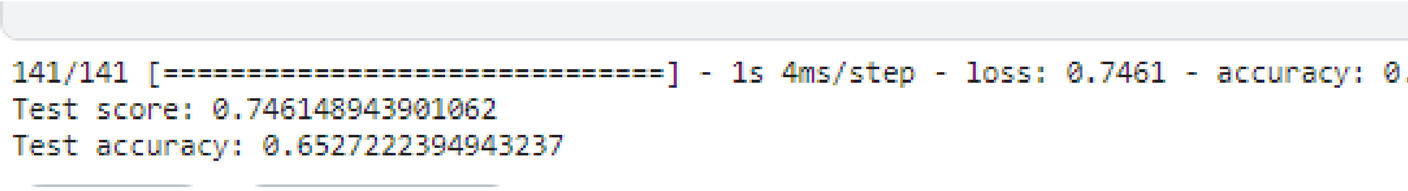
```
 [10070   974  3690  2951  2437   291  1501  4623  8148  1315]
 [ 1656 17237  2145   150  2216    41     0     1   598 11956]]
```

Confusion Matrix

True label \ Predicted label	8PSK	AM-DSB	BPSK	CPFSK	GFSK	PAM4	QAM16	QAM64	QPSK	WBFM
8PSK	11176	1010	3651	2892	2501	340	1563	4411	7129	1327
AM-DSB	1520	25135	2041	103	1666	35	0	2	604	4894
BPSK	3588	991	19753	535	2293	5962	38	74	1462	1304
CPFSK	3797	913	3382	20900	3036	211	207	88	2206	1260
GFSK	2268	1160	2778	741	25489	111	1	4	1073	2375
PAM4	2443	725	7415	394	1788	20860	51	157	1198	969
QAM16	6463	598	2388	1968	1578	434	2233	14972	4529	837
QAM64	5637	359	1616	1801	1082	459	2419	17914	4178	535
QPSK	10070	974	3690	2951	2437	291	1501	4623	8148	1315
WBFM	1656	17237	2145	150	2216	41	0	1	598	11956

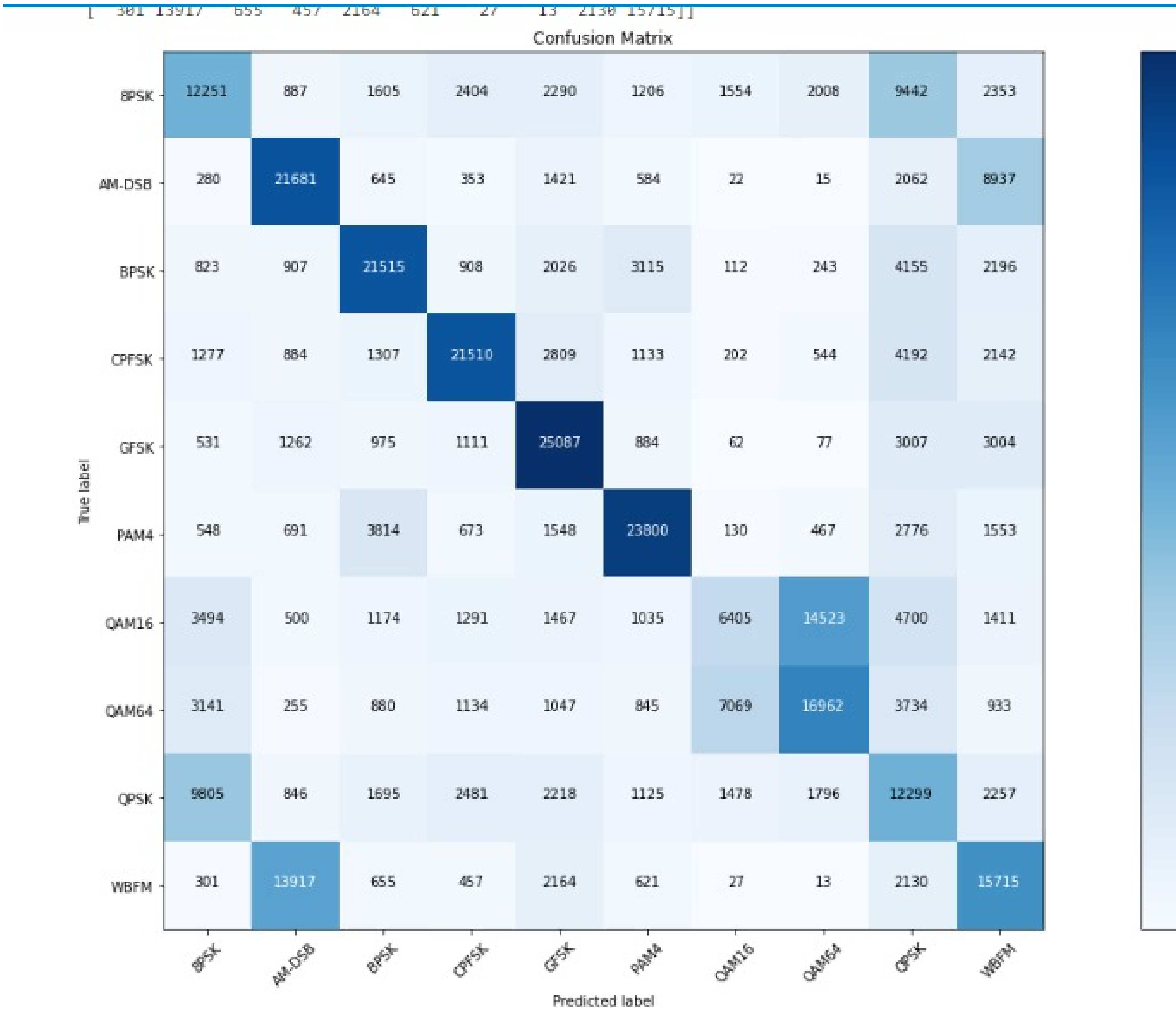


```
141/141 [==============================] - 0s 3ms/step - loss: 0.7349 - a
```



```
141/141 [==============================] - 1s 4ms/step - loss: 0.7461 - accuracy: 0.
Test score: 0.746148943901062
Test accuracy: 0.6527222394943237
```

