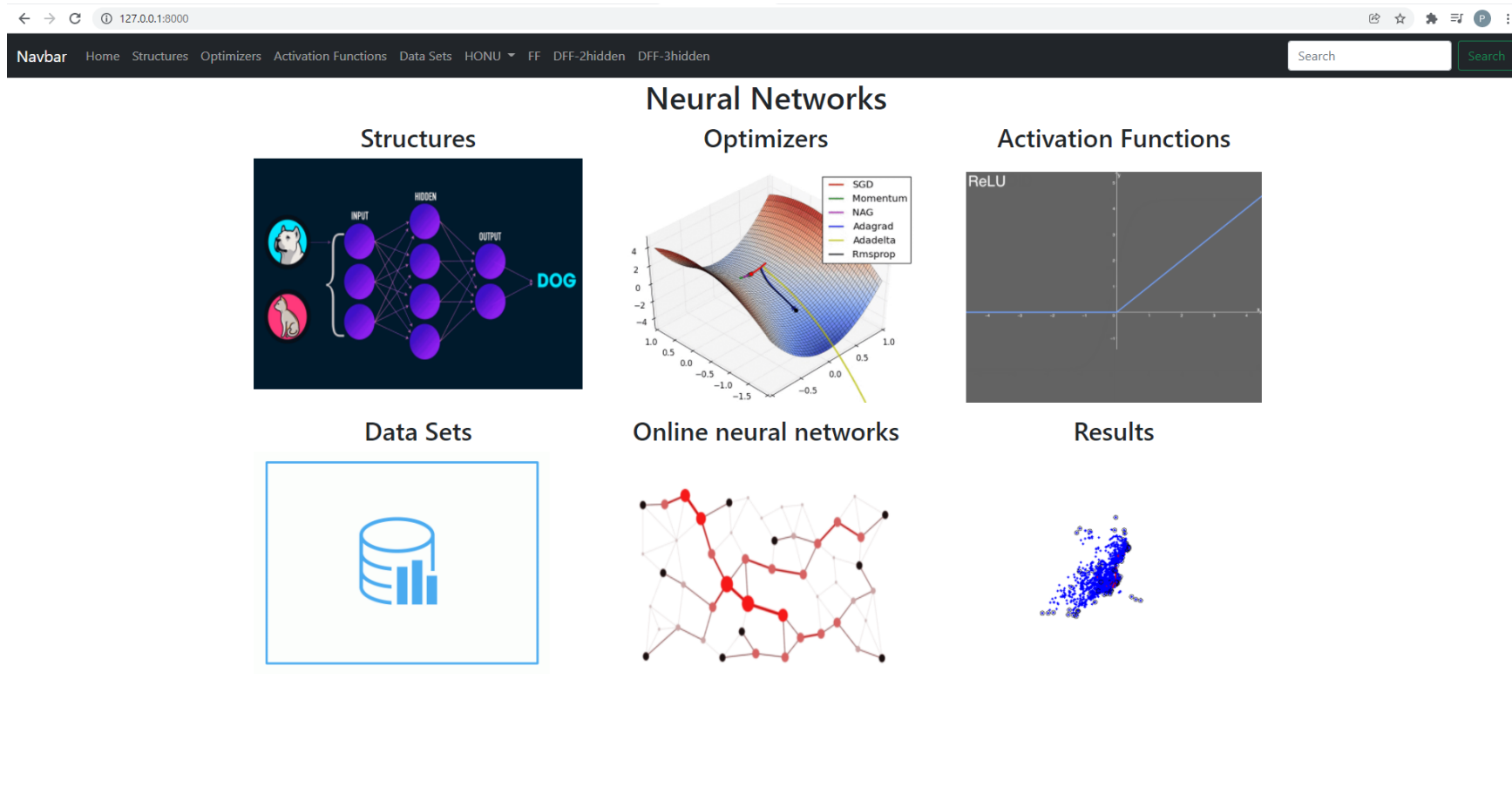


# Neural Networks

Patrik Zach

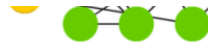
# Home Page



# Structures



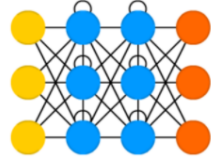
Jedná se o Feed Forward neuronovou síť s přidanou skrytou vrstvou, může jich být i více než 2.



Snížená složitost, nejedná se o plně propojenou neuronovou síť. Tvořena skrytými vrstvami, které jsou náhodně spojené. Účinnost velmi záleží na datech a úloze.

