Newcastle University

LEEDS BECKETT UNIVERSITY

KTH VETENSKAP OCH KONST

UNIVERSIDADE D COIMBRA

ЧНУ

1920

ХАІ

ЧДУ

НТУ

Vivere est cogitare

IT Альянс

SmartMe UNIVERSITY

Internet of Things
for Industry and Human Applications

Data Science for Internet of Things
and Internet of Everything

ALIoT

Internet of Things
for Industry and Human Applications

Data Science for Intenet of Things and Internet of Everything

PRACTICUM

Funded by the
Erasmus+ Programme
of the European Union

Ministry of Education and Science of Ukraine
Volodymyr Dahl East Ukrainian National University
National Aerospace University “Kharkiv Aviation Institute”
Ternopil National Economic University

I. S. Skarga-Bandurova, A. V. Gorbenko, T. O. Biloborodova,
V. S. Koval, A.O. Sachenko, O. M. Tarasyuk

Internet of Things for Industry and Human Applications

Data Science for Internet of Things and Internet of Everything

Practicum

Edited by I.S. Skarga-Bandurova and A.V. Gorbenko

Project
ERASMUS+ ALIOT “Internet of Things: Emerging Curriculum
for Industry and Human Applications”
(573818-EPP-1-2016-1-UK-EPPKA2-CBHE-JP)

2019

UDC 004.415/.416.04(076.5)=111
H34

Reviewers:
Dr. Ah-Lian Kor, Leeds Beckett University, UK
Prof. DrS. Gennady Krivoulya, Kharkiv National University of Radioelectonics, Ukraine

H34 Skarga-Bandurova I.S., Gorbenko A.V., Biloborodova T.O., Koval V.S., Sachenko A.O., Tarasyuk O. M. **Data Science for Internet of Things and Internet of Everything**: Practicum / Skarga-Bandurova I.S. and Gorbenko A.V. (Eds.) – Ministry of Education and Science of Ukraine, Volodymyr Dahl East Ukrainian National University, National Aerospace University "Kharkiv Aviation Institute", Ternopil National Economic University, 2019. – 169 p.

ISBN 978-617-7361-86-1

The materials of the practical part of the study course "MC2. Data Science for IoT and IoE", developed in the framework of the ERASMUS+ ALIOT project "Internet of Things: Emerging Curriculum for Industry and Human Applications" (573818-EPP-1-2016-1-UK-EPPKA2-CBHE-JP).

The structure of work on verification of residual knowledge in the discipline, the corresponding practical material, examples of tasks and criteria of evaluation are given. In the learning process, the theoretical aspects of development and implementation of IoT-based systems are presented. The basic concepts and approaches of data science for IoT systems are given.

It is intended for engineers, developers and scientists engaged in the development and implementation of of IoT-based systems, for postgraduate students of universities studying in areas of IoT, computer science, computer and software engineering, as well as for teachers of relevant courses.

Ref. – 76 items, figures – 60, tables – 7.

Approved by Academic Council of National Aerospace University "Kharkiv Aviation Institute" (record No 4, December 19, 2018).

ISBN 978-617-7361-86-1

Міністерство освіти і науки України
Східноукраїнський національний університет ім. Володимира Даля
Національний аерокосмічний університет
ім. М. Є. Жуковського «Харківський Авіаційний Інститут”
Тернопільський національний економічний університет

Скарга-Бандурова І.С., Горбенко А.В., Білобородова Т.О.,
Коваль В.С., Саченко А.О., Тарасюк О.М.

Інтернет речей для індустріальних і гуманітарних застосунків

Наука про дані для Інтернету Речей та Інтернету Всього

Практикум

Редактори Скарга-Бандурова І.С., Горбенко А.В.

Проект ERASMUS+ ALIOT
“Інтернет речей: нова освітня програма
для потреб промисловості та суспільства”
(573818-EPP-1-2016-1-UK-EPPKA2-CBHE-JP)

2019

УДК 004.415/.416.04(076.5)=111
Н34

Рецензенти: Др. А-Ліан Кор, Leeds Beckett University, Велика Британія

Д.т.н., проф. Геннадій Кривуля, Харківський національний університет радіоелектроніки

Скарга-Бандурова І.С., Горбенко А.В., Білобородова Т.О., Коваль В.С., Саченко А.О., Тарасюк О.М.

Н34 Наука про дані для Інтернету Речей та Інтернету Всього / За ред. І.С. Скарга-Бандурової та А.В. – МОН України, Східноукраїнський університет ім. Володимира Даля, Національний аерокосмічний університет ім. М. Є. Жуковського «ХАІ». – 169с.

ISBN 978-617-7361-86-1

Викладено матеріали практичної частини курсу “MC2. Data Science for IoT and IoE”, підготовленого в рамках проекту ERASMUS+ ALIOT “Internet of Things: Emerging Curriculum for Industry and Human Applications” (573818-EPP-1-2016-1-UK-EPPKA2-CBHE-JP).

Наведена структура робіт з перевірки знань з курсу, відповідний тренінговий матеріал, приклади виконання завдань та критерії оцінювання. В процесі навчання надаються теоретичні аспекти розроблення та впровадження IoT – систем. Наведені базові концепції та підходи науки про дані для IoT – систем.

Призначено для інженерів, розробників та науковців, які займаються розробкою та впровадженням IoT для промислових систем, для аспірантів університетів, які навчаються за напрямом комп’ютерних наук, комп’ютерної та програмної інженерії, а також для викладачів відповідних курсів.

Бібл. – 76, рисунків – 60, таблиць – 7.

Затверджено Вченою радою Національного аерокосмічного університету «Харківський авіаційний інститут» (запис № 4, грудень 19, 2018).

УДК 004.415/.416.04(076.5)=111

ISBN 978-617-7361-86-1

Abbreviations

AI – Artificial Intelligence

ANN – Artificial Neural Network

CAP – Consistency, Availability, Partition tolerance

CQL – Cassandra Query Language

DNN – Deep Neural Network

IoT - Internet of Things

MS – Microsoft

NoSQL – Not (Not Only) SQL

OS – Operating System

PDF – Probability Distribution Function

RAM – Random Access Memory

RDBMS – Relational Data Base Management System

SQL – Structured Query Language

INTRODUCTION

Training support package for course “Foundations of data science for IoT”, was designed for master students within the framework ERASMUS+ ALIOT “Internet of Things: Emerging Curriculum for Industry and Human Applications” (573818-EPP-1-2016-1-UK-EPPKA2-CBHE-JP)[1].

The main aim of the course is to give MSc students deep understanding of the fundamentals of big data science for IoT and IoE applications. The course covers typical data science algorithms with an emphasis on IoT datasets and specific examples in area IoT and IoE.

The module MC2.1 “Foundations of Data Science for IoT and IoE” contains 3 labs.

The first lab is intended to study data normalization and visualization technique for future data analysis and processing.

The second lab deals with approaches to time-series data segmentation and window function used for sensor data.

The third lab covers data analysis for real-time monitoring using visualization, clustering, and anomaly detection techniques.

The module MC2.2 “Data mining and processing for IoT” contains 4 laboratory works.

The first lab provides an exploratory data analysis and visualization technique for obtaining information of combine data from IoT-based systems and historical data.

[1] *The European Commission's support for the production of this publication does not constitute an endorsement of the contents, which reflect the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.*

The second lab covers is to study the data classification technique by combine data in IoT-based system for obtaining hidden knowledge from historical and IoT data for prediction purposes.

The third lab is intended to identify associations between input and output variables in IoT.

The fouth lab is about association analysis for pattern recognition where students encouraged discovering association analysis for IoT data analytics.

The module MC2.3 “Deep learning for IoT” contains 1 seminar and 3 labs.

The seminar is intended to obtain experience in data preparation for deep neural network using Matlab Datastore.

The first lab aims to obtain practical experience of applying the artificial neural networks for recognition of alphanumeric information

The second lab is intended to obtain practical skills in deep neural networks architecture design for Image classification problems for IoT devices.

The third lab is dirrected to discover the possibility of deep neural networks to recognize voice command.

The module MC2.4 “Big Data for IoT Based Systems” includes 5 lab works to acquaint students with the process of deploying and running Hadoop clusters provisioned by HDInsight on Linux VMs to process big data.

The first lab is intended to study learn how to create an HDInsight cluster running Linux.

The second lab describes how to connect to the Hadoop cluster deployed on Microsoft Azure Cloud remotely from a Windows or Linux PC using SSH.

The third lab uses Apache Hive on HDInsight Hadoop clusters to query and analyze data.

The forth lab explains how to use MapReduce operations coded in Python or other languages on HDInsight cluster to analyze text file.

Finally, in the fifth lab students will learn how to delete HDInsight cluster when it is no longer needed to avoid unnecessary charges.

The course is intended for engineers, developers and scientists engaged in the development and implementation of of IoT-based systems, for postgraduate students of universities studying in areas of IoT, cybersecurity in networks, computer science, computer and software engineering, as well as for teachers of relevant courses.

Practicum prepared by Professor, Head of Computer Science and Engineering Department of V. Dahl East Ukrainian National University, DrS. Skarga-Bandurova I.S., Professor of Computer systems and cybersecurity, NAU “KhAI” DrS. Gorbenko A.V., Associate Professor, Dr. Boloborodova T.O., Professor of Information Computing Systems and Control Department of Ternopil National Economic University, DrS.Sachenko A.O., Associate Professor, Dr. Koval V.S., Associate Professor, Dr. Tarasyuk O. M.

General editing was performed by Professor, Head of Computer Science and Engineering Department of V. Dahl East Ukrainian National University, DrS. Skarga-Bandurova I.S. and Professor of Computer systems and cybersecurity, NAU “KhAI” DrS. Gorbenko A.V.

The authors are deeply grateful to the reviewers, colleagues, staff of the departments of academic universities, and industrial partners for valuable information, assistance and constructive suggestions that were made during the course program discussion and assistance materials.

Foundations of Data Science for IoT and IoE

I.S. Skarga-Bandurova and T.O. Biloborodova

Laboratory work 1
DISCOVERING OF DATA PROCESSING TECHNIQUE FOR TIME SERIES

Goal and objectives: this laboratory work study of approaches to processing of time-series data analysis. We'll study data normalization and visualization technique for future data analysis and processing.

Learning objectives:

- study normalization technique of data obtained from IoT sensors / devices;
- study process of data visualization of data obtained from IoT sensors / devices.

Practical tasks:

- acquire practical skills in IoT data processing;
- to perform visualization for data of IoT sensors / devices.

Exploring tasks:

- discover using of data normalization for IoT data monitoring;
- investigate methods of data normalization.

Setting up

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual, and in [1];
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

Python 2.7/3.6 - https://www.python.org/
Jupyter Notebook - https://jupyter.org/

1.1 Synopsis

In this laboratory work you will learn processing of data analysis and visualization of IoT monitoring data.

1.2 Brief theoretical information:

IoT data usually collected from multiple sources. Sources may include multiple sensors, devices. Redundancy could arise during integration of data that we wish to have for knowledge discovery. So, data integration must be done carefully to avoid redundancy and inconsistency that in turn improve the accuracy and speed up the knowledge discover process [1].

The careful data integration is now acceptable but it needs to be transformed into forms suitable for data analysis. Data transformation involves smoothing, a generalization of the data, attribute construction and normalization.

Data transformation such as normalization may improve the accuracy and efficiency of data analysis involving machine learning algorithms. Such methods provide better results if the data to be analyzed have been normalized, that is, scaled to specific ranges.