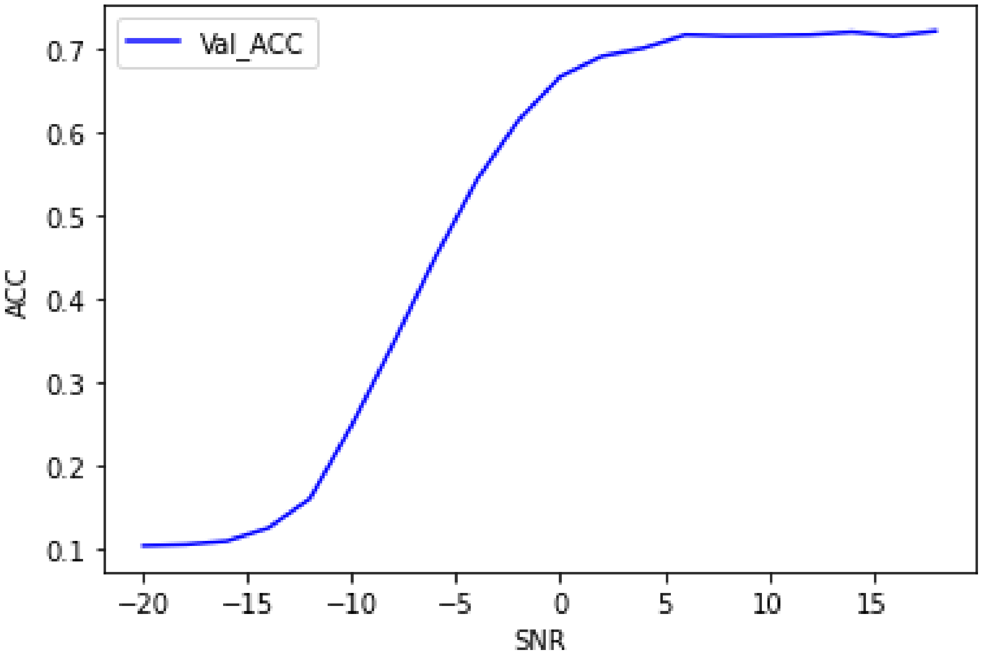
model-after-tunning-the-parameters

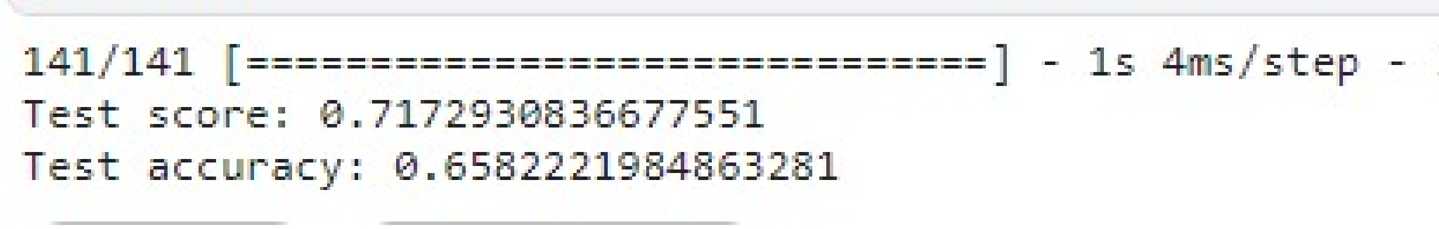


```
[  301 13917   655   457  2164   621    27    13  2130 15715]]
```

	8PSK	AM-DSB	BPSK	CPFSK	GFSK	PAM4	QAM16	QAM64	QPSK	WBFM
8PSK	12251	887	1605	2404	2290	1206	1554	2008	9442	2353
AM-DSB	280	21681	645	353	1421	584	22	15	2062	8937
BPSK	823	907	21515	908	2026	3115	112	243	4155	2196
CPFSK	1277	884	1307	21510	2809	1133	202	544	4192	2142
GFSK	531	1262	975	1111	25087	884	62	77	3007	3004
PAM4	548	691	3814	673	1548	23800	130	467	2776	1553
QAM16	3494	500	1174	1291	1467	1035	6405	14523	4700	1411
QAM64	3141	255	880	1134	1047	845	7069	16962	3734	933
QPSK	9805	846	1695	2481	2218	1125	1478	1796	12299	2257
WBFM	301	13917	655	457	2164	621	27	13	2130	15715

```
141/141 [==============================] - 1s 4ms/step - loss: 0.5518 - accura
```