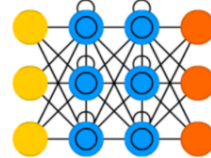
## Recurrent Neural Networks

### Recurrent Neural Network (RNN)



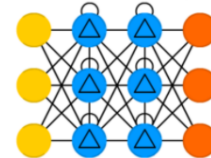
Používá se v případech, kdy je důležitý předchozí výstup, tzv. context. Předchozí výstup ovlivňuje následující výstup.

### Long / Short Term Memory (LSTM)



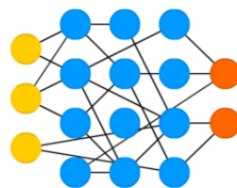
Obsahuje tzv. paměťovou buňku. Ta dokáže zpracovat data, které mají časové mezery. Síť dokáže udržet v paměti, co se již stalo. Jsou používány pro psaní a rozpoznání řeči.

### Gated Recurrent Unit (GRU)



Méně náročné než LSTM, ale téměř stejné účinné. Liší se pouze jiným typem paměťové buňky.

### Echo State Network (ESN)



Speciální tréninkový přístup.

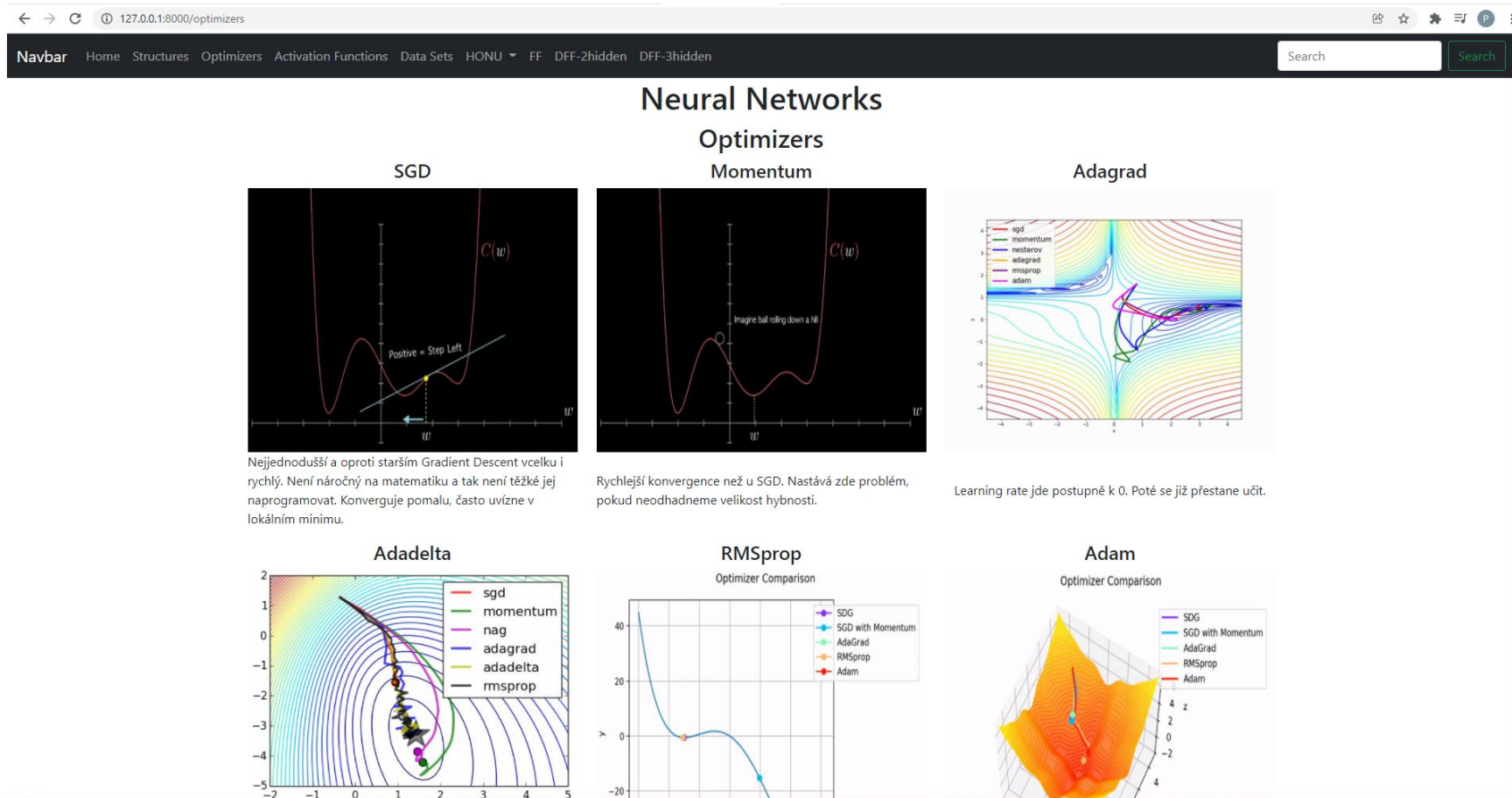
### Neural Turing Machine (NTM)



Extrahované paměťové buňky, síť může číst a zapisovat v paměti na základě aktuálního stavu.