An attribute is normalized by scaling its values so that they fall within a small-specified range, for example, such as -1 to 1. Normalization is particularly useful for classification algorithms involving neural networks, or distance measurements such as the nearest neighbor classification and clustering. If using the neural network backpropagation algorithm for classification mining, normalizing the input values for each attribute measured in the training samples will help speed up the learning phase. For distanced-based methods, normalization helps prevent attributes with initially large ranges from outweighing attributes with initially smaller ranges [1]. There are many methods for data normalization include min-max normalization, z-score normalization, and normalization by decimal scaling.

Min-max normalization performs a linear transformation on the original data. Suppose that min_a and max_a are the minimum and the maximum values for variable *a*. Min-max normalization maps a value *v* of *a* to *v'* in the range [new-min_a, new-max_a] by calculating:

$$v' = \left(\left(v - min_a\right) \ / \ \left(max_a - \ min_a\right) \right) \ * \ \left(new\text{-}max_a - \ new\text{-}min_a\right) + \\ + \ new\text{-}min_a .$$

In z-score normalization, the values of variable *a* are normalized based on the mean and standard deviation of *a*. A value *v* of *a* is normalized to *v'* by computing:

$$v' = ((v - \bar{a}) / \sigma_a),$$

where $\bar{a}$ and σ_a are the mean and the standard deviation respectively of variable *a*. This method of normalization is useful when the actual minimum and maximum of variable *a* are unknown.

Normalization by decimal scaling normalizes by moving the decimal point of values of variable *a*. The number of decimal points moved depends on the maximum absolute value of *a*. A value *v* of *a* is normalized to *v'* by calculating:

$$v' = (v / 10^j),$$

where *j* is the smallest integer such that Max(|v'|)<1.

Normalization can change the original data and it is necessary to save the normalization parameters (the mean and the standard deviation if using the z-score normalization and the minimum and the maximum values if using the min-max normalization) so that future data can be normalized in the same method.

1.3 Execution order and discovery questions:

1. Use data source or file of time-series data.

Instructions

2. Download & install Python (the version that is convenient for you) and Jupyter Notebook

3. Import necessary Python libraries:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.preprocessing import minmax_scale
from scipy import stats
Install PyQt4 (for Python 2.7); or PyQt5 (for Python 3.6)
Install pyqtgraph
```

4. Data processing

4.1 Data normalization, transform it to range from -1 to 1

```
df = minmax_scale(df, feature_range= (-1,1))
```

4.2 Define median and biased it on 0

```
stats.describe(df)
df = df - stats.describe(df)[2]
```

4.3 Get the plot of the first 600 time steps

```
plt.figure(figsize=[20,10])
plt.plot(df[:600])
```

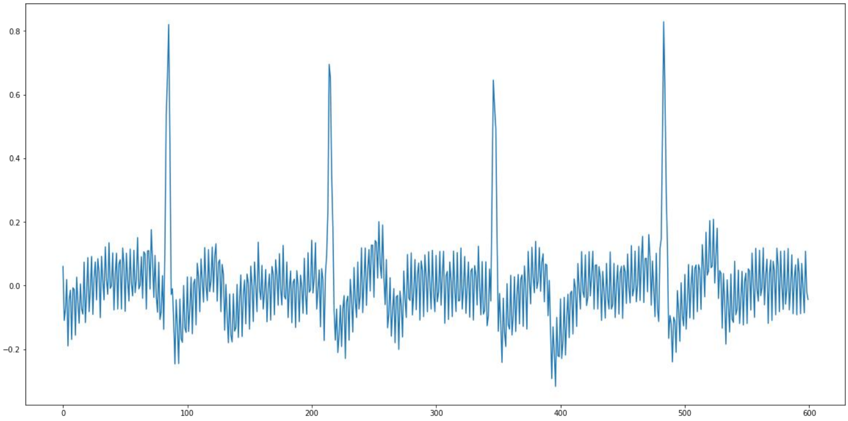


Fig. 1.1 – The visualized signal

1.4 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, figure / scheme of testing environment, procedures
- (R): narrate (like a story), tables, indicate final results;
- (D): answers on discovery questions, explanation of anomalies, conclusion / summary

1.5 Test questions:

1. Specify (*synonyms*: list, make a list, etc.) approaches for data normalization.
2. What is the purpose of data preprocessing?

3. How to perform data normalization?
4. How data normalization can improve IoT data?

1.6 Recommended literature:

1. J. Han, P. Jian, M. Kamber, ″*Data mining: concepts and techniques*″, Elsevier, 2011.

Laboratory work 2
EXPLORING OF SIGNAL SEGMENTATION AND USING OF WINDOW FUNCTION

Goal and objectives: this laboratory work study of approaches to time-series data segmentation and window function.

Learning objectives:

- study method of data segmentation for time-series data;
- study approach of window function using.

Practical tasks:

- acquire practical skills in signal segmentation;
- acquire practical skills in window function using.

Exploring tasks:

- investigate the technique of splitting the waveform into overlapping segments;
- investigate the segmentation as stages of clustering.

Setting up

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual, and in [1,2];
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

Python 2.7/3.6 - https://www.python.org/
Jupyter Notebook - https://jupyter.org/

2.1 Synopsis

In this laboratory work you will learn and explore technique of biomedical signal segmentation and window function used for data from sensor/devices monitoring.

2.2 Brief theoretical information:

Unsupervised learning from IoT-based system for health monitoring include:

- Automated detection of abnormal events.

- Automated detection of anomalies time series data.
- Models for data prediction.

Anomaly detection includes:

- anomalies detection by looking for any values beyond a certain threshold;
- anomalies detection by the structure of the waveform;
- more subtle errors - a change in the shape of a periodic waveform.

Cluster analysis is used to define an anomaly as being some pattern in the ECG waveform that hasn't been seen before. Created a library of normal waveform shapes is used to try and reconstruct a waveform to be tested. If the reconstruction is poor, then the waveform is likely to contain something abnormal and is, therefore, an anomaly.

Anomalies detection carried out as follow.

- Split ECG waveform into segments of n samples
- Space formation in n dimensions, with each segment representing one point
- Clustering determination of segment points, and determination the centroids of the clusters
- Cluster centroids provide a library of normal waveform shapes
- ECG waveform reconstruct for tested using cluster centroids learned during training
- Reconstruction error on any segment indicates the anomalous shape

Split the waveform into overlapping segments, with the section of the original data sampled sliding along by two samples each time. This approach is used to get instances of each waveform shape with a variety of horizontal translations. The approach is to apply a window function to the data, which leads the start and end of the signal to be zero.

2.3 Execution order and discovery questions:

1. Perform steps 1- 4 of laboratory work 1.

2. Signal segmentation

2.1 Divide data to segments

```
import numpy as np
segment_len = 64
```

```
slide_len = 4
segments = []
for start_pos in range(0, len(df), slide_len):
    end_pos = start_pos + segment_len
    # make a copy so changes to 'segments' doesn't
modify the original ecg_data
    segment = np.copy(df[start_pos:end_pos])
    # if we're at the end and we've got a truncated
segment, drop it
    if len(segment) != segment_len:
        continue
    segments.append(segment)
print("Produced %d waveform segments" % len(segments))
```

2.2 Get segments and visualized its

```
fig, axes = plt.subplots(5,5, figsize=[20,10])
graph_cnt = 0
for i in range(5):
    for j in range(5):
        axes[i,j].plot(segments[graph_cnt])
        graph_cnt+=3
```

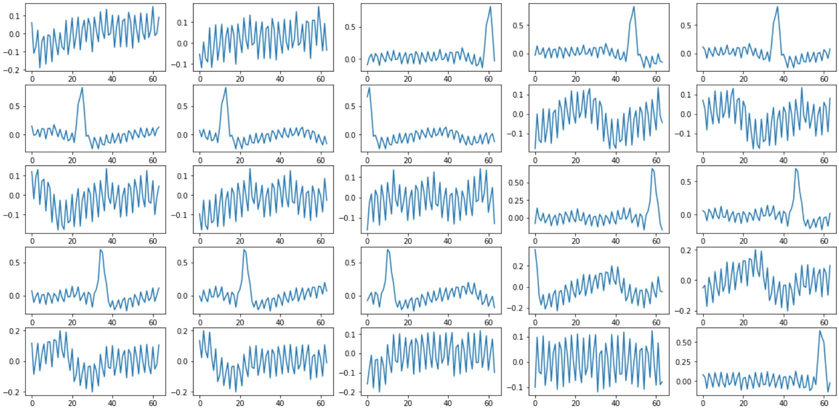


Fig. 2.1 – The visualization of got segments

3. Rationing the obtained segments using the sine-window function

3.1 Create the window

```
window_rads = np.linspace(0, np.pi, segment_len)
window = (np.sin(window_rads)**2)
```

```
plt.plot(window)
plt.show()
```

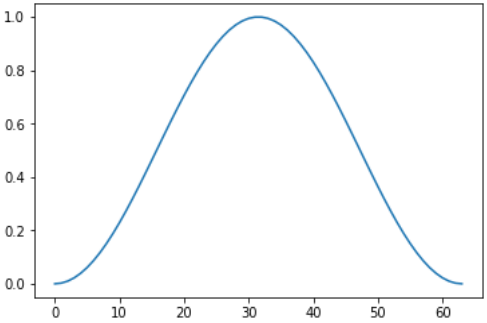


Fig. 2.2 – The visualization of created window of ECG signal

3.2 Used the sine-window function

```
windowed_segments = []
for segment in segments:
    windowed_segment = np.copy(segment) * window
    windowed_segments.append(windowed_segment)
```

3.3 Get windowed segments and visualized its

