

ИПЛИТ РАН; ГБОУ Школа № 924

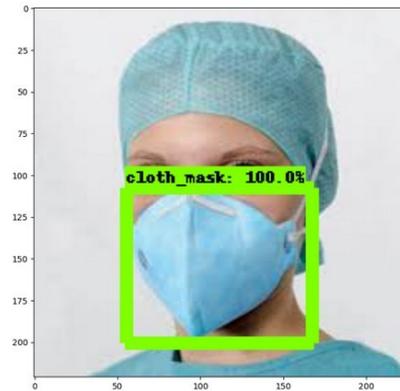
Развертывание нейросети на базе ОС "Альт" для обнаружения онкологических заболеваний

Воронин И.В., Дарсавелидзе А.А.
woronin05@yandex.ru

СПО от обучения до разработки. OSEUCONF-2022, Переславь-Залесский 2022 г.

Нейросети вокруг нас

<https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/quickstart/beginner.ipynb>



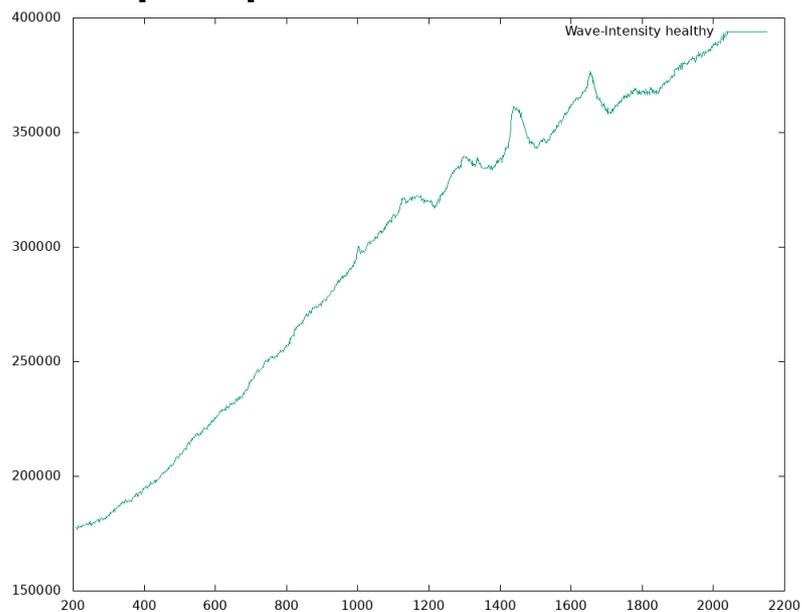
Диагностика заболеваний

- Традиционные: компьютерная томография (КТ), магнитно-резонансная томография (МРТ), позитронно-эмиссионная томография (ПЭТ) и однофотонная эмиссионная компьютерная томография (ОФЭКТ)
- С использованием нейросетей, дает преимущество: в скорости, в стоимости, в доступности

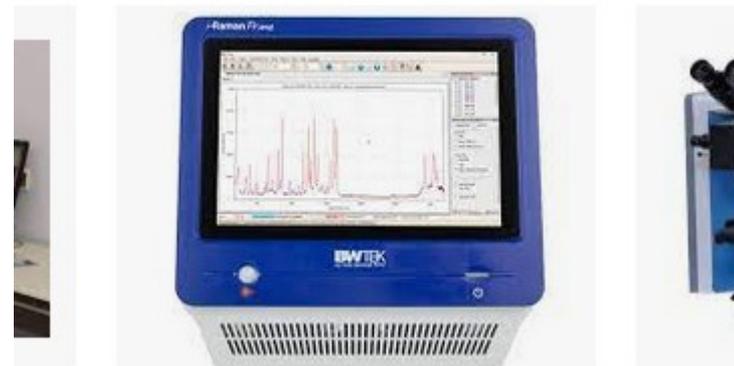


Набор данных

- Рамановский спектрометр
- Комбинационное рассеяние
- Спектры разных тканей



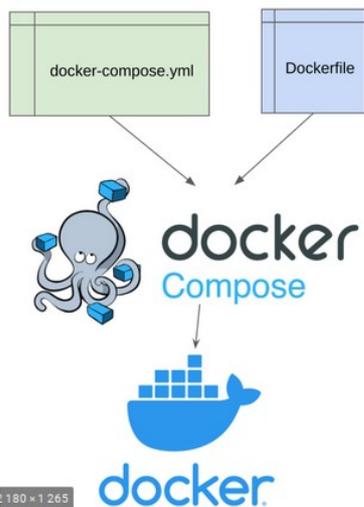
Рамановский спектрометр LabRAM HR Evolution - Найте...
nytek.ru



Шаги создания сети

На ОС Альт ставится Docker
и Docker-compose

Заливается среда
разработки : Jupyter
Notebook



jupyter

Files Running Clusters

Select items to perform actions on them.