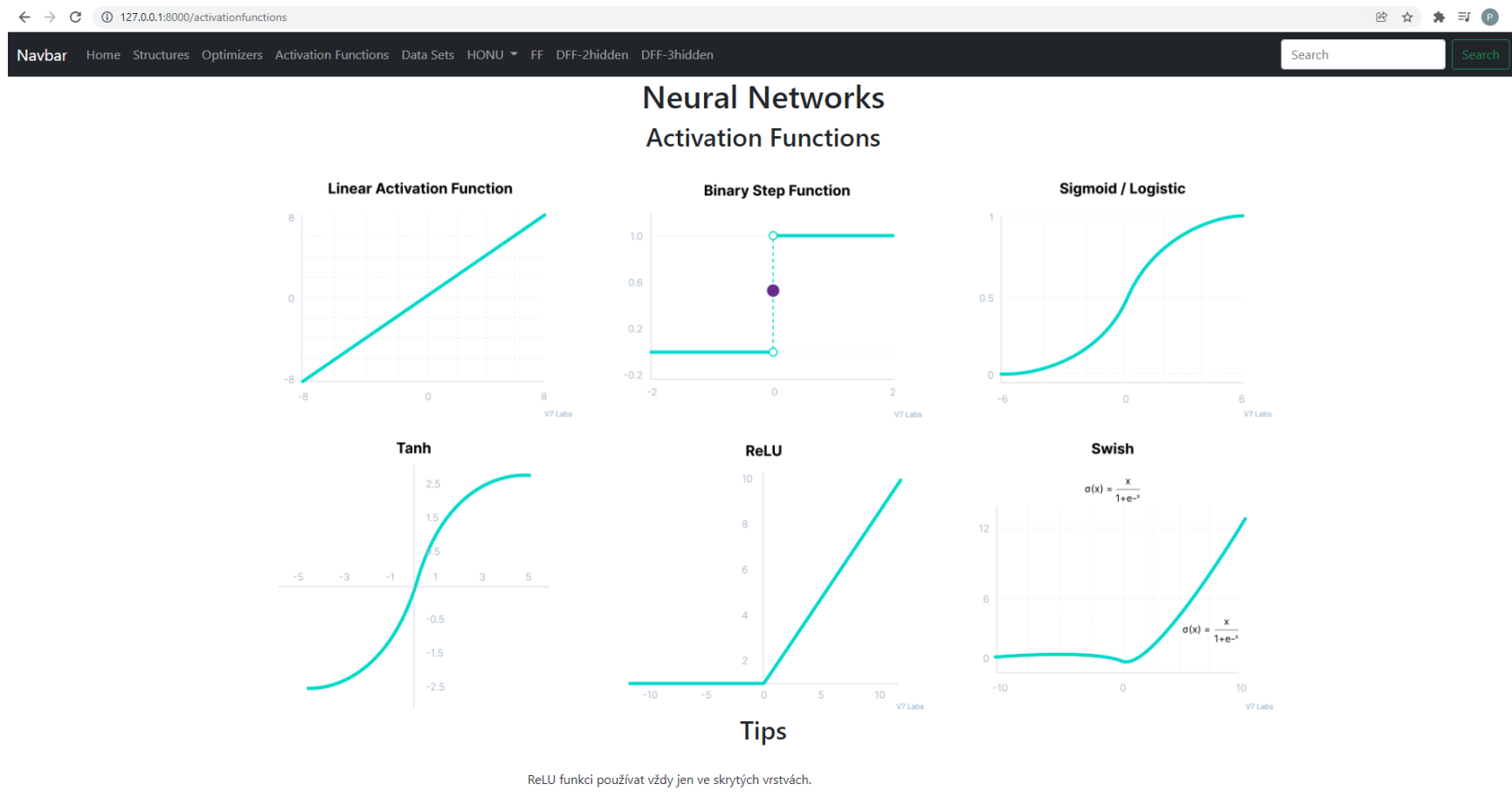
Popsány struktury neuronových sítí.