```
fig, axes = plt.subplots(3,2, figsize=[20,10])
graph_cnt = 0

for i in range(3):
    for j in range(2):
        axes[i,j].plot(windowed_segments[graph_cnt])
        graph_cnt+=3
```

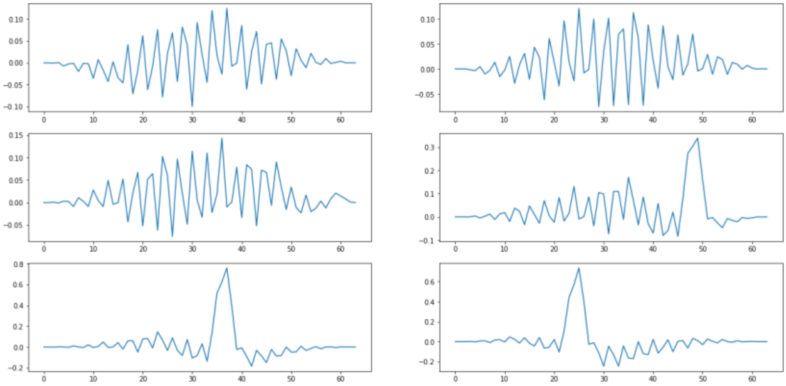


Fig. 2.3 – The visualization of got windowed segments

2.4 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, figure / scheme of testing environment, procedures
- (R): narrate (like a story), tables, indicate final results;
- (D): answers on discovery questions, explanation of anomalies, conclusion / summary

2.5 Test questions:

1. What are the main steps of anomaly detection?
2. How can is anomalies detection carried out?
3. What techniques can be used for splitting the waveform into overlapping segments?

2.6 Recommended literature:

1. T. Dunning, E. Friedman. "Time series databases." *New Ways to Store and Access data*, 2015.

2. S. Chen, W. Hua, Z. Li, J. Li, X. Gao, "Heartbeat classification using projected and dynamic features of ECG signal", *Biomedical Signal Processing and Control*, vol. 31, pp. 165–173. doi:10.1016/j.bspc.2016.07.010

Laboratory work 3
ANOMALY DETECTION IN TIME SERIES USING CLUSTERING

Goal and objectives: This laboratory work study of approaches to IoT monitoring. We'll study data analysis for real-time monitoring using visualization, clustering, and anomaly detection techniques.

Learning objectives:

- study clustering technique for biomedical signals;
- study signal reconstruction for anomaly detection.

Practical tasks:

- acquire practical skills in working with ECG signal for pattern recognition;
- to perform unsupervised learning technique, such as clustering to ECG signal labeling.

Exploring tasks:

- discover certain biomedical states using techniques of knowledge discovering;
- investigate clustering and anomaly detection technique to biomedical data analysis.

Setting up

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual, and in [1];
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

Python 2.7/3.6 - https://www.python.org/

Jupyter Notebook - https://jupyter.org/

3.1 Synopsis

In this laboratory work you will learn techniques of anomaly detection to real-time series monitoring using clustering.

3.2 Brief theoretical information:

Cluster analysis include follow step.

1. Feature selection or extraction. Feature selection chooses distinguishing features from a set of candidates and feature extraction uses data transformations to generate useful and novel features.

2. Clustering algorithm design or selection. This step is usually combined with the selection of a corresponding proximity measure. Patterns are grouped according to whether they resemble each other and the proximity measure directly affects the formation of the resulting clusters.

3. Cluster validation. Different approaches generally lead to different clusters and even for the same algorithm, parameter identification or the sequence of input patterns may affect the final results. Consequently, effective evaluation standards and criteria are important to provide the researcher with a degree of confidence for the results derived from the used algorithms.

4. Results interpretation. The ultimate goal of clustering is to provide meaningful insights from the original data.

The particular method we'll be using is called *k-means clustering*.

The k-means algorithm clusters data by trying to separate samples in *n* groups of equal variance, minimizing a criterion known as the inertia or within-cluster sum-of-squares. This algorithm requires the number of clusters to be specified. It scales well to a large number of samples and has been used across a large range of application areas in many different fields.

The k-means algorithm divides a set of *N* samples *X* into *K* disjoint clusters *C*, each described by the mean μ_j of the samples in the cluster. The means are commonly called the cluster "centroids"; note that they are not, in general, points from ***X***, although they live in the same space. The k-means algorithm aims to choose centroids that minimize the *inertia*, or within-cluster sum of squared criterion:

$$\sum_{i=0}^{n} \min_{\mu_i \in C} \left(\left\| x_j - \mu_i \right\|^2 \right). \tag{3.1}$$

Inertia, or the within-cluster sum of squares criterion, can be recognized as a measure of how internally coherent clusters are.

The properties of the k-means clustering algorithm are present in Table 3.1.

Table 3.1 – The properties of the k-means cluster algorithm

k-means algorithm	**Properties**
Parameters	Number of clusters
Scalability	Very large N samples, medium K clusters
Usecase	General-purpose, even cluster size, flat geometry, not too many clusters
Metric used	Distances between points (centroids)

ECG signal clustering consists of following steps.

Signal segmentation - splitting data into overlapping segments.

For the selected segment, find the centroid of the cluster that best matches this segment.

Segment reconstruction – the cluster centroid is used for this segment.

Data signal reconstruction.

Anomaly has produced a shape in the waveform that hadn't been seen before, the waveform around that point couldn't be reconstructed using the learned shape library. This gives a reconstruction error, that could be easily detected.

3.3 Execution order and discovery questions:

1. Perform steps 1- 4 of laboratory work 2
2. Data clustering provide the first experiments using segments by (2) from laboratory work 2:

2.1 Import clustering function

```
from sklearn.cluster import KMeans
```

2.2 Model initialization and training using the k-means algorithm without window function

```
cluster = KMeans(n_clusters=150)
cluster.fit(segments)
```

2.3 Defined clusters centroids

```
fig, axes = plt.subplots(3,3, figsize=[20,10])
```

```
graph_cnt = 0
for i in range(3):
    for j in range(3):
axes[i,j].plot(cluster.cluster_centers_[graph_cnt])
        graph_cnt+=15
```

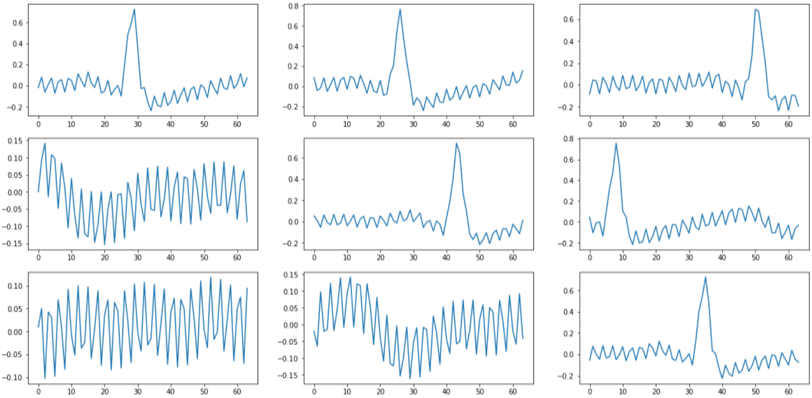


Fig. 3.1 – The visualization of defined clusters centroids

3. Run reconstraction

3.1 Import necessary library

```
import learn_utils
```

3.2 Define length of segments

```
slide_len = segment_len/2
test_segments = learn_utils.sliding_chunker(
    df,
    window_len=segment_len,
    slide_len=int(slide_len)
```

3.3 Reconstraction of single segment

```
  centroids = cluster.cluster_centers_
  segment = np.copy(test_segments[10])
  # remember, the clustering was set up using the windowed
data
  # so to find a match, we should also window our search
key
  windowed_segment = segment * window
```

```
   # predict() returns a list of centres to cope with the
possibility of multiple
   # samples being passed
   nearest_centroid_idx =
cluster.predict(windowed_segment.reshape(1,-1))[0]
   nearest_centroid =
np.copy(centroids[nearest_centroid_idx])
   plt.figure(figsize=[20,10])
   plt.plot(segment, label="Original segment")
   plt.plot(windowed_segment, label="Windowed segment")
   plt.plot(nearest_centroid, label="Nearest centroid")
   plt.legend()
   plt.show()
```

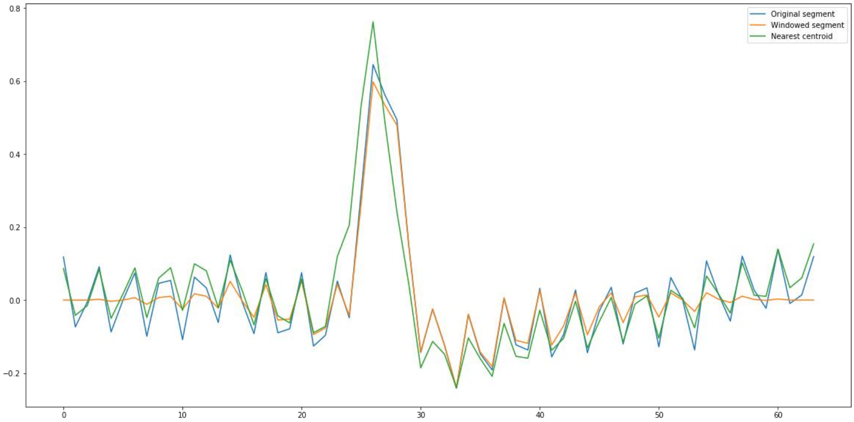


Fig. 3.2 – The visualization of original, windowed segments, clusters centroid

```
   fig, axes = plt.subplots(3,1, figsize=[20,30])
   original_s, = axes[0].plot(segment)
   windowed_s, = axes[1].plot(windowed_segment,
color='orange')
   nearest_s, = axes[2].plot(nearest_centroid, color =
'green')
   plt.legend([original_s,windowed_s,nearest_s], ["Original
segment","Windowed segment","Nearest centroid"])
   plt.show();
```

3.4 Reconstraction of all segments

```
   reconstruction = np.zeros(len(df))
   slide_len = segment_len/2
   for segment_n, segment in enumerate(test_segments):
       # don't modify the data in segments
       segment = np.copy(segment)
   #     segment *= window
       nearest_centroid_idx =
cluster.predict(segment.reshape(1,-1))[0]
       centroids = cluster.cluster_centers_
       nearest_centroid =
np.copy(centroids[nearest_centroid_idx])

       # overlay our reconstructed segments with an overlap
of half a segment
       pos = segment_n * int(slide_len)
       reconstruction[pos:pos+segment_len] +=
nearest_centroid

   n_plot_samples = 600

   error = reconstruction[0:n_plot_samples] -
df[0:n_plot_samples]
   error_98th_percentile = np.percentile(error, 98)
   print("Maximum reconstruction error was %.1f" %
error.max())
   print("98th percentile of reconstruction error was %.1f"
% error_98th_percentile)

   plt.figure(figsize=[20,10])
   plt.plot(df[0:n_plot_samples], label="Original EKG")
   plt.plot(reconstruction[0:n_plot_samples],
label="Reconstructed EKG")
   plt.plot(error[0:n_plot_samples], label="Reconstruction
Error")
   plt.legend()
   plt.show()
```

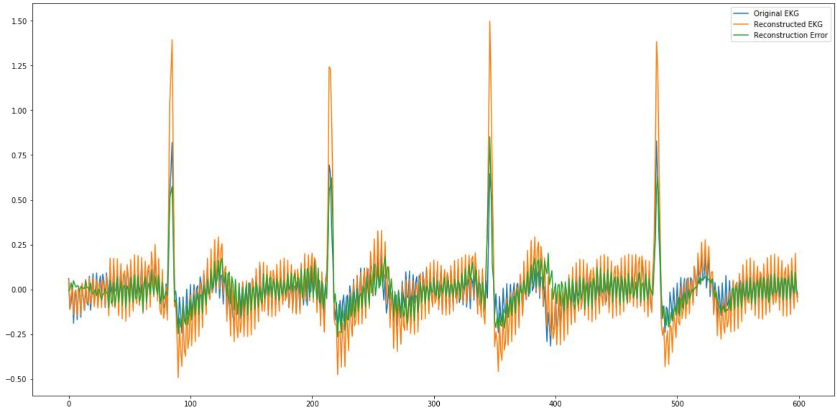


Fig. 3.3 – The visualization of original, reconstructed ECG signals, reconstructed error

```
   fig, axes = plt.subplots(3,1, figsize=[20,30])
   original_ekg, = axes[0].plot(df[0:n_plot_samples])
   reconstructed_ekg, =
axes[1].plot(reconstruction[0:n_plot_samples],
color='orange')
   reconsruction_error, =
axes[2].plot(error[0:n_plot_samples], color = 'green')
   plt.legend([original_s,windowed_s,nearest_s], ["Original
ekg","Reconstructed EKG","Reconstruction Error"])
   plt.show();
```

4. Data clustering provide the second experiments using windowed segments by (3) from laboratory work 2.

4.1 Model initialization and training using the k-means algorithm with window function

```
cluster = KMeans(n_clusters=150)
cluster.fit(windowed_segments)
```

4.2 Defined clusters centroids

```
fig, axes = plt.subplots(3,3, figsize=[20,10])
graph_cnt = 0
for i in range(3):
    for j in range(3):
```

```
axes[i,j].plot(cluster.cluster_centers_[graph_cnt])
           graph_cnt+=15
```

5. Run reconstraction

5.1 Define length of segments

```
slide_len = segment_len/2
test_segments = learn_utils.sliding_chunker(
    df,
    window_len=segment_len,
    slide_len=int(slide_len)
)
```

5.2 Reconstraction of single windowed segment

```
centroids = cluster.cluster_centers_
segment = np.copy(test_segments[10])
# remember, the clustering was set up using the windowed
data
# so to find a match, we should also window our search
key
windowed_segment = segment * window