- 0 ▾ /
- / 1
- / books
- / data_input
- / data_input2
- / data_input4
- / data_input5
- / pict1
- / newTestBook.ipynb
- / tf_task_dataprep.ipynb

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import os
from keras.models import Sequential
from keras.layers import Flatten, Dense, Lambda
from keras.layers import Convolution2D
from keras.layers.pooling import MaxPooling2D
from keras.layers import Cropping2D
from keras.layers.core import Dense, Dropout, Activation
from keras.layers import Dense, Conv1D, Flatten, MaxPooling1D
from tensorflow.keras import regularizers
from keras.layers import BatchNormalization
from keras.utils import np_utils
from tensorflow.keras.utils import plot_model
import graphviz, pydot, pydotplus
```

Выполнение обучения сети

Используем Keras модель с TensorFlow
NumPy

Загружаются в питон данные из ОС
`measurements.append(df[['Intensity']].to_numpy())`

```
if str(sample) == 'abouttumoral':
```

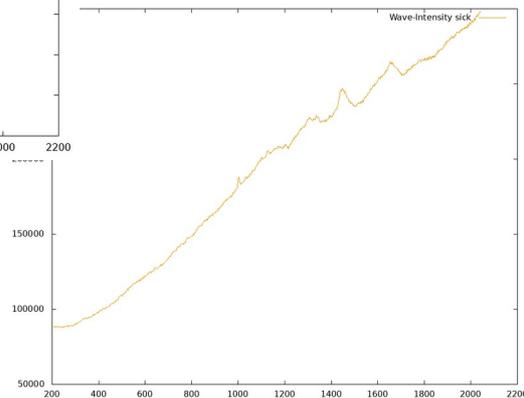
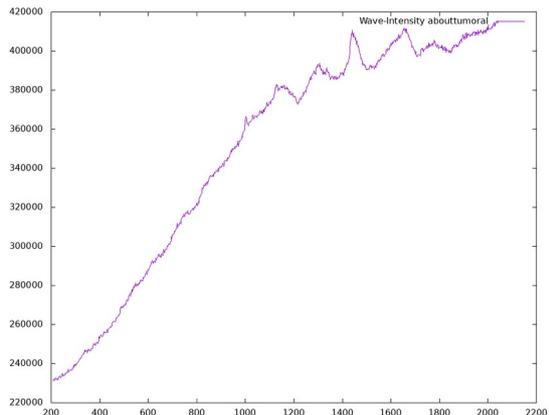
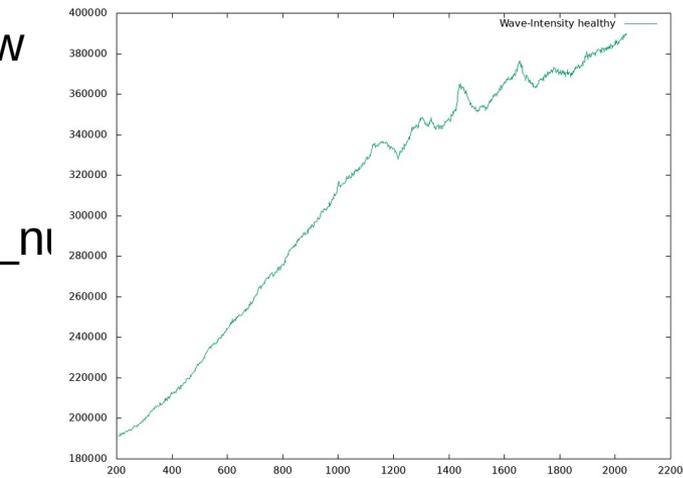
```
    classifications.append([1,0,0])
```