# Optimizers



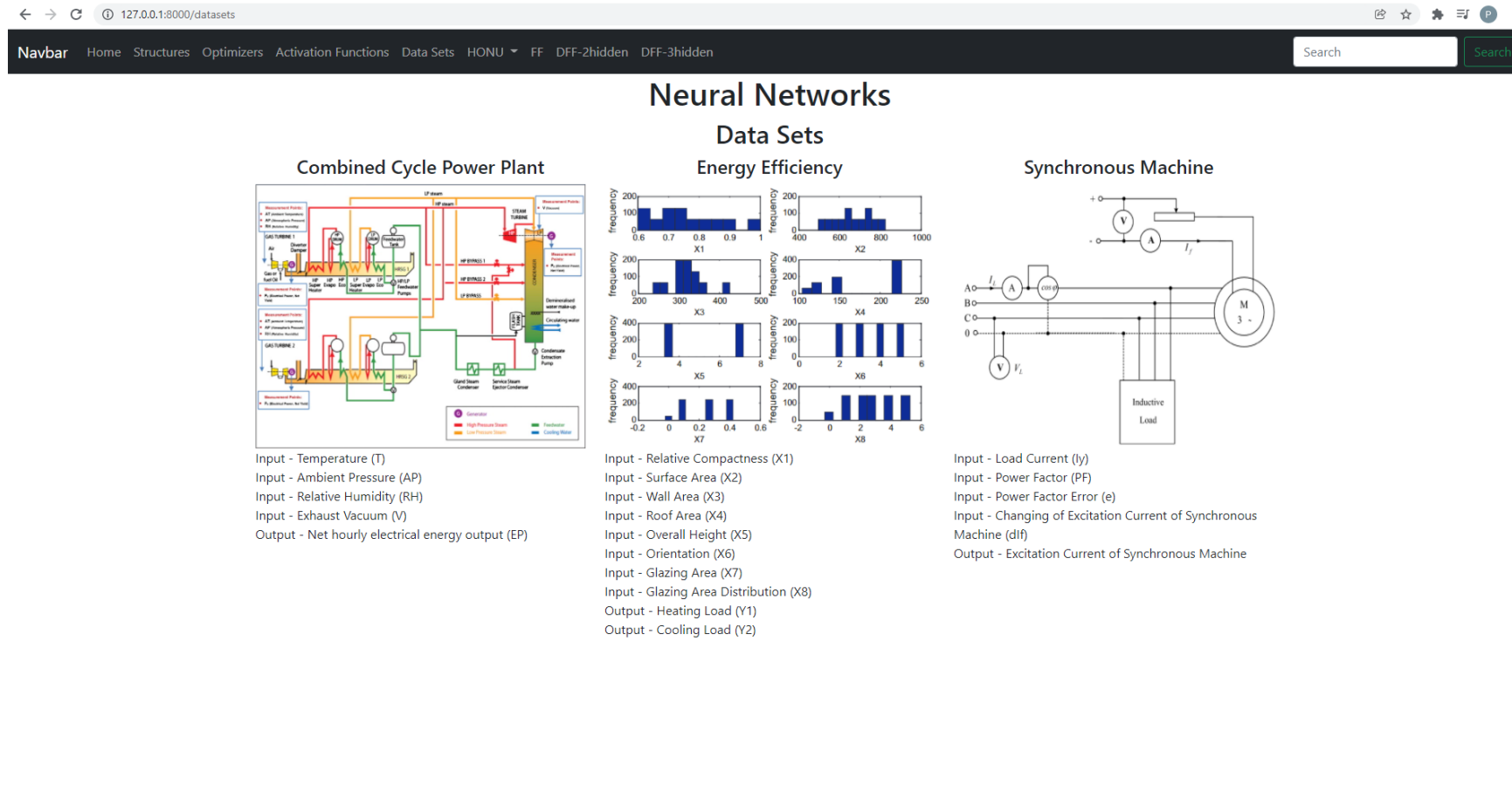
Popsány druhy optimalizátorů + vývoj

# Activation Functions



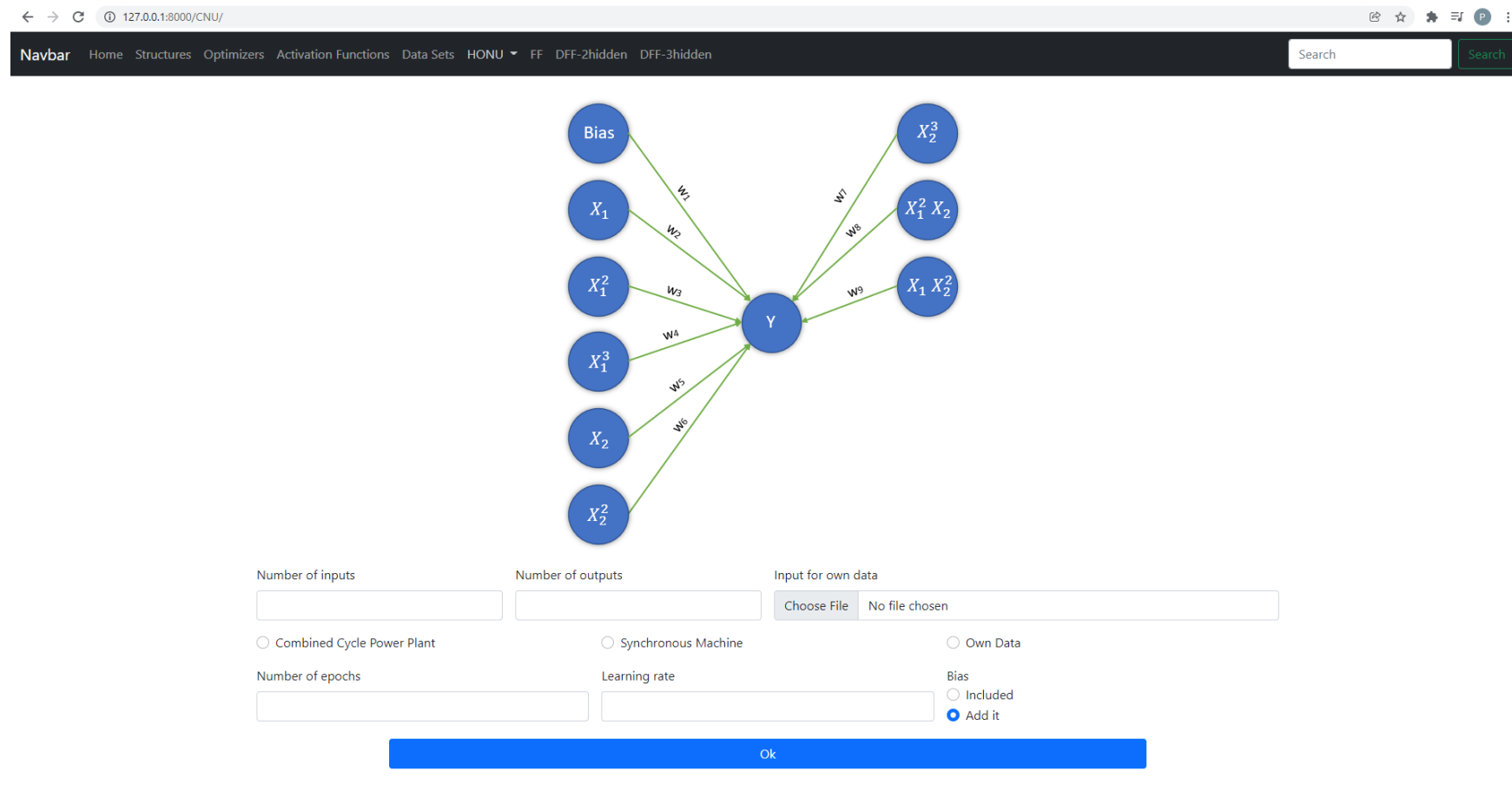
Základní aktivační funkce

# Datasets



3 základní datasety, které se dají použít, pokus nemáme vlastní

# HONU – LNU, QNU, CNU



Zobrazen CNU, Typ sítě - Perceptron

# FF

127.0.0.1:8000/FF/

$x_1$   $x_2$   $W_1$   $W_2$   $y$

Inputs Hidden layer Output layer

Number of inputs:

Number of outputs:

Input for own data:  No file chosen

☐ Combined Cycle Power Plant ☐ Energy Efficiency ☐ Synchronous Machine ☐ Own Data

HiddenLayer1 - number of nodes:

HiddenLayer1 - activation function:

OutputLayer - activation function:

Optimizer:

Number of epochs:

Learning rate:

Batch Size:

Test Data:

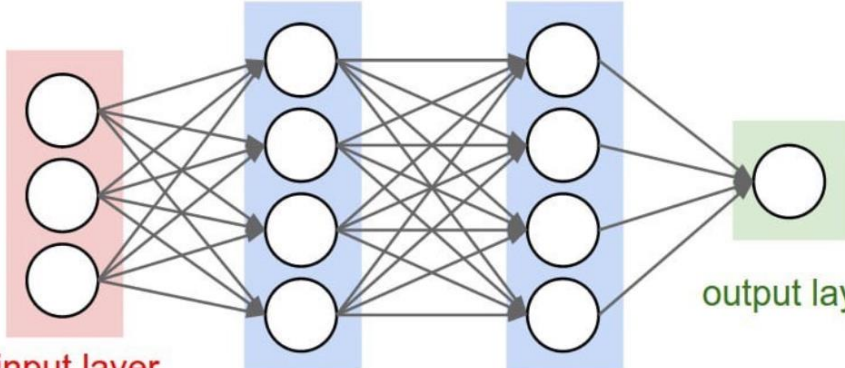
Shuffle:

Bias: ☐ Included ☒ Add it



# DFF – 2 hidden layers

127.0.0.1:8000/DFF2hiddenlayers/



input layer

hidden layer 1

hidden layer 2

output layer

Number of inputs

Number of outputs

Input for own data

Choose File No file chosen

☐ Combined Cycle Power Plant ☐ Energy Efficiency ☐ Synchronous Machine ☐ Own Data

HiddenLayer1 - number of nodes

HiddenLayer2 - number of nodes

HiddenLayer1 - activation function

HiddenLayer2 - activation function

Choose activation function Choose activation function

OutputLayer - activation function

Choose activation function

Optimizer

Adam

Number of epochs

Learning rate

Batch Size

Test Data

Shuffle

False

Bias

☐ Included ☒ Add it

Ok

# DFF – 3 hidden layers

← → ↻ 127.0.0.1:8000/DFF3hiddenlayers/

Visible input layer

Hidden layers

Visible output layer

$w_{11}^{(1)}$   $w_{11}^{(2)}$   $w_{11}^{(3)}$   $w_{11}^{(4)}$

Number of inputs

Number of outputs

Input for own data  
 No file chosen

☐ Combined Cycle Power Plant ☐ Energy Efficiency ☐ Synchronous Machine ☐ Own Data