# predict() returns a list of centres to cope with the
possibility of multiple
# samples being passed
nearest_centroid_idx =
cluster.predict(windowed_segment.reshape(1,-1))[0]
nearest_centroid =
np.copy(centroids[nearest_centroid_idx])
plt.figure(figsize=[20,10])
plt.plot(segment, label="Original segment")
plt.plot(windowed_segment, label="Windowed segment")
plt.plot(nearest_centroid, label="Nearest centroid")
plt.legend()
plt.show()
```

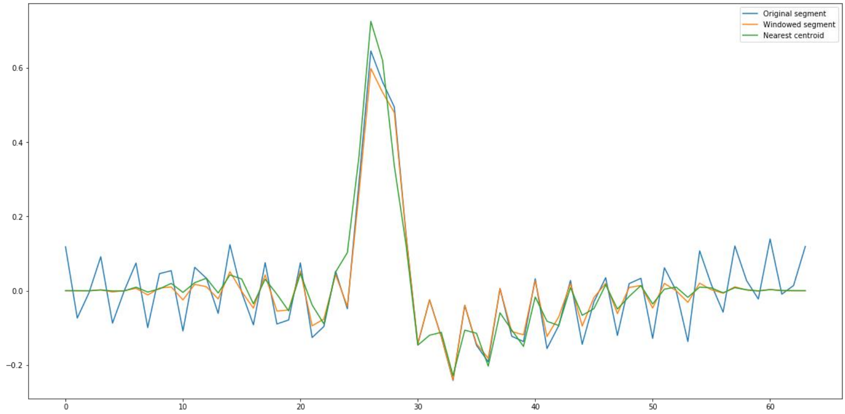


Fig. 3.4 – The visualization of original, windowed segments, nearest centroid

```
   fig, axes = plt.subplots(3,1, figsize=[20,30])
   original_s, = axes[0].plot(segment)
   windowed_s, = axes[1].plot(windowed_segment,
color='orange')
   nearest_s, = axes[2].plot(nearest_centroid, color =
'green')
   plt.legend([original_s,windowed_s,nearest_s], ["Original
segment","Windowed segment","Nearest centroid"])
   plt.show();
```

5.3 Reconstraction of all windowed segments

```
   reconstruction = np.zeros(len(df))
   slide_len = segment_len/2
   for segment_n, segment in enumerate(test_segments):
       # don't modify the data in segments
       segment = np.copy(segment)
   #     segment *= window
       nearest_centroid_idx =
cluster.predict(segment.reshape(1,-1))[0]
       centroids = cluster.cluster_centers_
       nearest_centroid =
np.copy(centroids[nearest_centroid_idx])
       # overlay our reconstructed segments with an overlap
of half a segment
       pos = segment_n * int(slide_len)
```

```
      reconstruction[pos:pos+segment_len] +=
nearest_centroid

  n_plot_samples = 600

  error = reconstruction[0:n_plot_samples] -
df[0:n_plot_samples]
  error_98th_percentile = np.percentile(error, 98)
  print("Maximum reconstruction error was %.1f" %
error.max())
  print("98th percentile of reconstruction error was %.1f"
% error_98th_percentile)

  plt.figure(figsize=[20,10])
  plt.plot(df[0:n_plot_samples], label="Original EKG")
  plt.plot(reconstruction[0:n_plot_samples],
label="Reconstructed EKG")
  plt.plot(error[0:n_plot_samples], label="Reconstruction
Error")
  plt.legend()
  plt.show()
```

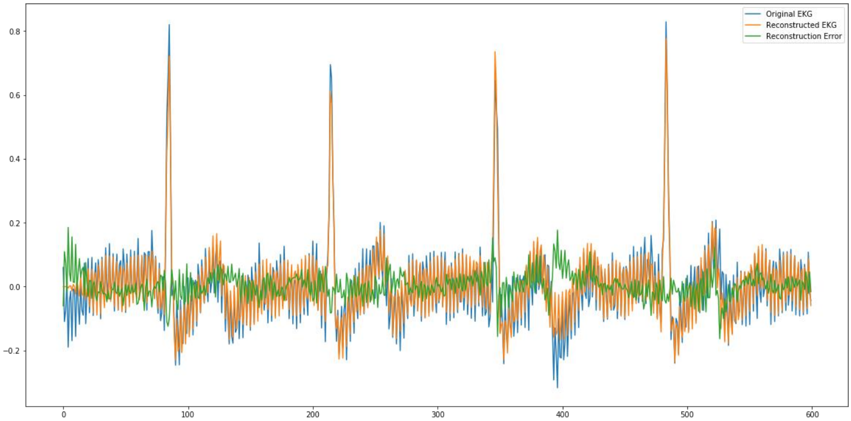


Fig. 3.5 – The visualization of original, reconstructed ECG signals, reconstructed error

```
  fig, axes = plt.subplots(3,1, figsize=[20,30])
  original_ekg, = axes[0].plot(df[0:n_plot_samples])
  reconstructed_ekg, =
axes[1].plot(reconstruction[0:n_plot_samples],
color='orange')
```

```
   reconsruction_error, =
axes[2].plot(error[0:n_plot_samples], color = 'green')
   plt.legend([original_s,windowed_s,nearest_s], ["Original
ekg","Reconstructed EKG","Reconstruction Error"])
         plt.show();
```

5. Get the maximum reconstruction error.
6. Describe and concluded obtained result.

3.4 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, figure / scheme of testing environment, procedures
- (R): narrate (like a story), tables, indicate final results;
- (D): answers on discovery questions, explanation of anomalies, conclusion / summary

3.5 Test questions:

1. Specify (*synonyms*: list, make a list, etc.) steps of clustering analysis.
2. Specify (*synonyms*: list, make a list, etc.) clustering algorithm, defined it properties.
3. What steps ECG signal clustering consist of?

3.6 Recommended literature:

1. F. Al-Turjman, "Artificial Intelligence in IoT", *Transactions on Computational Science and Computational Intelligence*, 2019. doi:10.1007/978-3-030-04110-6

Reading list

1. T. Dunning, E. Friedman. "Time series databases." *New Ways to Store and Access data*, 2015.

2. "IoT - Internet of Things", *San Diego Consulting Group* [Online]. Available: https://www.sandiegoconsultinggroup.com/ioe-m2m-iot-paas [Accessed: 22- June- 2019].

3. A. Jaokar, "Data Science for Internet of Things (IoT): Ten Differences From Traditional Data Science", *KDnuggets*, 2016. [Online]. Available: http://www.kdnuggets.com/2016/09/data-science-iot-10-differences.html [Accessed: 22- June- 2019].

4. M. Abu-Elkheir, M. Hayajneh, N.A. Ali, "Data management for the internet of things: design primitives and solution", *Sensors* (Basel, Switzerland) vol. 13,11 15582-612. 14 Nov. 2013, doi:10.3390/s131115582

5. V. Granville, "Data Science for IoT vs Classic Data Science: 10 Differences", Data Science Central. [Online]. Available: https://www.datasciencecentral.com/profiles/blogs/data-science-for-iot-vs-classic-data-science-10-differences. [Accessed: 30-Jul-2019].

6. Y. Le Cun, Y. Bengio, and G. Hinton, "Deep learning", *Nature*, vol. 521, no. 7553, pp. 436, 2015.

7. Hu, X., Hu, S., Zhang, J., Kong, W. and Cao, Y.,. A fatal drawback of the widely used Granger causality in neuroscience. *IEEE International Conference on Information Science and Technology (ICIST)*, pp. 61-65, May 2016.

8. A. Jaokar, J.-J. Bernard, "Data Science for Internet of Things", *CreateSpace Independent Publishing Platform*, 64 p. 2015.

Data Mining and Processing for IoT

I.S. Skarga-Bandurova and T.O. Biloborodova

Laboratory work 1
EXPLORATORY DATA ANALYSIS AND DATA VISUALIZATION FOR IOT

Goal and objectives: the aim of the laboratory work is to study the exploratory data analysis and visualization technique for obtaining information of combine data from IoT-based systems and historical data.

Learning objectives:

- study methods of data properties obtained from IoT-based systems and historical data;
- study methods of data visualization for exploratory data analysis.

Practical tasks:

- acquire practical skills in IoT data analytics;
- acquire practical skills in obtaining information of combine data from IoT-based systems and historical data;
- to perform visualization for data properties obtaining;
- to perform statistical analysis for data properties obtaining.

Exploring tasks:

- discover using of exploratory data analysis for IoT data analytics;
- investigate data visualization technique.

Setting up

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual, and in [1-5];
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

Python 2.7/3.6 - https://www.python.org/
Jupyter Notebook - https://jupyter.org/

1.1 Synopsis

In this laboratory work you will learn exploratory data analysis and visualization of combine IoT and historical data for more deep knowledge about its properties obtaining.

1.2 Brief theoretical information:

The primary aim of designed experiments is to understand the sources of variation in one or more responses and to assess the effects of factors on those responses relative to other unexplained variation. A clear understanding of the various variables (attributes) and the way they are organized in the experimental process is vital to proper data analysis.

Many experiments focus on the interplay of several variables. Sometimes it is possible to isolate the main effect of each variable, allowing direct interpretation of the effect of changing levels of that factor on response without regard to any other variable. However, there can be interaction (synergy or antagonism) among variables, which requires careful interpretation of the effects of one variable on response. The analysis is more complicated with unbalanced designs. Some experiments involving several variables do not readily lend themselves to interpretation in terms of main effects and interactions.

What is the nature of the data? How were they obtained and how are they maintained? While this information is primarily in the form of numbers, the processes surrounding the gathering and organizing of data are crucial to sound data analysis. Data quality greatly affects the chance of successfully addressing key questions. We must endeavor to understand how the data are gathered and what are the possible sources of error which arise during the experimental process.

Also, for making a decision concerning managing data, it is necessary to take into account the data type (continuous, ordinal, nominal, etc.). According to [1], data can be classified into follow types: numerical and categorical.

Numerical data can be further broken into two types: discrete and continuous. Discrete data represent items that can be counted; they take on possible values that can be listed out. The list of possible values may be fixed (also called finite); or it may go from 0, 1, 2, on to infinity (making it countably infinite). Continuous data represent measurements; their possible values cannot be counted and can only be described using intervals on the real number line.

Categorical data can be further broken into two types: nominal and ordinal. Nominal values or observation can be present in the form of a number where the number are simply labels. Nominal data can be count, but not order or measure. Nominal data are used for labeling variables, without any quantitative value.

Ordinal values or observation can be ranked (put in order) or have a rating scale attached. Ordinal data can be count and order, but not measure. When the categories may be ordered, these are called ordinal variables. Ordinal data mixes numerical and categorical data. The data fall into categories, but the numbers placed on the categories have meaning. Ordinal data the numbers do have mathematical meaning.

Categorical data is a measurement expressed not in terms of numbers, but rather by means of a natural language description. Categorical data represent characteristics such as a person's gender, marital status, hometown, or the types of movies they like.

The preliminary stage of data processing is descriptive analytics. It used to create a summary of data to yield useful information and possibly prepare the data for further analysis.

Descriptive analytics in IoT usually used traditional approaches. It used in different areas of the IoT. There are many opinions about the classification of analytics in the IoT domains. Bertolucci et al. [2] propose descriptive, predictive, and prescriptive categories while Gartner [3, 4] proposes the extra category of diagnostic analytics. Siow et. al. [5] formulated a comprehensive classification of analytic capabilities consisting of five categories: descriptive, diagnostic, discovery, predictive, and prescriptive analytics.

Descriptive analytics helps us to answer the question, "What happened?" It can take the form of describing, summarizing, or presenting raw IoT data that has been gathered. Data are decoded, interpreted in context, fused, and then presented so that it can be understood and might take the form of a chart, a report, statistics, or some aggregation of information.

1.3 Execution order and discovery questions:

1. Use data interested you.

Instructions

2. Download & install Python (the version that is convenient for you) and Jupyter Notebook

2.1 Import necessary Python libraries:

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import matplotlib
import seaborn as sns
%matplotlib inline
```

