Searching Pulsars Using Neural Networks

Training convolutional neural networks to detect very faint signals in very long time series

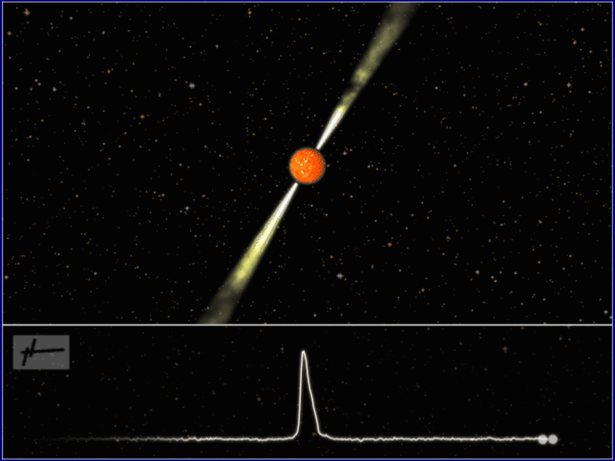
Lars Künkel - lars.kuenkel@uni-bielefeld.de

Universität Bielefeld

Big Data Science in Astroparticle Research

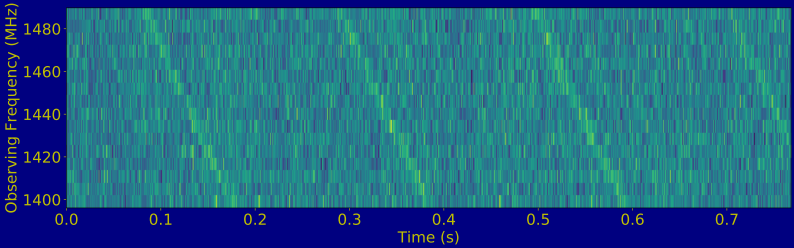
Outline

Rotating Neutron Stars



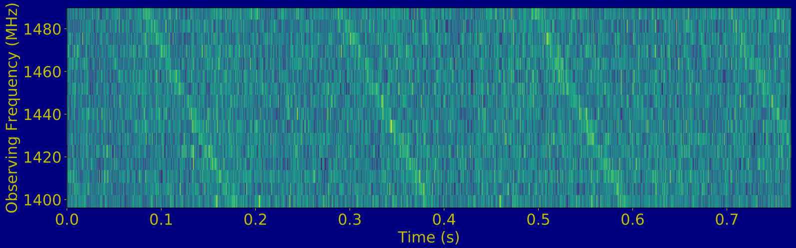
A pulsar and its signal. Source: Joeri van Leeuwen

Dispersed Pulses



A simulated pulsar embedded in noise.

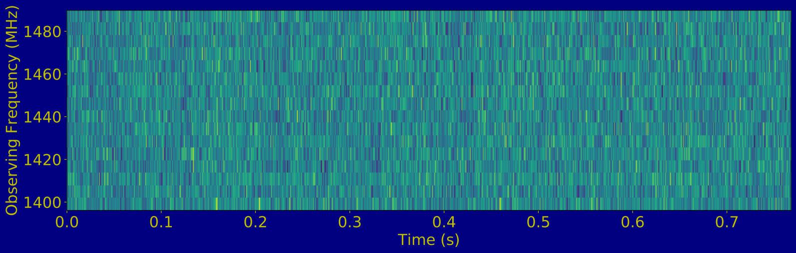
Dispersed Pulses



A simulated pulsar embedded in noise.

- Dispersion measure (DM) unknown prior to detection
- Most pulsars too weak to have individually detectable pulses
- Many interesting pulsars have a modulated period due to binary companions

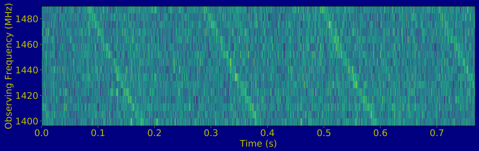
Dispersed Pulses

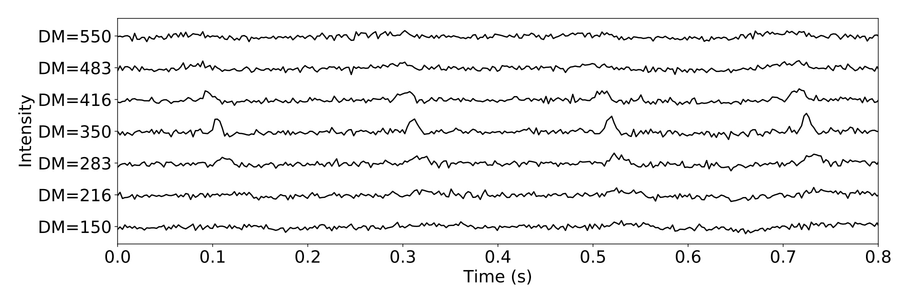


A simulated pulsar embedded in noise.