```
if str(sample) == 'healthy':
```

```
    classifications.append([0,1,0])
```

```
if str(sample) == 'sick':
```

```
    classifications.append([0,0,1])
```



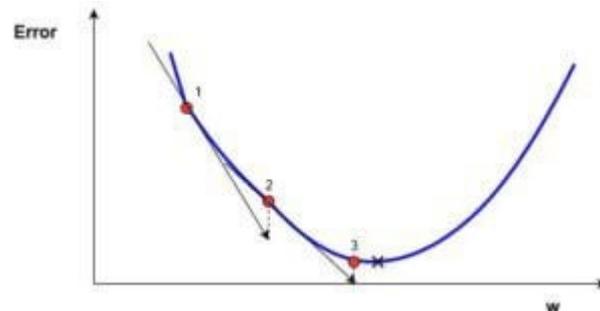
Выбор тестов и алгоритма

Выделение тестовой выборки из представленных датасетов 85 на 15 %

```
Ввод [7]: xtrain, xtest, ytrain, ytest=train_test_split(X_train, Y_train, test_size=0.15)
print(f'Total amount of train measurements: {xtrain.shape}')
print(f'Total amount of train labels: {ytrain.shape}')
print(f'Total amount of test measurements: {xtest.shape}')
print(f'Total amount of test labels: {ytest.shape}')
```

```
Total amount of train measurements: (482, 1015)
Total amount of train labels: (482, 3)
Total amount of test measurements: (86, 1015)
Total amount of test labels: (86, 3)
```

Стохастический градиентный спуск — самый используемый алгоритм обучения



Многослойная модель сети

- Слой на входе 1014
- Слой на выходе 3
- Промежуточные слои
- Количество эпох, размер пакета
- Количество параметров, которые участвуют в настройке сети

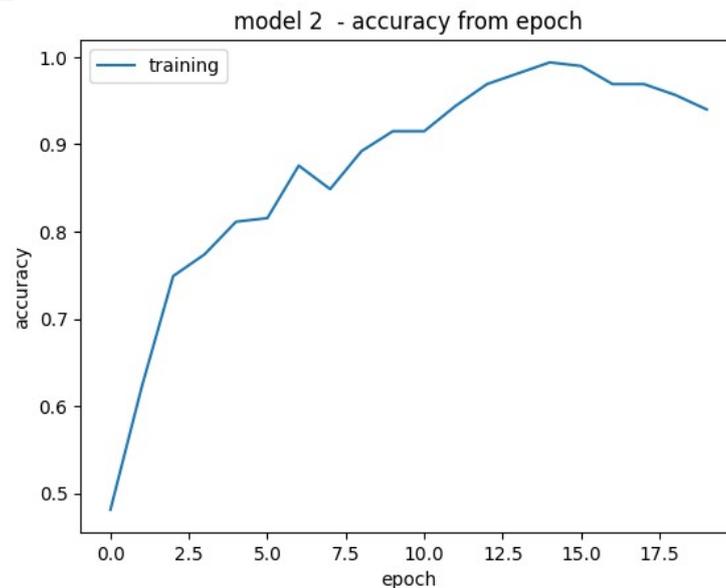
```
Ввод [8]: # Model 1, updated some layers
model1 = Sequential()
model1.add(Conv1D(512, 2, activation='relu', input_shape=(1015,1)))
model1.add(Dense(256, activation = 'relu'))
model1.add(MaxPooling1D())
model1.add(Flatten())
model1.add(Dense(128, activation = 'relu'))
model1.add(Dropout(0.2))
model1.add(Dense(3, activation = 'softmax'))
model1.summary()
model1.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model1.fit(xtrain, ytrain ,batch_size=128, epochs=64, verbose=1)
```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 1014, 512)	1536
dense (Dense)	(None, 1014, 256)	131328
max_pooling1d (MaxPooling1D)	(None, 507, 256)	0
flatten (Flatten)	(None, 129792)	0
dense_1 (Dense)	(None, 128)	16613504
dropout (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 3)	387

Обучение сети

