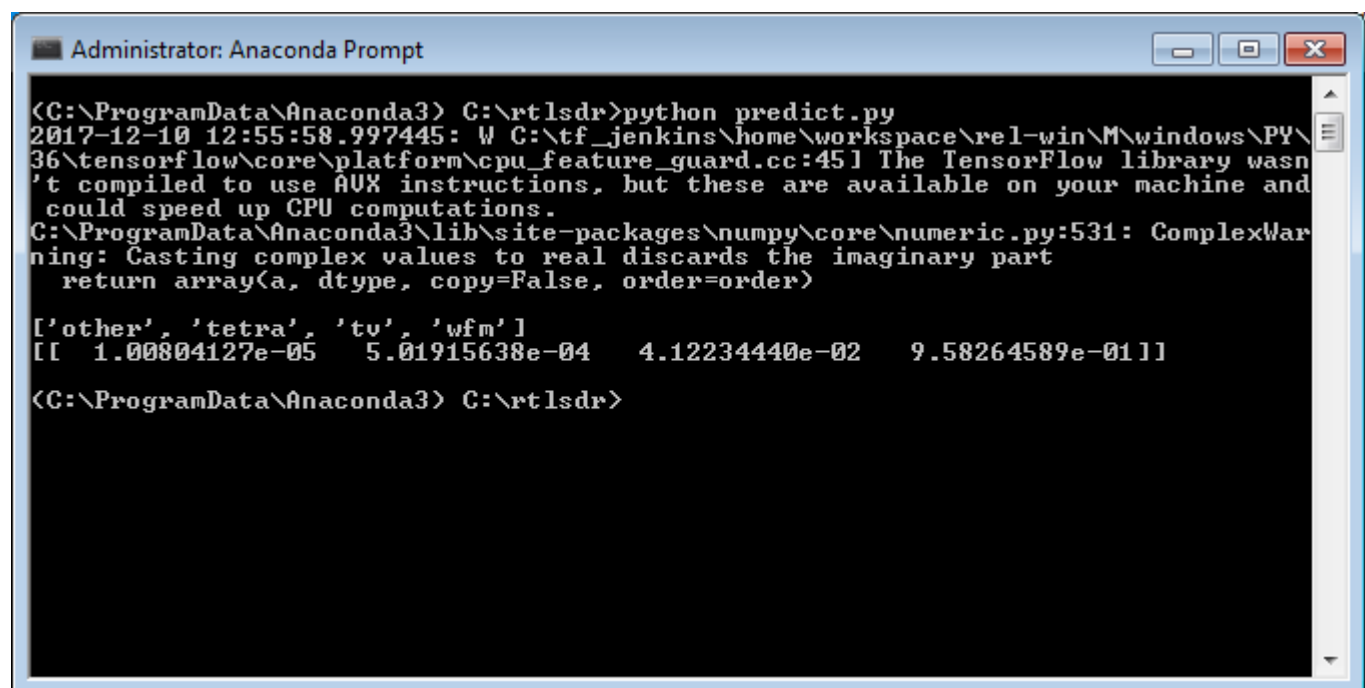


Neural Network signal recognition rtl_sdr

Description

Pre-trained version is able to scan desired frequency range and classify (better or worse) 4 kinds of found signals: wfm, tv secam carrier, tetra and others.