- Dispersion measure (DM) unknown prior to detection
- Most pulsars too weak to have individually detectable pulses
- Many interesting pulsars have a modulated period due to binary companions

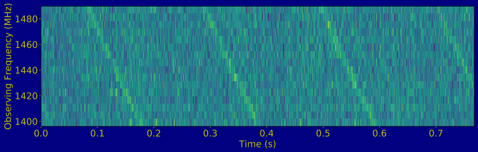
Brute Force Dedispersion

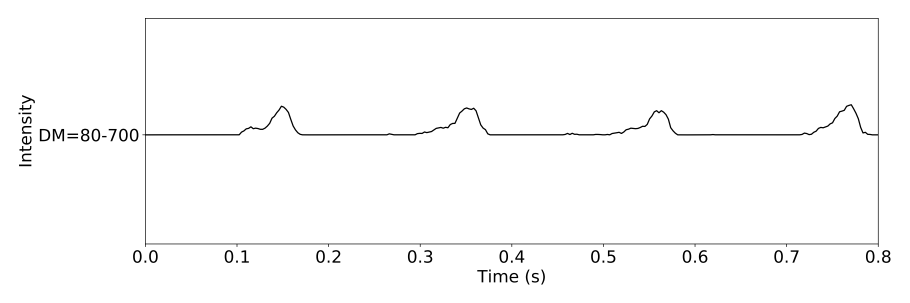




Normal search pipelines require brute force dedispersion.

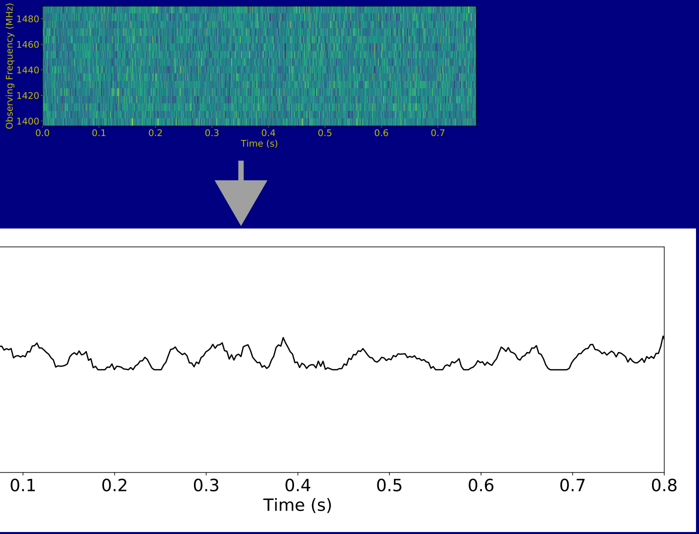
Neural Net Dedispersion I





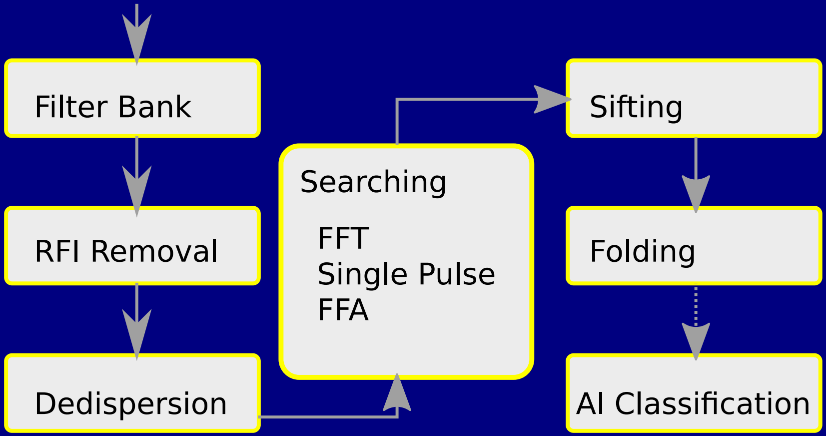
Neural Networks can create useful outputs for a range of DMs.