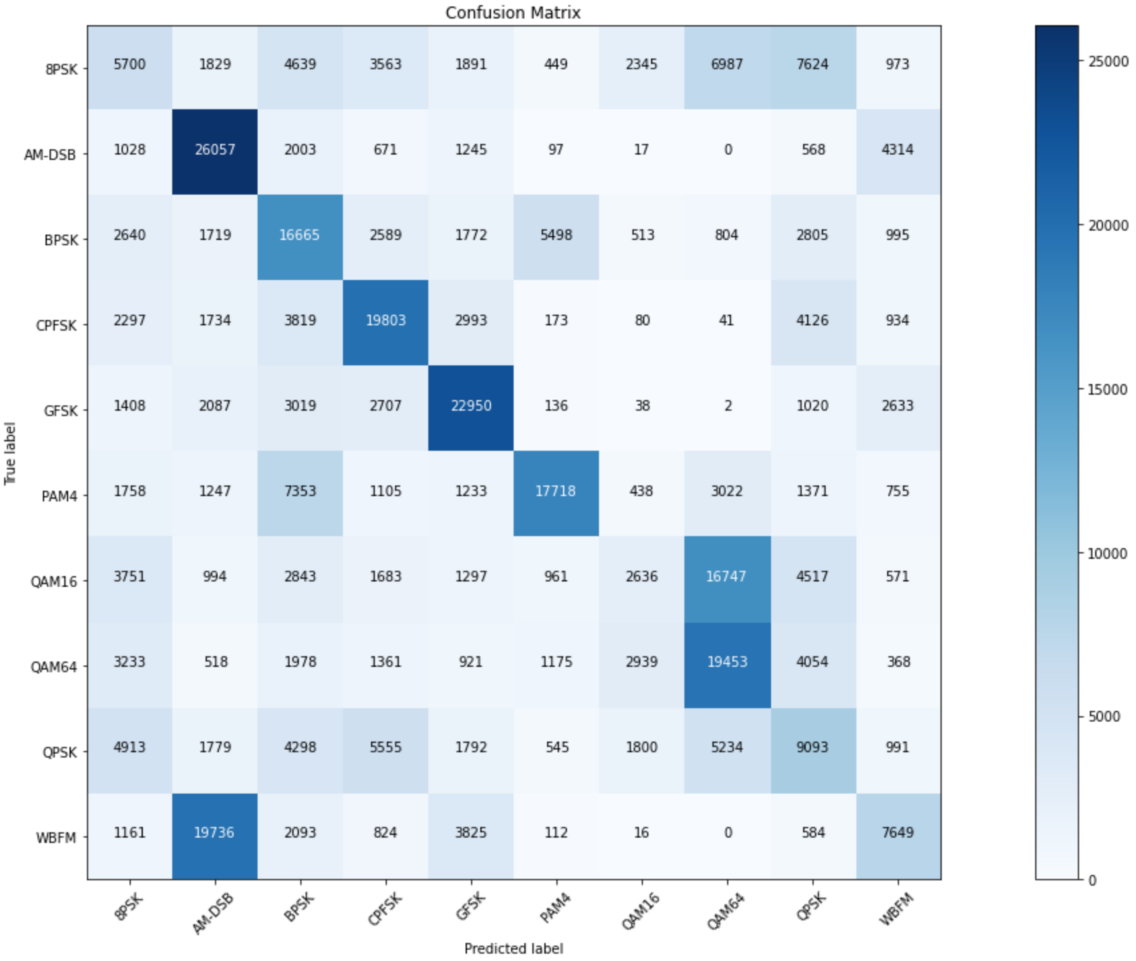




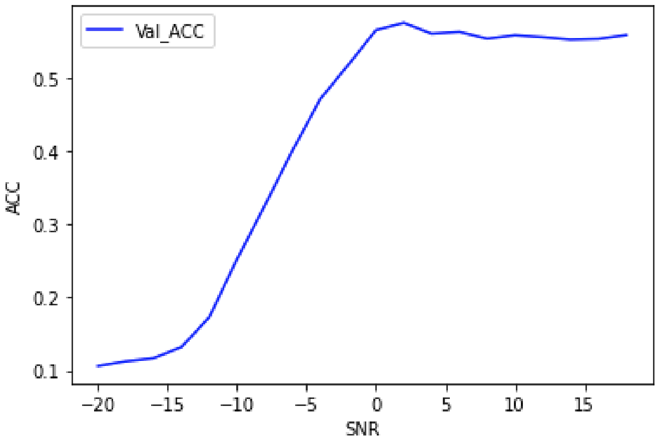
```
141/141 [==============================] - 1s 4ms/step -
Test score: 0.7172930836677551
Test accuracy: 0.6582221984863281
```