HiddenLayer1 - number of nodes

HiddenLayer2 - number of nodes

HiddenLayer3 - number of nodes

HiddenLayer1 - activation function  
 ▾

HiddenLayer2 - activation function  
 ▾

HiddenLayer3 - activation function  
 ▾

OutputLayer - activation function  
 ▾

Optimizer  
 ▾

Number of epochs

Learning rate

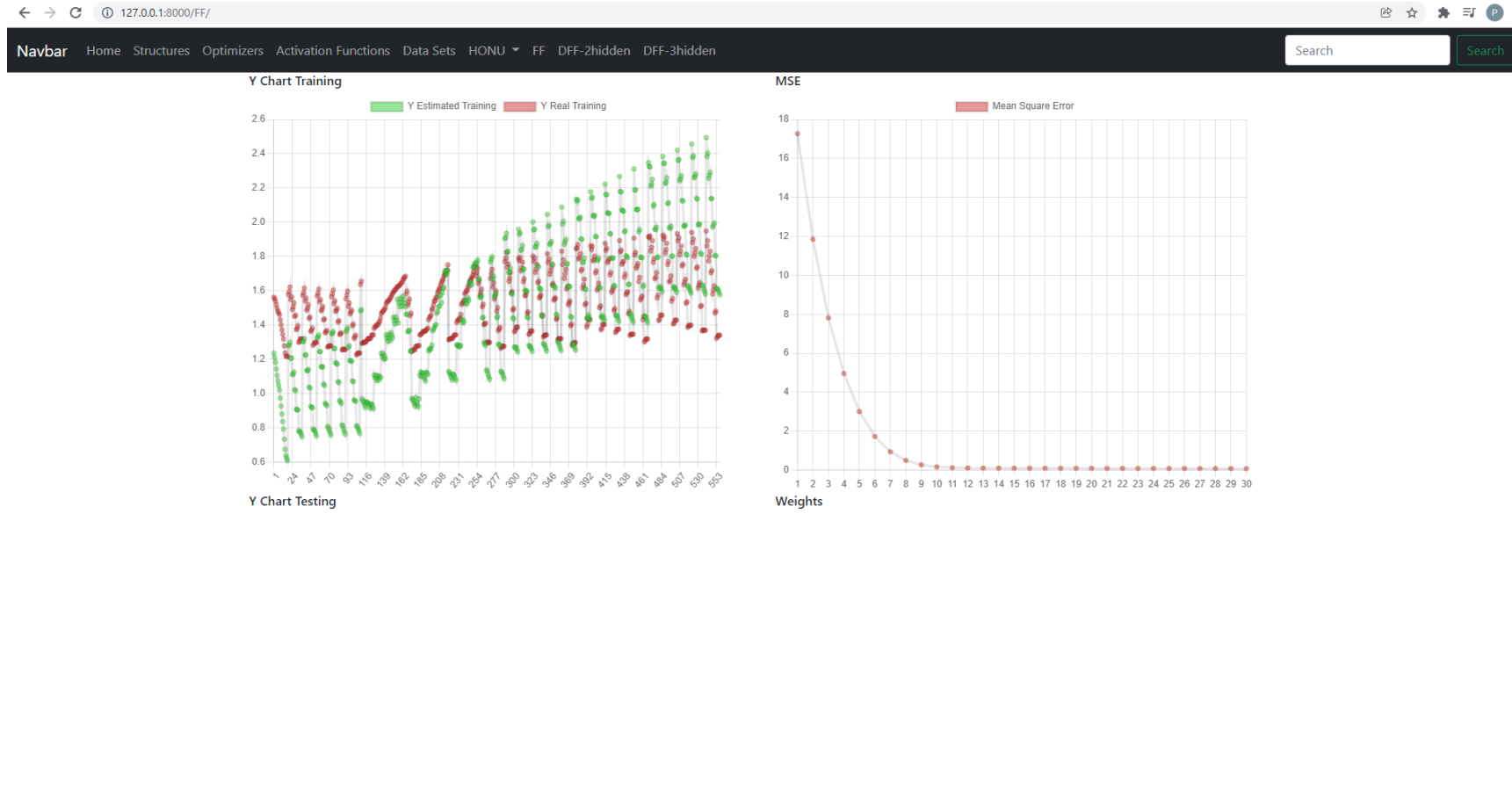
Batch Size

Test Data

Shuffle  
 ▾

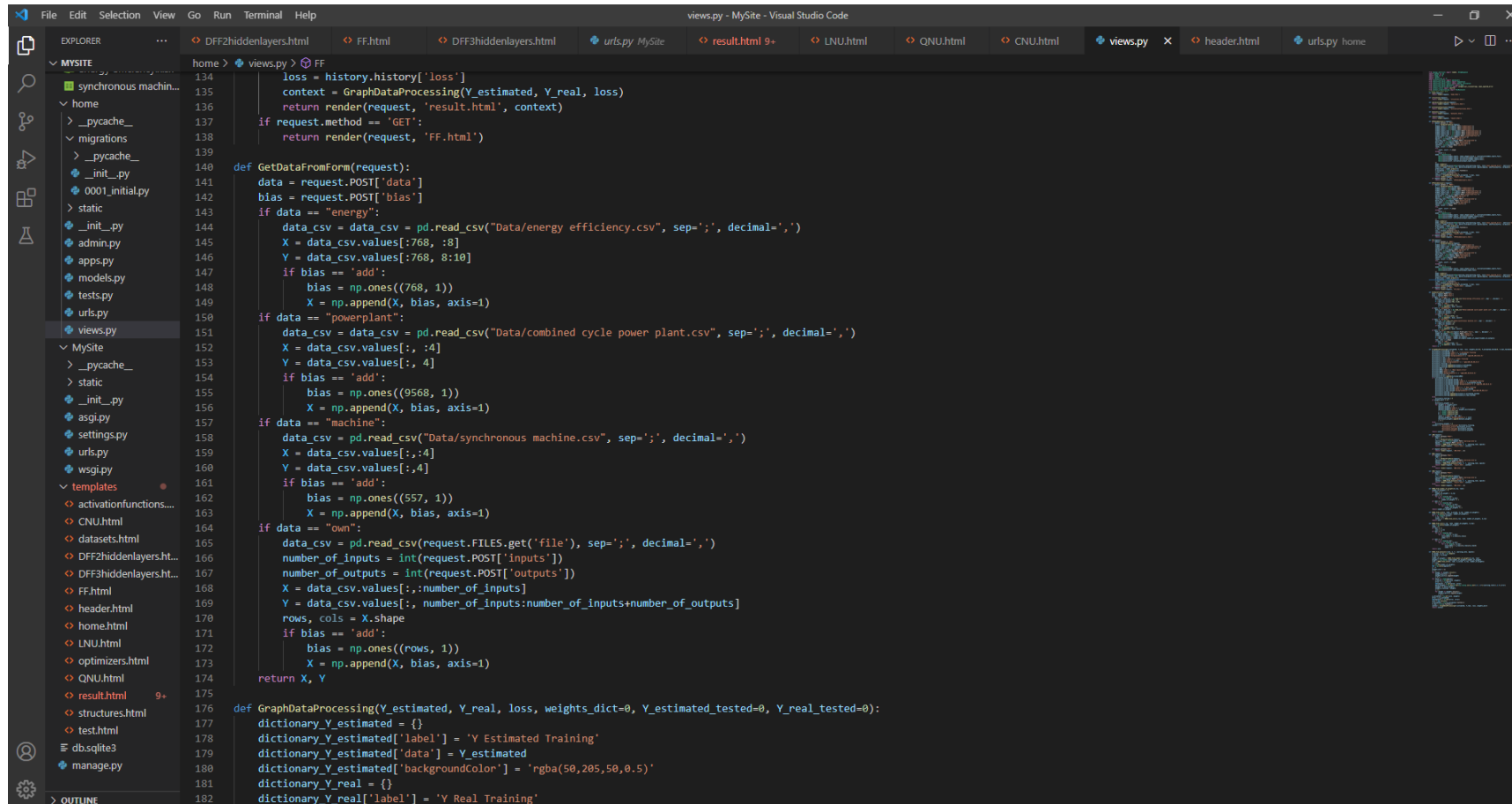
Bias  
☐ Included ☒ Add it

# Results



Po odeslání vyplněného formuláře se objeví grafy z výsledky.

# Code Structure



The screenshot shows the Visual Studio Code interface with a project named 'MySite'. The Explorer sidebar on the left displays the file structure, including folders like 'home' and 'templates', and files like 'views.py'. The main editor area shows the content of 'views.py', which includes a Flask application with a 'GET' route and a 'GetDataFromForm' function. The code uses pandas to read CSV files and numpy to process data. The 'GraphDataProcessing' function is also visible at the bottom of the file.

```
134     loss = history.history['loss']
135     context = GraphDataProcessing(Y_estimated, Y_real, loss)
136     return render(request, 'result.html', context)
137 if request.method == 'GET':
138     return render(request, 'FF.html')
139
140 def GetDataFromForm(request):
141     data = request.POST['data']
142     bias = request.POST['bias']
143     if data == "energy":
144         data_csv = pd.read_csv("Data/energy efficiency.csv", sep=';', decimal=',')
145         X = data_csv.values[:, 768, :8]
146         Y = data_csv.values[:, 768, 8:10]
147         if bias == 'add':
148             bias = np.ones((768, 1))
149             X = np.append(X, bias, axis=1)
150     if data == "powerplant":
151         data_csv = pd.read_csv("Data/combined cycle power plant.csv", sep=';', decimal=',')