Neural Net Dedispersion II



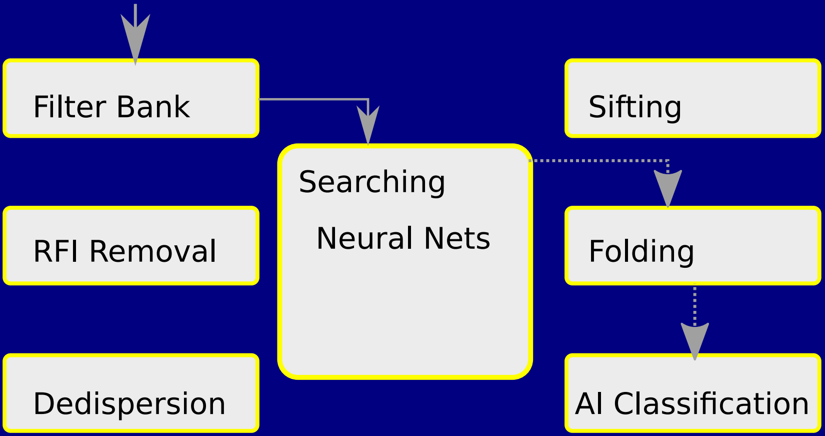
Weak pulsars are not immediately apparent in the dedispersed output.

Normal Search Pipeline



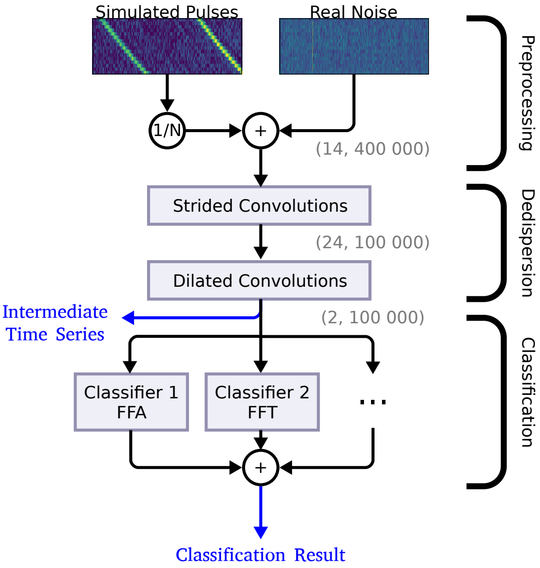
Structure of a normal pulsar search pipeline.

Neural Network Search Pipeline



Structure of a neural network based search pipeline.

Neural Network Architecture



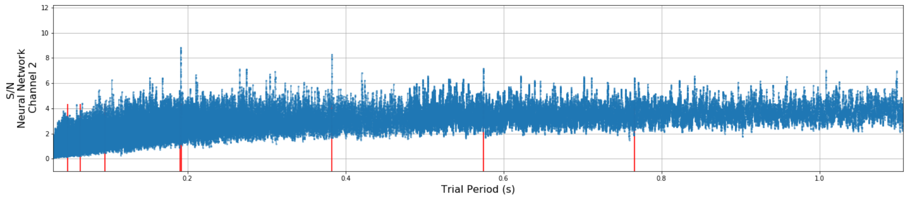
Basic Building Block

```
(3): TemporalBlock(
  (net): Sequential(
    (0): Conv1d(16, 16, kernel_size=(5,), stride=(1,),
        padding=(500,), dilation=(125,))
    (1): Chomp1d_acausal()
    (2): LeakyReLU(negative_slope=0.01)
    (3): GroupNorm(4, 16, eps=1e-05, affine=True)
    (4): Dropout(p=0.0)
    (5): Conv1d(16, 16, kernel_size=(5,), stride=(1,),
        padding=(500,), dilation=(125,))
    (6): Chomp1d_acausal()
    (7): LeakyReLU(negative_slope=0.01)
    (9): GroupNorm(4, 16, eps=1e-05, affine=True)
    (10): Dropout(p=0.0)
```