3. Use data source or file of data:

```
df = pd.read_csv('dataset.csv')
df.head()
```

4. Obtained information of dataset:

```
df.info()

RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
customerID          7043 non-null object
gender              7043 non-null object
SeniorCitizen       7043 non-null int64
Partner             7043 non-null object
Dependents          7043 non-null object
tenure              7043 non-null int64
PhoneService        7043 non-null object
MultipleLines       7043 non-null object
InternetService     7043 non-null object
OnlineSecurity      7043 non-null object
OnlineBackup        7043 non-null object
DeviceProtection    7043 non-null object
TechSupport         7043 non-null object
StreamingTV         7043 non-null object
StreamingMovies     7043 non-null object
Contract            7043 non-null object
PaperlessBilling    7043 non-null object
PaymentMethod       7043 non-null object
MonthlyCharges      7043 non-null float64
TotalCharges        7043 non-null object
Churn               7043 non-null object
dtypes: float64(1), int64(2), object(18)
```

5. Obtained statistical parameters of dataset attributes:

```
df.describe(include='all')
```

6. Obtained information of data distribution

```
df.columns
Index(['attribute_1', 'attribute_2', 'attribute_3', …, 'attribute_n'], dtype='object')
```

6.1 Obtained information of categorical attributes

```
f, axes = plt.subplots(4, 2, sharey=True, figsize=(30,30))
axes = axes.flatten()
info_categorial = ['categorical_attribute_1', 'categorical_attribute_2', …, 'categorical_attribute_n']
```

```
for  i,col in enumerate(info_categorial):
sns.countplot(df[col], ax=axes[i])
```

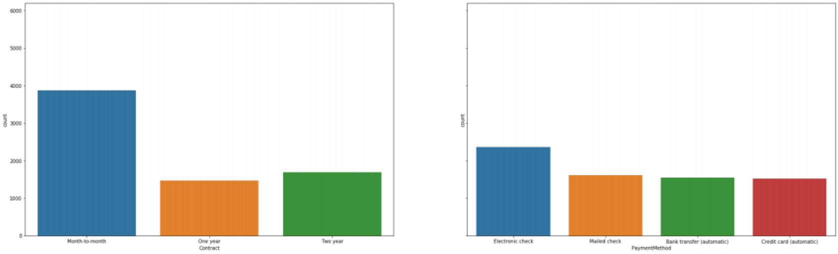


Fig. 1.1 – The visualization of categorical variables properties

6.2 Obtained information of numerical attributes

```
numeric_infos          =          ['numerical_attribute_1',
'numerical_attribute_2', …, 'numerical_attribute_n']
f, ax = plt.subplots(5,1,figsize=(20,35))
for i,col in enumerate(numeric_infos):
sns.distplot(df[col], ax = ax[i])
```

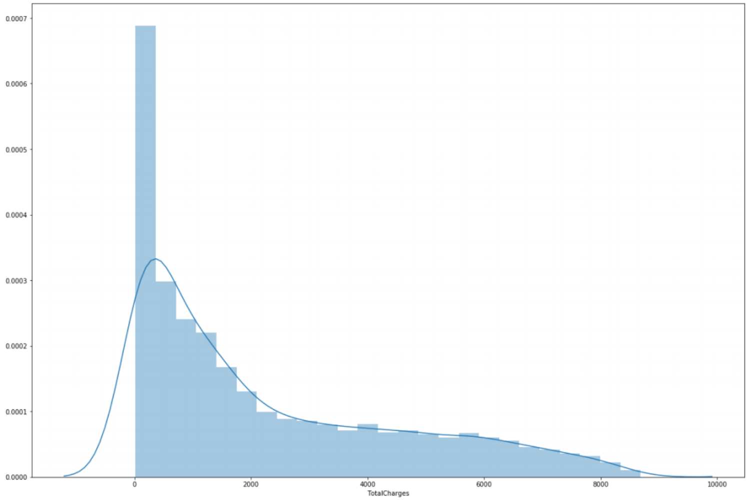


Fig. 1.2 – The visualization of numerical variables properties

7. Describe and conclude all obtained plots of data visualization

1.4 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, figure / scheme of testing environment, procedures
- (R): narrate (like a story), tables, indicate final results;
- (D): answers on discovery questions, explanation of anomalies, conclusion / summary

1.5 Test questions:

5. What types of data you can named?
6. Describe each type of data.
7. What is descriptive analytics?

1.6 Recommended literature:

1. "Everything about data science", *Scaryscientist.blogspot.com*, 2015. [Online]. Available: http://scaryscientist.blogspot.com/2015/02/classification-of-data-types.html. [Accessed: 20- Feb- 2019]
2. J. Bertolucci, "Big data analytics: Descriptive vs. predictive vs. prescriptive", *Informationweek.com*, 2013. [Online]. Available: http://goo.gl/dyNDFV. [Accessed: 20- Feb- 2019].
3. L. Kart, "Advancing Analytics. Technical Report. Gartner Inc.", *Informs.org,* 2012. [Online]. Available: http://meetings2.informs.org/analytics2013/AdvancingAnalytics. [Accessed: 20- Feb- 2019].
4. N. Chandler, B. Hostmann, N. Rayner, G. Herschel, "Gartner's Business Analytics Framework. Technical Report. Gartner Inc.", *Gartner.com*, 2011. Online]. Available: http://www.gartner.com/imagesrv/summits/docs/na/businessintelligence/gartners. [Accessed: 20- Feb- 2019].
5. E. Siow, T. Tiropanis, W. Hall, "Analytics for the Internet of Things", *ACM Computing Surveys*, vol.51, no.4, pp. 1–36. doi:10.1145/3204947.

Laboratory work 2
CLASSIFICATION AND PREDICTION MODELING IN IOT SYSTEMS

Goal and objectives: the aim of the laboratory work is to study the data classification technique by combine data in IoT-based system for obtaining hidden knowledge from historical and IoT data for prediction model creating.

Learning objectives:

- study methods of data classification technique for IoT-based systems;
- study methods of prediction model creating.

Practical tasks:

- acquire practical skills in IoT data analytics;
- acquire practical skills in data classification technique of combine data from IoT-based systems and historical data;
- acquire creating prediction model using Decision Tree algorithm by combine data from IoT-based systems and historical data;
- to perform checking of prediction model for model accuracy estimation;
- to perform approaches for prediction model improving.

Exploring tasks:

- discover using of combine from IoT-based systems and historical data for prediction model obtaining;
- investigate methods for prediction model improving.

Setting up

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual, and in [1];
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

Python 2.7/3.6 - https://www.python.org/
Jupyter Notebook - https://jupyter.org/

2.1 Synopsis

In this laboratory work you will learn and explore data classification technique by combine data in IoT-based system for obtaining hidden knowledge from historical and IoT data for prediction model creating and prognosis future event.

2.2 Brief theoretical information:

Classification is an important technique in data mining that assigns items in a collection of target categories or classes. Classifying the data sets into different categories would make it easier to understand the data. This is called supervised learning technique. In this technique, a training data is given by the use of which a classifier model is build and based on this model the future pattern is predicted.

Classification is a technique used to predict the label for data instances and is a process. The classification process aims to identify unknown data, on the basis of a training set of data containing observations whose classes are known. For example, classification in biomedicine faces several difficulties: researchers spend a long time to accumulate enough knowledge to distinguish different related cases, as normal and abnormal. Labeled data can be presented as the follow matrix, where *x* is value of input attribute, and *y* is output attribute or label.

$$\begin{bmatrix} x_{11} & \dots & x_{1m} \\ \dots & \dots & \dots \\ x_{n1} & \dots & x_{nm} \end{bmatrix} = \begin{bmatrix} y_1 \\ \dots \\ y_n \end{bmatrix}$$

Supervised learning is widely used in IoT data mining [1]. The goal of supervised learning is to predict the corresponding output vector for a given input vector. Tasks in which the output label value is discrete are known as classification problems. Classification assumes some prior knowledge to guide the partitioning process to construct a set of classifiers to represent the possible distribution of patterns.

The use of classification methods is a solution to the problem of uncertainty and incompleteness of IoT data. In this context, the use of data mining always includes solving two related tasks: defining regular links between data elements and using these patterns to solve classification problems: predicting the values of some elements from known values of other elements. There are a lot of classification methods, they also include Decision Tree learning, Naive Bayes classifier, k-nearest neighbor classifier, and classification with network information and regression methods such as Linear Regression and Logistic Regression.

The evaluation of the classification quality can be carried out using the following parameters: accuracy, sensitivity, specificity. To calculate these parameters, used classification assessment by matrix confusion. Based on the matrix confusion the sensitivity, the specificity, and the accuracy are calculated as follows (2.1-2.3) using true positive ς_{00}, false negative ς_{10}, true negative ς_{11} and false positive ς_{01} results of *n* observations.

$$Sensitivity = \frac{\varsigma_{00}}{\varsigma_{00} + \varsigma_{10}} * 100\% , \tag{2.1}$$

$$Specificity = \frac{\varsigma_{11}}{\varsigma_{11} + \varsigma_{01}} * 100\% , \tag{2.2}$$

$$Accuracy = \frac{\varsigma_{00} + \varsigma_{11}}{n} * 100\% \cdot \tag{2.3}$$

2.3 Execution order and discovery questions:

1. Use data interested you.

Instructions

2. Download & install Python (the version that is convenient for you) and Jupyter Notebook.

2.1 Import necessary Python libraries:

```
   from sklearn.tree import DecisionTreeClassifier,
export_graphviz
   from sklearn.metrics import accuracy_score
   from sklearn.model_selection import train_test_split,
GridSearchCV
```

3. Use data source or file of data:

```
   df = pd.read_csv('dataset.csv')
   df.head()
```

4. Delete unnecessary variable

```
   df.drop(['unnecessary_variable_1'],        inplace=True,
axis=1)
```

5. Transform categorical variable to binary

```
   for columns in df.columns:
   if len(df[columns].unique())==2:
   df[columns]         =         df[columns].map({'class_1_
of_variable_1':1,                                 'class_2_
of_variable_1':0,''class_1_of_variable_2':1,'class_2_of_va
riable_2':0,1:1,0:0})
```

6. Visualized describing of data

```
   df.describe(include='all').T
```

7. Divide column of categorical variables into binary

```
   df  =  pd.concat([df,  pd.get_dummies(df['variable__1'],
prefix="variable_1"),
   pd.get_dummies(df['variable__2'], prefix="variable_2"),
   pd.get_dummies(df['…'], prefix="…"),
   pd.get_dummies(df['variable_n'], prefix="attribute_n")],
axis=1)
```

8. Delete dividing column

```
   df.drop(['variable_1', 'variable_2', '…', 'variable_n'],
axis=1, inplace=True )
```

9. Output dataset

```
   df.head()
```

10. Defined output variable

```
   df_y = df['output_variable']
   df.drop(['output_variable'], axis=1, inplace=True)
```

11. Divide dataset into train and test dataset

```
   df_train,   df_holdout,   df_y_train,   df_y_holdout   =
train_test_split(df.values, df_y, test_size = 0.3)
```

12. Obtained prediction model and get the information about it

```
   tree = DecisionTreeClassifier()
   tree.fit(df_train, df_y_train)
```

13. Check the prediction model accuracy

```
tree_predict = tree.predict(df_holdout)
result = accuracy_score(df_y_holdout, tree_predict)
print( 'Accuracy: ', result)
df.describe(include='all')
```

14. Improved model accuracy using cross-validation

```
tree_params = {'max_depth': range(1,11), 'max_features': range(4,19)}
tree_grid = GridSearchCV(tree, tree_params, cv=5, n_jobs=-1, verbose=True)
tree_grid.fit(df_train, df_y_train)
```

15. Print the best model setting on cross-validation

```
print( 'The best model setting on cross-validation: ', tree_grid.best_params_)
```

16. Check prediction model accuracy of the best model setting on cross-validation and model without setting defining

```
result_cv = accuracy_score(df_y_holdout, tree_grid.predict(df_holdout))
print(' 'The best model setting on cross-validation:' , result_cv)
print( 'The model without setting defining:', result)
```

17. Compare accuracy result and make conclusion.

2.4 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, figure / scheme of testing environment, procedures
- (R): narrate (like a story), tables, indicate final results;
- (D): answers on discovery questions, explanation of anomalies, conclusion / summary

2.5 Test questions:

1. What is data classification in IoT-based systems?

2. How can accuracy of prediction model calculate?

2.7 Recommended literature:

1. C. M. Bishop, "Pattern Recognition and Machine Learning", Springer, 2006. DOI: 10.1117/1.2819119.

Laboratory work 3
ASSOCIATION ANALYSIS FOR FREQUENT PATTERN MINING IN IOT SYSTEMS

Goal and objectives: the aim of the laboratory work is to study the IoT data in order to identify associations between input and output variables.