4.4 ConvLstm

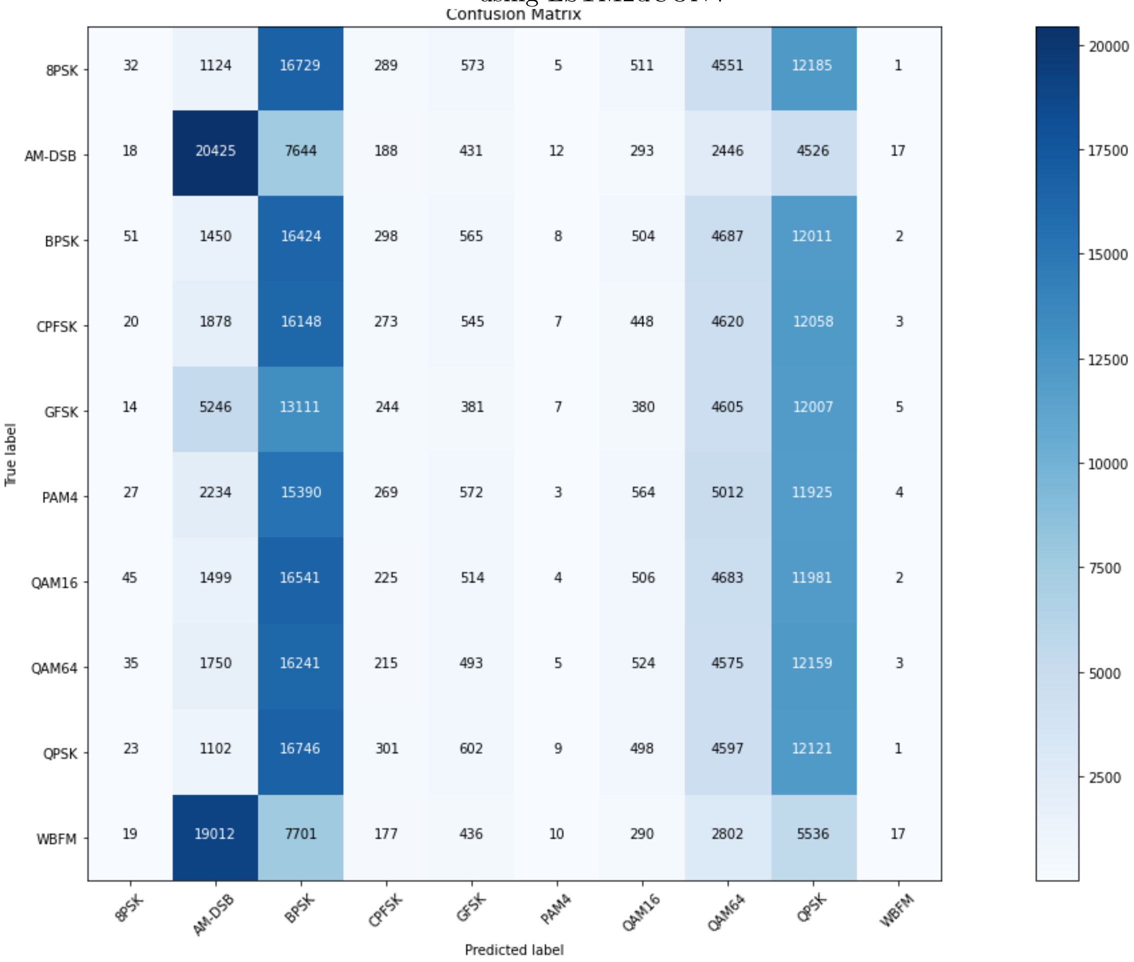
using CLDNN


Confusion Matrix

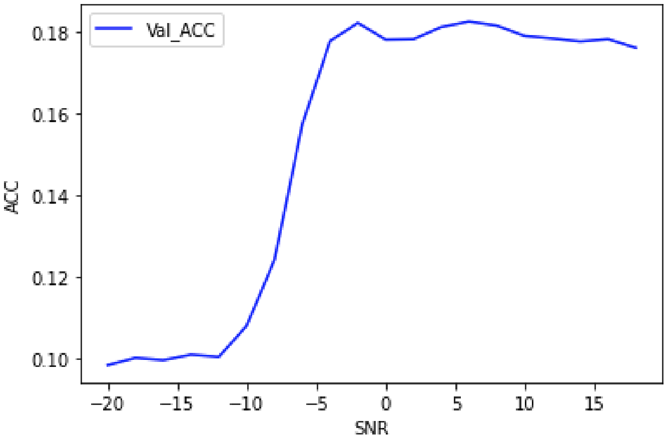
True label \ Predicted label	8PSK	AM-DSB	BPSK	CPFSK	GFSK	PAM4	QAM16	QAM64	QPSK	WBFM
8PSK	5700	1829	4639	3563	1891	449	2345	6987	7624	973
AM-DSB	1028	26057	2003	671	1245	97	17	0	568	4314
BPSK	2640	1719	16665	2589	1772	5498	513	804	2805	995
CPFSK	2297	1734	3819	19803	2993	173	80	41	4126	934
GFSK	1408	2087	3019	2707	22950	136	38	2	1020	2633
PAM4	1758	1247	7353	1105	1233	17718	438	3022	1371	755
QAM16	3751	994	2843	1683	1297	961	2636	16747	4517	571
QAM64	3233	518	1978	1361	921	1175	2939	19453	4054	368
QPSK	4913	1779	4298	5555	1792	545	1800	5234	9093	991
WBFM	1161	19736	2093	824	3825	112	16	0	584	7649



using LSTM2dCONV



	8PSK	AM-DSB	BPSK	CPFSK	GFSK	PAM4	QAM16	QAM64	QPSK	WBFM
8PSK	32	1124	16729	289	573	5	511	4551	12185	1
AM-DSB	18	20425	7644	188	431	12	293	2446	4526	17
BPSK	51	1450	16424	298	565	8	504	4687	12011	2
CPFSK	20	1878	16148	273	545	7	448	4620	12058	3
GFSK	14	5246	13111	244	381	7	380	4605	12007	5
PAM4	27	2234	15390	269	572	3	564	5012	11925	4
QAM16	45	1499	16541	225	514	4	506	4683	11981	2
QAM64	35	1750	16241	215	493	5	524	4575	12159	3
QPSK	23	1102	16746	301	602	9	498	4597	12121	1
WBFM	19	19012	7701	177	436	10	290	2802	5536	17



4.5 most-confusing-classes

1.QAM16
2.WBFM
3.8PSK

this report written by latex format and code attached to zip file