Training Procedure

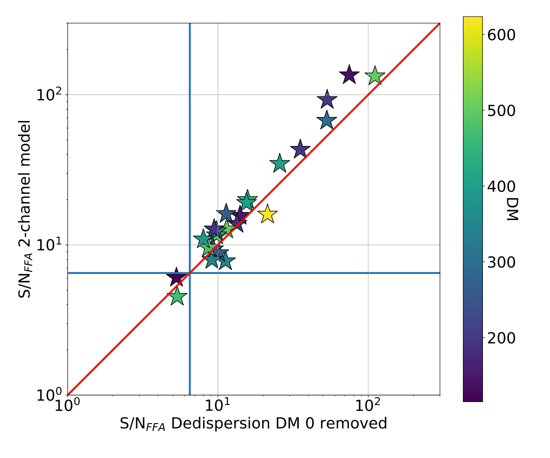
- Two loss functions: One for the intermediate, dedispersed output and one for the classification
- Trained using simulated pulsars and real noise
- Noise increases during training
- Three training steps:
  - After first step input length is increased
  - After second step computationally expensive classifier based on the fast folding algorithm is added

Fast Folding Algorithm



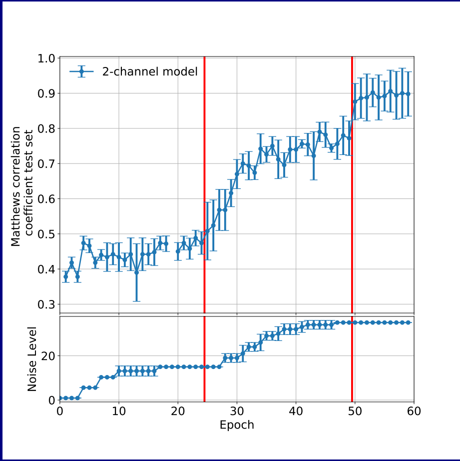
Result of the fast folding algorithm for the output of a neural net. A real pulsar is visible with a period of 0.19 s. The period and its harmonics are shown with vertical red lines.

Quality of Dedispersed Output



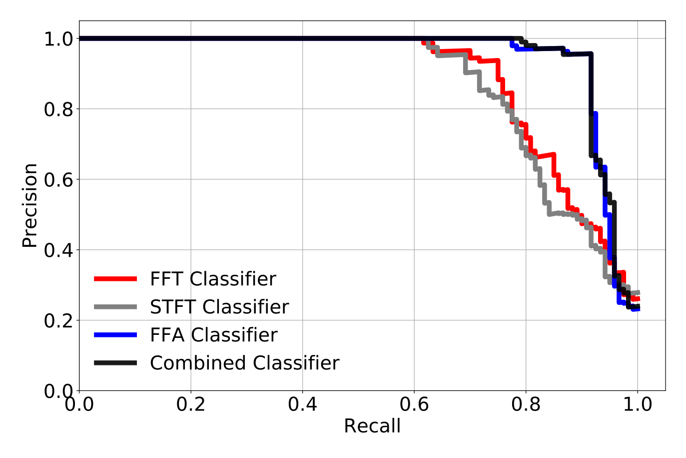
Dedispersing real pulsars using a neural network without knowing the DM creates a competitive output compared to dedispersing them at their correct DM.

Classification of Real Observations I



Performance of the classifiers on the test set during training. A high value indicates good performance. The vertical lines separate the three training steps.

Classification of Real Observations II



Performance of the classifiers on the test set. The final classification result is the combination of 3 classifiers.

Discarded Ideas I: RNNs

- Initial idea: Use Recurrent Neural Networks (RNN)
- Probably not ideal when used with 400 000 and more time steps
- Convolutional neural network easier to use as a feature extractor
- Might still be useful as a classifier in cases where the performance of the FFT and FFA performs worse than usual

Discarded Ideas II: Autoencoder Architecture

- Denoising Autoencoder: Noisy Filterbank –> Clean Filterbank
- Works well
- But dedispersed output more useful and requires less computations
- Having an additional objective other than classification proved useful

Discarded Ideas III: Purely Neural Network Based Classifiers

- Many ways to classify a time series
- Most methods work at low noise levels
- At higher noise levels FFA and FFT based methods outperform neural network based classifiers
- For accelerated signals neural network based classifiers might help

Conclusion

- We built a convolutional neural network that is able to detect pulsars
- Combining simulated pulsars with real noise allows us to easily increase the noise during training
- Classifiers which explicitly use the periodicity of the data outperform purely neural network based classifiers
- Training a FFT based classifier hugely increased our dedispersion performance
- Model is able to suppress radio frequency interference (RFI)