Learning objectives:

- study methods of pattern recognition in data from IoT-based systems;
- study process of association rules generation.

Practical tasks:

- acquire practical skills in IoT data analytics;
- to perform association analysis for pattern recognition.

Exploring tasks:

- discover using of association analysis for IoT data analytics;
- investigate how to association rules generate through recognition of frequent pattern.

Setting up

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual, and in [1];

to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

Python 2.7/3.6 - https://www.python.org/
Jupyter Notebook - https://jupyter.org/

3.1 Synopsis

In this laboratory work you will learn and explore approach of association analysis for pattern recognition in data from IoT-based applications.

3.2 Brief theoretical information:

Association rule-based predictive model in the Internet of things can predict important events. Rules and associations involve many items that hard to interpret and generate a lot of outcomes. Worth-while, noting that despite all the algorithms improvements, the obtained association rules can either be too obvious, or contradict a priori knowledge, or contain redundant information. The task of IoT mining association rules is to generate minimal set of rules providing complete coverage of important signs with support and confidence greater or equal than some pre-specified thresholds of minimum support and mini-mum confidence, respectively.

In general case, association analysis algorithms generate a huge number of items and can produce up to hundreds of association rules. An association rule is an implication expression R : X→Y, where X denoted antecedent and Y denotes consequent $X \bigcap Y = \varnothing$. Both X and Y are considered as a set of conjuncts of the form c_1, c_2 ..., c_k. The strength of the association rule is measured in terms of its support, confidence and interestingness [1].

The confidence determines how frequently items in set of consequences Y appear in cases containing antecedents X. The lift computes the ratio between the confidence of rule and the support of the set in the rule consequent.

For pair of rule-candidates, binary variables R_1 and R_2 the lift is equivalent to interest factor, which is defined as follows:

$$I(R_1, R_2) = \frac{s(R_1, R_2)}{s(R_1) \cdot s(R_2)}. \tag{3.1}$$

The measure of interestingness in this case can be interpreted as follows:

$$I^{I}(R_1, R_2) \begin{cases} = 1, & \text{if } R_1 \text{ and } R_2 \text{ are independent,} \\ > 1, & \text{if } R_1 \text{ and } R_2 \text{ are positively correlated,} \\ < 1, & \text{if } R_1 \text{ and } R_2 \text{ are negatively correlated.} \end{cases} \tag{3.2}$$

In order to increase the information importance of rules, it is necessary to reduce their number and focus on potentially interesting ones. Further exploration of interestingness leads us to discovering different subjective and probabilistic measures of interestingness. To determine the interestingness of the rule, various probabilistic measures are used: support, confidence, Goodman-Kraskal, Pyatetsky-Shapiro, Laplace, etc.

3.3 Execution order and discovery questions:

1. Use data interested you

Instructions

2. Download & install Python (the version that is convenient for you) and Jupyter Notebook

2.1 Import necessary Python libraries:

```
from apyori import apriori, dump_as_json
try:
from StringIO import StringIO
except ImportError:
from io import StringIO
import json
```

3. Use data source or file of data and output information about variables

```
df = pd.read_csv('dataset.csv')
df.columns
```

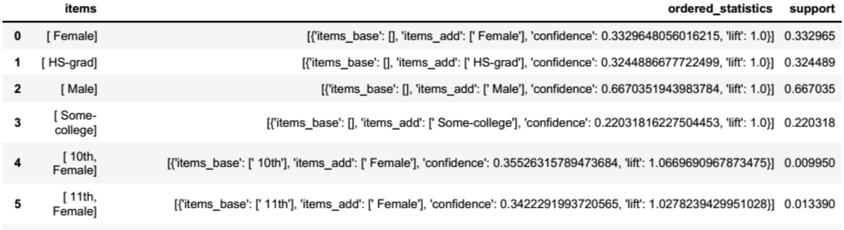
4. Defined variables interest to you, check the min support, confidence, lift, length of rule. Obtained association rules

```
transactions = df[['variables_1', 'variables_2']]
transactions = np.array(transactions)
transactions = [list(transaction) for transaction in
transactions]
transactions
result = list(apriori(transactions, min_support = 0.003,
min_confidence = 0.2, min_lift = 1, min_length = 2))
output = []
for RelationRecord in result:
o = StringIO()
dump_as_json(RelationRecord, o)
output.append(json.loads(o.getvalue()))
data_df = pd.DataFrame(output)
```

5. Visualized association rules

```
pd.set_option('display.max_colwidth', -1)
```

```
data_df
```



	items	ordered_statistics	support
0	[Female]	[{'items_base': [], 'items_add': [' Female'], 'confidence': 0.3329648056016215, 'lift': 1.0}]	0.332965
1	[HS-grad]	[{'items_base': [], 'items_add': [' HS-grad'], 'confidence': 0.3244886677722499, 'lift': 1.0}]	0.324489
2	[Male]	[{'items_base': [], 'items_add': [' Male'], 'confidence': 0.6670351943983784, 'lift': 1.0}]	0.667035
3	[Some-college]	[{'items_base': [], 'items_add': [' Some-college'], 'confidence': 0.22031816227504453, 'lift': 1.0}]	0.220318
4	[10th, Female]	[{'items_base': [' 10th'], 'items_add': [' Female'], 'confidence': 0.35526315789473684, 'lift': 1.0669690967873475}]	0.009950
5	[11th, Female]	[{'items_base': [' 11th'], 'items_add': [' Female'], 'confidence': 0.3422291993720565, 'lift': 1.0278239429951028}]	0.013390

Fig. 3.1 – The fragment of obtained rules

6. Interpret obtained rules.

3.4 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, figure / scheme of testing environment, procedures
- (R): narrate (like a story), tables, indicate final results;
- (D): answers on discovery questions, explanation of anomalies, conclusion / summary

3.5 Test questions:

1. What is associative analysis of IoT data.
2. What does support, confidence, lift show?

3.7 Recommended literature:

1. C. Ordonez, C. A. Santana, Levien de Braal ″Discovering Interesting Association Rules in Medical Data″, *In ACM SIGMOD workshop on research issues in data mining and knowledge discovery*, May 2000, pp. 78–85.

Laboratory work 4
ANALYSIS OF INTERESTINGNESS MEASURES IN FREQUENT PATTERNS OF IOT DATA

Goal and objectives: the aim of the laboratory work is to study the IoT data in order to identify interestingness measures of associative rules between input and output variables.

Learning objectives:

- study methods of pattern recognition in data from IoT-based systems;
- study objective and subjective measures of interestingness for association rules.

Practical tasks:

- acquire practical skills in IoT data analytics;
- to perform association analysis for pattern recognition;
- to perform measures of interestingness for association rules.

Exploring tasks:

- discover using of association analysis for IoT data analytics;
- investigate how to reduce number of rules-candidates.

Setting up

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual, and in [1,2];
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

R - https://www.r-project.org/

RStudio - https://www.rstudio.com/

R-package "aruls" - https://cran.r-project.org/web/packages/arules/index.html

4.1 Synopsis

In this laboratory work you will learn and explore approach of association analysis for objective and subjective measures of interestingness of association rules and anti-monotonic constraint.

4.2 Brief theoretical information:

For reducing number of rules suggest three level technique with detection of deviations in data, then testing of differences among adjusted attributes and quantifying the interestingness of association rules.

1. Detecting deviations in data is performed as follows.

The every conjunct cj from association rule set is represented in the form <A = V>, where A is an item name (attribute), Dom (A) is the domain of A, and I (value)$\in$Dom (A). Degree of deviation is defined as deviation between two conjuncts $\Delta(c_i, c_j)$ and is calculated on the basis of the comparison between the items of the two conjuncts. For conjuncts c_i, c_j deviation of c_i with respect to c_j is defined as a Boolean function as follows:

$$\Delta(c_i, c_j) = \begin{cases} 0, & \text{if } A_i = A_j \text{ and } V_i = V_j, \\ 1, & \text{if } A_i = A_j \text{ and } V_i \neq V_j. \end{cases} \tag{4.1}$$

2. The differences among adjusted attributes can be calculated using the following formula:

$$\bar{d}(R_1, R_2) = \begin{cases} 0, \ \text{if } |R_1| = |R_2| \forall c_i \in R_1, \exists\, c_j \in R_2, \text{ that } \Delta(c_i, c_j) = 0, \\ 1 \qquad\qquad\quad \forall c_i \in R_1, \neg\exists\, c_j \in R_2, \text{ that } \Delta(c_i, c_j) = 1, \\ \dfrac{\sum\limits_{c_i \in R_1, c_j \in R_2} \min \Delta(c_i, c_j)}{|R_1|}, \ \text{othervise}, \end{cases} , \tag{4.2}$$

where R_1 and R_2 are considered as two sets of conjuncts c_i and c_j.

Parameter value $\bar{d}$ = 0 indicates that R_1 and R_2 are identical, $\bar{d}$ = 1 indicates the maximum deviation between rule sets, and the other $\bar{d}$ values between 0 and 1 are defined as a transient deviation.

3. Quantifying the interestingness of association rules

Let $R_1 : X_1 \rightarrow Y_1$ and $R_2 : X_2 \rightarrow Y_2$ be two association rules, then interestingness of a rule R_1 with respect to the rule R_2 is calculated as follows:

$$I^{II}(R_1, R_2) = \begin{cases} 0, & \text{if } \bar{d}(X_1, X_2) = 0 \text{ and } \bar{d}(Y_1, Y_2) = 0, \\ \left(\min\limits_{S \in R} \bar{d}(X_1, X_2) + \bar{d}(Y_1, Y_2)\right) / 2, & \text{if } \bar{d}(X_1, X_2) \geq \bar{d}(Y_1, Y_2), \\ \left(\bar{d}(X_1, X_2) + \min\limits_{S \in R} \bar{d}(Y_1, Y_2)\right) / 2, & \text{if } \bar{d}(X_1, X_2) < \bar{d}(Y_1, Y_2), \\ 1, & \text{if } \bar{d}(X_1, X_2) = 1 \text{ and } \bar{d}(Y_1, Y_2) = 1. \end{cases} \quad (4.3)$$

According to formula (4.3), $I^{II} = 0$ indicates that R_1 and R_2 are identical, $I^{II} = 1$ denotes maximum deviation between R_1 and R_2. Other cases indicate different deviations in the interestingness of association rules. To select interesting rules the user should specify the threshold of their interestingness.

The anti-monotone property based on the threshold of the measure of interestingness can be applied to reduce the dimension of the resulting rule set. The anti-monotone property is that the measure of the interest of any set of elements should not exceed the minimal measure of interest of any of its subsets. This property greatly facilitates the mining rules.