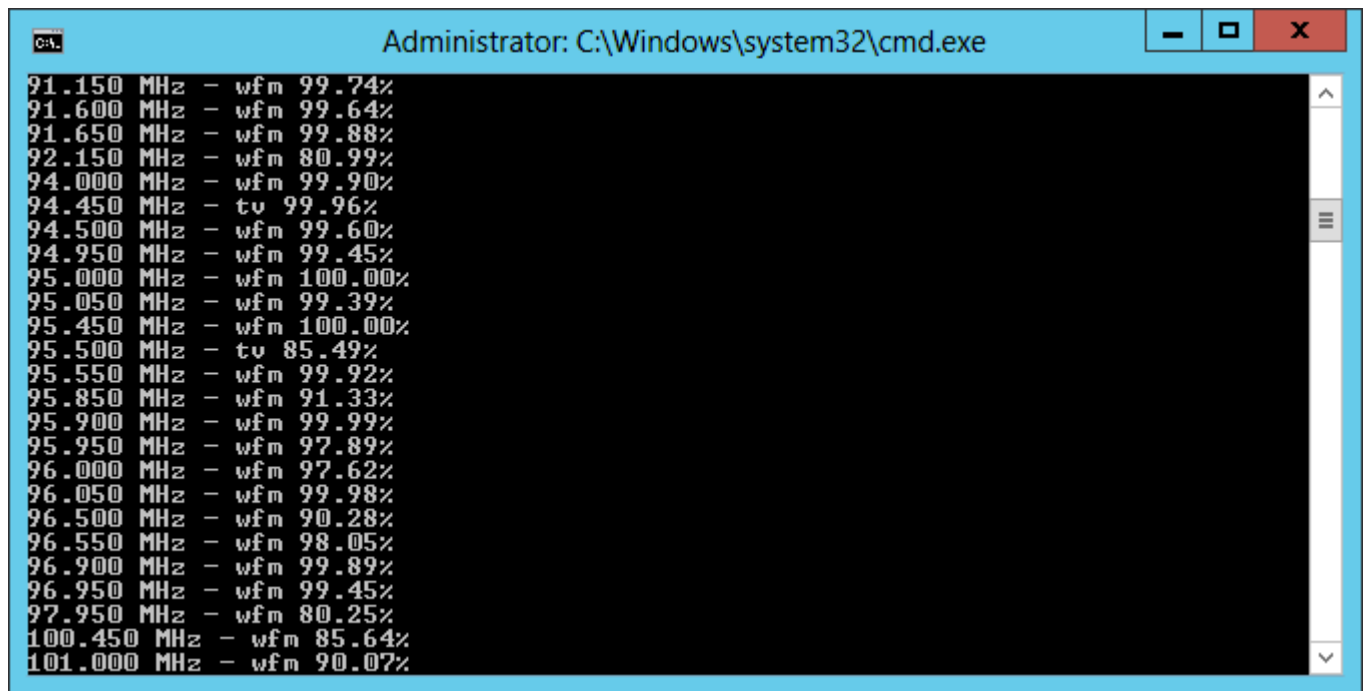
You may train neural network to classify your own signals. What you really need for, is a good videoadapter and lots of free time.

A screenshot of an Anaconda Prompt window titled "Administrator: Anaconda Prompt". The window shows the execution of a Python script named "predict.py" in the directory "C:\rtl_sdr". The output of the script is displayed in the terminal. It includes a warning about TensorFlow not being compiled for AVX instructions, a warning from NumPy about casting complex values to real, and a list of predicted signal types: ['other', 'tetra', 'tv', 'wfm']. Below this, a list of numerical values is shown: [1.00804127e-05, 5.01915638e-04, 4.12234440e-02, 9.58264589e-01].

```
<C:\ProgramData\Anaconda3> C:\rtl_sdr>python predict.py
2017-12-10 12:55:58.997445: W C:\tf_jenkins\home\workspace\rel-win\M\windows\PY\
36\tensorflow\core\platform\cpu_feature_guard.cc:45] The TensorFlow library wasn
't compiled to use AVX instructions, but these are available on your machine and
could speed up CPU computations.
C:\ProgramData\Anaconda3\lib\site-packages\numpy\core\numeric.py:531: ComplexWar
ning: Casting complex values to real discards the imaginary part
  return array(a, dtype, copy=False, order=order)

['other', 'tetra', 'tv', 'wfm']
[[ 1.00804127e-05  5.01915638e-04  4.12234440e-02  9.58264589e-01]]

<C:\ProgramData\Anaconda3> C:\rtl_sdr>
```



```
Administrator: C:\Windows\system32\cmd.exe
91.150 MHz - wfm 99.74%
91.600 MHz - wfm 99.64%
91.650 MHz - wfm 99.88%
92.150 MHz - wfm 80.99%
94.000 MHz - wfm 99.90%
94.450 MHz - tv 99.96%
94.500 MHz - wfm 99.60%
94.950 MHz - wfm 99.45%
95.000 MHz - wfm 100.00%
95.050 MHz - wfm 99.39%
95.450 MHz - wfm 100.00%
95.500 MHz - tv 85.49%
95.550 MHz - wfm 99.92%
95.850 MHz - wfm 91.33%
95.900 MHz - wfm 99.99%
95.950 MHz - wfm 97.89%
96.000 MHz - wfm 97.62%
96.050 MHz - wfm 99.98%
96.500 MHz - wfm 90.28%
96.550 MHz - wfm 98.05%
96.900 MHz - wfm 99.89%
96.950 MHz - wfm 99.45%
97.950 MHz - wfm 80.25%
100.450 MHz - wfm 85.64%
101.000 MHz - wfm 90.07%
```

This is a kind of proof-of-concept, but it really works

<https://github.com/randaller/cnn-rtlsdr>