```
Ввод [9]: # Model 2, updated some layers
# This model is showing better results
model2 = Sequential()
model2.add(Conv1D(64, 2, activation='relu', input_shape=(1015,1)))
model2.add(Conv1D(128, 2, activation='relu'))
model2.add(Conv1D(256, 2, activation='relu'))
model2.add(Conv1D(512, 2, activation='relu'))
model2.add(MaxPooling1D())
model2.add(Flatten())
model2.add(Dense(256, activation = 'relu'))
model2.add(Dense(128, activation = 'relu'))
model2.add(Dense(3, activation = 'softmax'))
model2.summary()
model2.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
model2.fit(xtrain, ytrain, batch_size=128, epochs=64, verbose=1)
Epoch 56/64
4/4 [=====] - 8s 2s/step - loss: 0.0377 - accuracy: 0.9938
Epoch 57/64
4/4 [=====] - 8s 2s/step - loss: 0.0716 - accuracy: 0.9772
Epoch 58/64
4/4 [=====] - 8s 2s/step - loss: 0.0512 - accuracy: 0.9938
Epoch 59/64
4/4 [=====] - 8s 2s/step - loss: 0.0341 - accuracy: 0.9876
Epoch 60/64
4/4 [=====] - 8s 2s/step - loss: 0.0239 - accuracy: 0.9938
Epoch 61/64
4/4 [=====] - 8s 2s/step - loss: 0.0157 - accuracy: 0.9979
Epoch 62/64
4/4 [=====] - 8s 2s/step - loss: 0.0144 - accuracy: 0.9979
Epoch 63/64
```

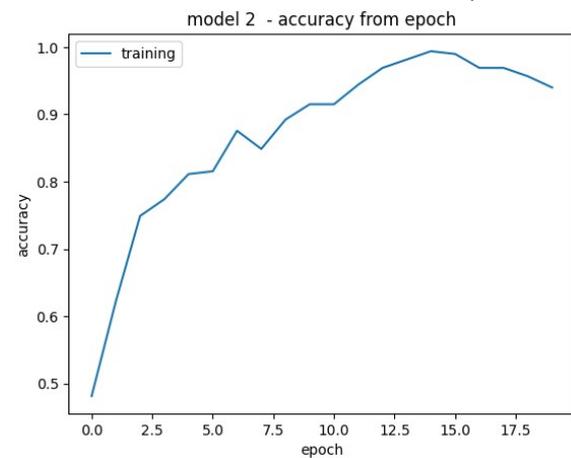
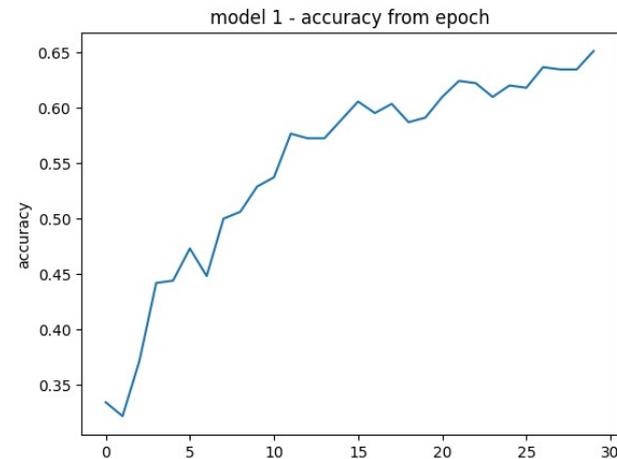
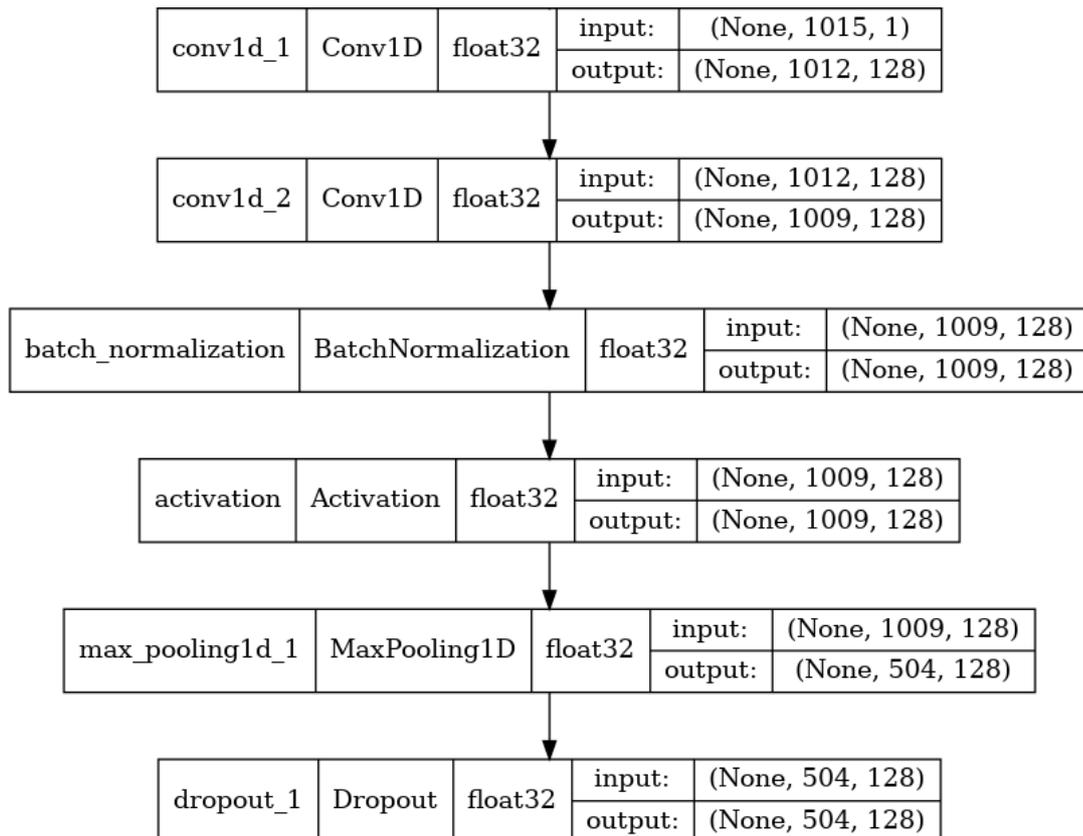


Тренировка на результат