4.3 Execution order and discovery questions:

1. Take a dataset of interest to you.

2. Download & install R, RStudio, R-package “aruls”.

3. Download dataset

```
read.table("dataset.csv", sep=",", head=TRUE)
```

4. Transform dataset to matrix and output number of steps and variables

```
dataset_matrix<-read.table("dataset.csv", sep=",",
head=TRUE)
View(dataset_matrix)
dim(birt)
```

5. Transform numerical variables to categorical according to defined label

```
   dataset_matrix[["variable_1"]] <-
ordered(cut(dataset_matrix[[ "variable_1"]],
c(min_value_of_variable,lable_1_value_of_variable,lable_2_
value_of_variable,
max_value_of_variable)),labels=c("label_1", "label_2",
"label_3"))
   dataset_matrix[[ "…"]] <- ordered(cut(dataset_matrix[[
"…"]], c(min_value_of_variable,…,…,
max_value_of_variable)),labels=c("…", "…", "…"))
   dataset_matrix[[ "variable_n"]] <-
ordered(cut(dataset_matrix[[ "variable_n"]],
c(min_value_of_variable,lable_1_value_of_variable,
max_value_of_variable)),labels=c("label_1", "label_2"))
```

6. Delete unnecessary variable

```
   dataset_matrix[["unnecessary_variable_1"]] <- NULL
   dataset_matrix[["unnecessary_variable_2"]] <- NULL
```

7. Transform data to transaction dataset and get information about frequent item

```
   transaction_dataset <- as(birt, "transactions")
   summary(transaction_dataset)
```

8. Defined min support, confidence, lift and mining associating rules

```
   itemFrequencyPlot(transaction_dataset, support = 0.1,
cex.names=0.8)
   > rules <- apriori(transaction_dataset,parameter =
list(support = 0.01, confidence = 0.6))
```

9. Summary information about obtained rules

```
   rules
   summary(rules)
```

10. Sampling rules for the positive and negative class of interest output variable, sort it, and defined number of necessary rules (n) for interest class value

```
   rulesNegative <- subset(rules, subset = rhs %in%
"output_variable=negative")
   rulesPositive <- subset(rules, subset = rhs %in% "
output_variable=positive")
```

```
   inspect(head(sort(rulesNegative, by = "confidence"), n =
3))
```

11. Pruning the number of candidates in compliance with the lift =1.

```
   rules <- apriori(birtMatrix, parameter=list(support=0.1,
confidence=0.5))
```

12. Save obtaining rules with values of support, confidence, lift to file.

13. Calculating interestingness of rules with respect to each other starting with the first rule sorted by lift.

13.1 Calculated on the basis of the comparison between the items of the two conjuncts as follows (4.1).

For example, the interpretation of the rules obtained let’s analyze rule 213 {Placental grading at 30 to 38 weeks = 1}= > {Diagnosis = neonatal hypoxia} and rule 250 {Placental grading at 30 to 38 weeks = 1} = > {Amniotic fluid index at 30 to 38 weeks = low}. These rules have identical attribute {Placental grading at 30 to 38 weeks} and the same values (V_{213}=1and V_{250}=1), therefore the degree of deviation is equal to 0, i.e. the comparable elements are identical.

The degree of deviation at the lowest level for the rule 213 {Placental grading at 30 to 38 weeks = 1} = > {Diagnosis = neonatal hypoxia} and rule 230 {Placental grading at 30 to 38 weeks = 3} = > {Diagnosis = normal} is equal to 1. In this case we have identical attribute {Placental grading at 30 to 38 weeks} and different values (V213=1andV230= 3). This means the comparable elements are different.

13.2 Calculated differences among adjusted attributes as follows (4.2).

For example, for the rule 283 {Erythrocyte speed rate at 21 week = high, Amniotic fluid index at 30 to 38 weeks = low} = > {Placental maturity degree at 30 to 38 weeks = second} and rule 284 {Erythrocyte speed rate at 21 week = high, Amniotic fluid index at 30 to 38 weeks = low} = > {Diagnosis = neonatal hypoxia} the number of elements is equal. The elements of rule 283 and their values belong to the set of elements and values of rule 284 and have a degree of deviation at the lowest level equal to 0. Thus, the degree of deviation between the comparable sets of elements on the average level is equal to 0, and, consequently, the set of elements of the conditions are identical.

For rule 297 {Amniotic fluid index at 30 to 38 weeks = low, Placenta thickness at 30 to 38 weeks = normal} = > {Diagnosis = neonatal hypoxia} and rule 173 {Amniotic fluid index at 30 to 38 weeks = normal, Placental maturity grade at 30 to 38 weeks = second} = > {Diagnosis = normal}, in the set of values of elements of the rule 297 there is no value belonging to the set of values of elements of the rule 173 with the degree of deviation at the lowest level = 1. In this case, the degree of deviation between the compared sets of elements on the average level = 1, and, therefore, the sets are different.

In case of non-compliance with the above conditions, for the rule 290 {Erythrocyte speed rate at 21 week = high, Erythrocyte speed rate at 30 week = high, Amniotic fluid index at 30 to 38 weeks = low} = > {Diagnosis = neonatal hypoxia} and rule 150 {Erythrocyte speed rate at 21 week = high, Amniotic fluid index at 30 to 38 weeks = low, placental grading at 30 to 38 weeks = second} = > {Prothrombin index = five}, with equal number of elements we can determine the quantitative estimation of the

deviation for the conditions of the rules at the median level. The degree of deviation at the lowest level for the values of these elements is 0, 0, 1, respectively. By (2) the average deviation for these rules calculated as follow

$$\frac{1}{3}(0,0,1)=0.33\cdot 0+0.33\cdot 0+0.33\cdot 1=0.33$$

The interestingness of the rules is computed in terms of a certain deviation at the lowest and average level.

13.3 Calculated interestingness for association rules is performing as follows (3).

For example, the estimation of interestingness for rule 286 {Erythrocyte speed rate at 21 week = high, Amniotic fluid index at 30 to 38 weeks = low, Placenta thickness at 30 to 38 weeks = normal} = > {Diagnosis = neonatal hypoxia} in relation to rule 290 {Erythrocyte speed rate at 21 week = high, Erythrocyte speed rate at 30 week = high, Amniotic fluid index at 30 to 38 weeks = low} = > {Diagnosis = neonatal hypoxia} can be performed as follows. The deviation of the elements of the rule's conditions at the lowest level is (0, 0, 1). The deviation at the average level is 0.33. The deviation of the effects of the rules at the lowest level is 0. The deviation of the effects of the rules at the average level is 0. Since the deviation of the conditions at the average level is 0.33 and means more than the deviation of the effects on the average level of 0, then the interestingness of rule 286 in relation to rule 290 is $I^{II}(286, 290) = (0.33 + 0)/2 = 0.165$.

14. Define threshold of the measure of interestingness equal to 0.2.

15. Using anti-monotone property with the threshold of the measure of interestingness and reducing the number of rules in compliance to it.

16. Interpret the result.

4.4 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, figure / scheme of testing environment, procedures
- (R): narrate (like a story), tables, indicate final results;
- (D): answers on discovery questions, explanation of anomalies, conclusion / summary

4.5 Test questions:

1. What are the algorithms used for associative data analysis?
2. Formulate the anti-monotonicity property.

4.6 Recommended literature:

1. Kaur, H., Wasan, S. K., Al-Hegami, A. S. and Bhatnagar, V.: A Unified Approach for Discovery of Interesting Association Rules in Medical Database. In Advances in Data Mining.

2. https://cran.r-project.org/web/packages/arules/arules.pdf

Reading list

1. A. Jaokar, "Data Science for Internet of Things (IoT): Ten Differences from Traditional Data Science" *Kdnuggets. com,* 2019. [Online]. Available: https://www.kdnuggets.com/2016/09/data-science-iot-10-differences.html [Accessed: 22 December 2018].
2. F. Chen, P. Deng, J. Wan, D. Zhang, A. Vasilakos and X. Rong, "Data Mining for the Internet of Things: Literature Review and Challenges", *International Journal of Distributed Sensor Networks*, vol. 11, no. 8, p. 431047, 2015. DOI: 10.1155/2015/431047.
3. A. Mukhopadhyay, U. Maulik, S. Bandyopadhyay, C. A. C. Coello, "A survey of multiobjective evolutionary algorithms for data mining: part I", *IEEE Transactions on Evolutionary Computation*, vol.18, no.1, pp. 4–19, 2014. DOI: 10.1109/TEVC.2013.2290086.
4. D. Lee, H. Lee, "IoT service classification and clustering for integration of IoT service platforms", *The Journal of Supercomputing*, vol.74, no.12, pp.1-17. DOI: 10.1007/s11227-018-2288-7.
5. C.-W. Tsai, C-F. Lai, M.-C. Chiang, L.T. Yang, "Data Mining for Internet of Things: A Survey". *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 77-97. DOI: 10.1109/SURV.2013.103013.00206.
6. M.S. Mahdavinejad, M. Rezvan, M. Barekatain, P. Adibi, P. Barnaghi, A.P. Sheth, "Machine learning for Internet of Things data analysis: A survey". *Digital Communications and Networks,* vol.4, no.3, pp.161-175. DOI: 10.1016/j.dcan.2017.10.002.
7. C. M. Bishop, "Pattern Recognition and Machine Learning", Springer, 2006. DOI: 10.1117/1.2819119.
8. A. Boukerche, S. Samarah, "Novel Algorithm for Mining Association Rules in Wireless Ad-hoc Sensor Networks", *IEEE Transactions on Parallel and Distributed Systems*, vol.19, no.7, pp. 865-877. DOI: 10.1109/TPDS.2007.70789.
9. S.K. Tanbeer, C.F. Ahmed and B.S. Jeong, "An Efficient SinglePass Algorithm for Mining Association Rules from Wireless Sensor Networks", *IETE Technical Review*, vol. 26, no.4, pp. 280-289. DOI: 10.4103/0256-4602.52997.
10. M. Rashid, I. Gondal, and J. Kamruzzaman, "Mining associated patterns from wireless sensor networks", *IEEE Transaction on*

Computers, vol. 64, no. 7, pp. 1998–2011, 2014. DOI: 10.1109/TC.2014.2349515.

11. H. Kaur, S.K. Wasan, A.S. Al-Hegami, V. Bhatnagar, "A unified approach for discovery of interesting association rules in medical databases". *In: Perner, P. (ed.) ICDM 2006. LNCS*, July 2016 vol. 4065, pp. 53–63. doi:10.1007/11790853_5.

12. A.Shi-Nash, D.R. Hardoon, "Data analytics and predictive analytics in the era of big data", *Internet of Things and Data Analytics Handbook*, pp.329-345. DOI: 10.1002/9781119173601.ch19.

13. H. Cai, B. Xu, L. Jiang, A. V. Vasilakos, "IoT-based Big Data Storage Systems in Cloud Computing: Perspectives and Challenges", *IEEE Internet of Things Journal*, vol.1, no.4, pp. 1–1. doi:10.1109/jiot.2016.2619369

14. M. van Leeuwen, J. Vreeken, "Mining and Using Sets of Patterns through Compression", In Frequent Pattern Mining, Aggarwal, C. & J. Han, Springer, 2014. pp. 165-198.

15. I. Skarga-Bandurova, T. Biloborodova, M. Nesterov, "Extracting Interesting Rules from Gestation Course Data for Early Diagnosis of Neonatal Hypoxia", *Journal of medical systems*, vol. 43, no.1, p.8.