152         X = data_csv.values[:, :4]
153         Y = data_csv.values[:, 4]
154         if bias == 'add':
155             bias = np.ones((9568, 1))
156             X = np.append(X, bias, axis=1)
157     if data == "machine":
158         data_csv = pd.read_csv("Data/synchronous machine.csv", sep=';', decimal=',')
159         X = data_csv.values[:, :4]
160         Y = data_csv.values[:, 4]
161         if bias == 'add':
162             bias = np.ones((557, 1))
163             X = np.append(X, bias, axis=1)
164     if data == "own":
165         data_csv = pd.read_csv(request.FILES.get('file'), sep=';', decimal=',')
166         number_of_inputs = int(request.POST['inputs'])
167         number_of_outputs = int(request.POST['outputs'])
168         X = data_csv.values[:, :number_of_inputs]
169         Y = data_csv.values[:, number_of_inputs:number_of_inputs+number_of_outputs]
170         rows, cols = X.shape
171         if bias == 'add':
172             bias = np.ones((rows, 1))
173             X = np.append(X, bias, axis=1)
174     return X, Y
175
176 def GraphDataProcessing(Y_estimated, Y_real, loss, weights_dict=0, Y_estimated_tested=0, Y_real_tested=0):
177     dictionary_Y_estimated = {}
178     dictionary_Y_estimated['label'] = 'Y Estimated Training'
179     dictionary_Y_estimated['data'] = Y_estimated
180     dictionary_Y_estimated['backgroundColon'] = 'rgba(50,205,50,0.5)'
181     dictionary_Y_real = {}
182     dictionary_Y_real['label'] = 'Y Real Training'
```

# Objectives

- Upravit Results tak, aby bylo možné vybrat např. pouze 100 výsledků ze vzorku.
- Neuronové sítě naprogramovat více bez použití knihoven.
- Přidat další druhy neuronových sítí.
- Upravit čekání na výsledky, možná přidat časový odhad
- Přidat více Optimizerů – momentálně funguje jen Adam (což je ale jeden z nejefektivnějších optimizerů)
- Lépe vizualizovat Results.