```
Ввод [11]: acc = model2.evaluate(X_train, Y_train)
print("Loss:", acc[0], " Accuracy:", acc[1])
pred = model2.predict(xtest)
print(np.round(pred,2))

18/18 [=====] - 3s 136ms/step - loss: 0.0489 - accuracy: 0.9842
Loss: 0.04887872189283371 Accuracy: 0.9841549396514893
[[0.  0.2  0.8 ]
 [0.5  0.35 0.16]
 [0.39 0.43 0.18]
 [0.03 0.76 0.21]
 [0.  0.03 0.97]
 [0.  0.05 0.95]
 [0.  0.47 0.53]
 [0.  0.  1.  ]
 [0.  0.21 0.79]
 [0.47 0.36 0.17]
 [0.47 0.36 0.17]
 [0.  0.65 0.35]
 [0.55 0.26 0.19]
 [0.03 0.82 0.16]
 [0.  0.9  0.1 ]
 [0.  0.  1.  ]
 [0.01 0.63 0.35]
 [0.  0.39 0.61]
 [0.42 0.39 0.19]
 [0.  0.  1.  ]
 [0.  0.14 0.86]
 [0.02 0.64 0.34]
 [0.47 0.36 0.17]
 [0.  0.2  0.8 ]
 [0.  0.26 0.74]
 [0.  0.64 0.36]
 [0.  0.36 0.64]
 [0.  0.15 0.85]
 [0.  0.65 0.35]
 [0.47 0.36 0.17]
 [0.  0.  1.  ]
 [1.  0.  0.  ]
 [0.47 0.36 0.17]
 [0.  0.  1.  ]
```

Анализ архитектуры и точность



ИТОГИ И ВЫВОДЫ

- Задача решается в среде ОС ALT p10
- Аппаратные средства: могут быть использованы ARM процессоры
- Orange Pi
- Образовательный проект УМКИ



Е. С. Глазман, О. А. Кожина, Ю. Л. Хотунцев, Е. Н. Кудачова
ТЕХНОЛОГИЯ



7

ДРОФД



ВЕРТИКАЛЬ

Е. С. Глазман, О. А. Кожина, Ю. Л. Хотунцев, Е. Н. Кудачова
ТЕХНОЛОГИЯ



8-9

ДРОФД



ВЕРТИКАЛЬ