Deep Learning for IoT

A. O. Sachenko and V. S. Koval

Seminar 1
DATA PREPARATION FOR DEEP NEURAL NETWORK

Goal and objectives: Obtaining practical skills in data preparation for Deep Neural Networks application using the Matlab

Learning objectives:

– studying the principles of Datastore;

– using Matlab for creating of training and test sets for Deep Neural Networks;

Practical tasks:

– obtaining experience working with Matlab image sets;

– getting practical skills in working with parameters of imageDatastore command;

– obtaining of practical skills in working with image processing to create imageDatastore;

– formation of training vectors for Deep Neural Network;

– obtaining practical skills in ImageDatastore for image classification task.

Exploring tasks:

– discover hosts on the closed network;

– investigate how to restrict the application scans to specific sets of port numbers.

Setting up

In preparation for laboratory work it is necessary:

– to clear the goals and mission of the research;

– to study theoretical material contained in this manual, and in references;

– to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

– computers with OS Windows;

– Matlab software (version 2018b, RAM - not less than 4Gb);

– access to files at *https://github.com/vkoukr/ERASMUS-ALIOT-2019/blob/master/1_Seminar_Appendix.zip*;

– The AlexNet network model for image classification that should be installed in Matlab toolbox;

- skills from the basics of programming in Matlab environment;

1.1 Synopsis

In this seminar you will obtain experience of working with and understanding the Datastores in Matlab. These practical skills are very impotant for implementation in Deep neural networks for the developed applications and executing the following laboratory work during studding the module of Deep Neural Network.

1.2 Brief theoretical information

1.2.1 What is a Datastore?

A datastore is an object for reading a single file or a collection of files or data. The datastore acts as a repository for data that has the same structure and formatting [1]. For example, each file in a datastore must contain data of the same type (such as numeric or text) appearing in the same order, and separated by the same delimiter.

Table 1.1 – The list of MATLAB Datastores

Type of File or Data	**Datastore Type**
Text files containing column-oriented data, including CSV files.	TabularTextDatastore
Image files, including formats that are supported by imread such as JPEG and PNG.	ImageDatastore
Spreadsheet files with a supported Excel® format such as .xlsx.	SpreadsheetDatastore
Key-value pair data that are inputs to or outputs of mapreduce.	KeyValueDatastore
Custom file formats. Requires a provided function for reading data.	FileDatastore
Datastore for checkpointing tall arrays.	TallDatastore

1.2.2 Content Image Datastore

Assume that there is a database of images of some objects that need to be used to train by the neural network (Fig. 1.1). The image

Datastore contain the reference to the list of files of files. It could be used for creating the set of datas for Deep Neural Network applications.

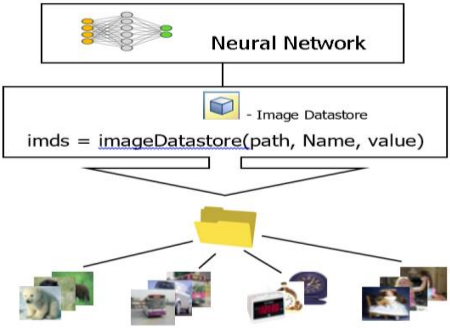


Fig. 1.1 – The content of image Datastore

1.2.3 Creating an object of imageDatastore

To begin, you need to create an object as imageDatastore. This object contains links and parameters of all image files that are part of it. This data organization allows you to manipulate a plurality of image files (for example, for DNN), and not just work with one. To create the Datastore object in the Matlab environment, you must execute the following command:

```
imds = imageDatastore(location,Name,Value)
```

The above command creates a datastore imds from the collection of image data specified by location and additional parameters and properties for imds using one or more name-value pair arguments.

There are three parameters: «location», «name», «value». Let consider in more details.

Location

location – represent files or folders to include in datastore. The type of location is character vector or cell array of character vectors or string scalar or string array

If the files are not in the current folder, then location must be full or relative paths. You can use the wildcard character (*) when specifying location. This character indicates that all matching files or all files in the matching folders are included in the datastore.

IMPORTANT. Files within subfolders of the specified folder are not automatically included in the datastore.

When location represents a folder, the datastore includes only supported image file formats and ignores any other format. Supported files have an imformatsformat.

Example: 'file1.jpg'
Example: '../dir/data/file1.png'
Example: {'C:\dir\data\file1.tif','C:\dir\data\file2.tif'}
Example: 'C:\dir\data\*.jpg'

Name-Value Pair Arguments

Specify optional comma-separated pairs of Name,Value arguments. Name is the argument name and Value is the corresponding value. Name must appear inside quotes. You can specify several name and value pair arguments in any order as Name1,Value1,...,NameN,ValueN. The list of possible Name-Value arguments represents in table 1.2.

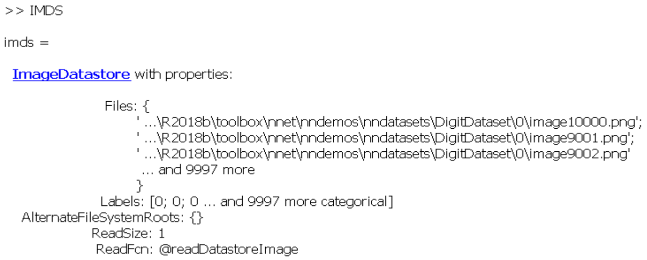
Table 1.2 – Name-Vlaue arguments of imageDatastore

Name	Value
'IncludeSubfolders' — Subfolder inclusion flag	true or false | 0 or 1
'FileExtensions' — Image file extensions character	vector | cell array of character vectors | string scalar | string array
'AlternateFileSystemRoots' — Alternate file system root paths	string vector | cell array
'LabelSource' — Source providing label data	'none' (default) | 'foldernames'

Example:

```
imds=imageDatastore('c:\MATLAB\R2018b\toolbox\nnet\nndemos\nndatasets\', ...
    'FileExtensions',{'.jpg','.tif'},...
    'LabelSource', 'foldernames', ...
    'IncludeSubfolders', 'true ')
```

The result of executing the command for creating image Datasore in Matlab is shown on Fig. 1.2.



```
>> IMDS

imds =

  ImageDatastore with properties:

                       Files: {
                              ' ...\R2018b\toolbox\nnet\nndemos\nndatasets\DigitDataset\0\image10000.png';
                              ' ...\R2018b\toolbox\nnet\nndemos\nndatasets\DigitDataset\0\image9001.png';
                              ' ...\R2018b\toolbox\nnet\nndemos\nndatasets\DigitDataset\0\image9002.png'
                               ... and 9997 more
                              }
                      Labels: [0; 0; 0 ... and 9997 more categorical]
    AlternateFileSystemRoots: {}
                    ReadSize: 1
                     ReadFcn: @readDatastoreImage
```

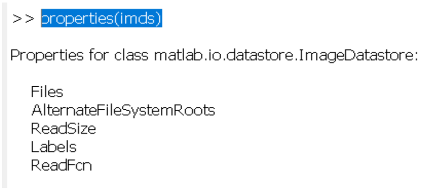
Fig.1.2 – The screenshort of creating object like image Datastore

1.2.4 Properties of imageDatastore

ImageDatastore properties describe the data and specify how to read the data from the datastore. In order to view the properties of imageDatastore, let perform the following command:

```
properties(imds)
```

The result of the above command execution is shown in Fig.1.3.



```
>> properties(imds)

Properties for class matlab.io.datastore.ImageDatastore:

    Files
    AlternateFileSystemRoots
    ReadSize
    Labels
    ReadFcn
```

Fig.1.3 – The property of image Datastore

It is possible to specify the value of ImageDatastore properties using name-value pair arguments when you create the datastore object. To view or modify a property after creating the object, use the dot notation.

For example, you can create an ImageDatastore object and specify the 'ReadFcn' parameter:

imds = imageDatastore('peppers.png','ReadFcn',@customreader);

Alternatively, you can assign 'ReadFcn' to @customreader after you create the ImageDatastore:

```
imds = imageDatastore('peppers.png');
imds.ReadFcn = @customreader;
```

Table 1.3 represents properties of imageDatastore. Let try to look at the filenames and relative lables that were accepted after creating the imds ImageDatastore by using the following command:

```
imds.Files
imds.Labels
```

Table 1.3 – Properties of imageDatastore

Name	Value
Files — list of files included in datastore	character vector | cell array of character vectors | string scalar | string array
ReadSize — Number of image files to read. Number of image files to read in a call to the read function, specified as a positive integer scalar. Each call to the read function reads at most ReadSize images	1 (default) | positive integer scalar
Labels — File labels for the files in the datastore, specified as a vector, a cell array, or a string array. The order of the labels in the array corresponds to the order of the associated files in the datastore. If you specify 'LabelSource','foldernames' when creating the ImageDatastore object, then the label name for a file is the name of the folder containing it. If you do not specify 'LabelSource','foldernames', then Labels is an empty cell array or string array. If you change the Files property after the datastore is created, then the Labels property is not automatically updated to incorporate the added files	categorical, logical, or numeric vector | cell array | string array
ReadFcn — Function that reads image data.	@readDatastoreImage

Function that reads image data, specified as a function handle. The function must take an image file name as input, and then it outputs the corresponding image data. For example, if customreader is the specified function to read the image data, then it must have a signature similar to this: function data = customreader(filename) ... End If more than one output argument exists, then imageDatastore uses only the first argument and ignores the rest.	(default) | function handle

1.2.5 Methods of imageDatastore

The list of methods (functions) that could be used for processing the data in the imageFatasore is shown below in table 1.4:

Table 1.4 – Functions of imageDatastore

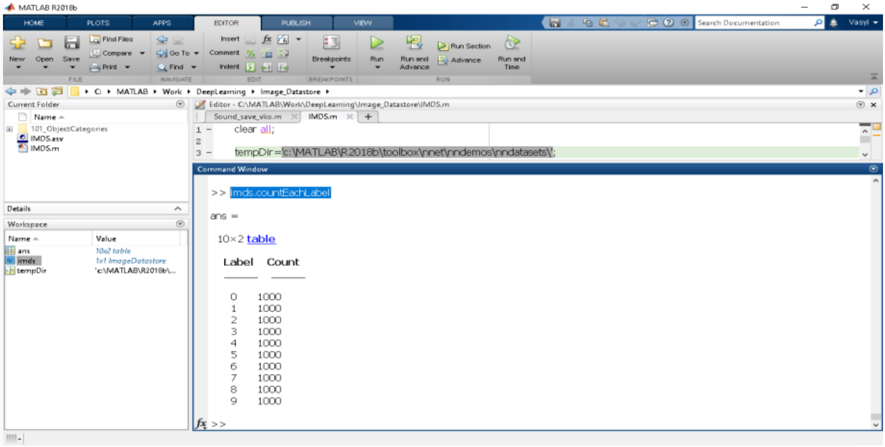
Command	Description
countEachLabel	Count files in ImageDatastore labels
hasdata	Determine if data is available to read
partition	Partition a datastore
numpartitions	Number of datastore partitions
preview	Subset of data in datastore
read	Read data in datastore
readall	Read all data in datastore
readimage	Read specified image from datastore
reset	Reset datastore to initial state
shuffle	Shuffle files in ImageDatastore
splitEachLabel	Split ImageDatastore labels by proportions

1.2.5.1 counteachlabel

T = countEachLabel(imds) – returns a summary table of the labels in imds and the number of files associated with each. For example, in order to look for number of file images per each label let execute the following code:

```
imds.countEachLabel
```

The result is presented on Fig.1.4. As you cans see there exsit 1000 images per each of 10 labels.



Name	Value
ans	10x2 table
imds	1x1 ImageDatastore
tempDir	'c:\MATLAB\R2018b\...

```
clear all;

tempDir='c:\MATLAB\R2018b\toolbox\nnet\nndemos\nndatasets\';
```

```
>> imds.countEachLabel

ans =

  10×2 table

    Label    Count
    _____    _____

      0      1000
      1      1000
      2      1000
      3      1000
      4      1000
      5      1000
      6      1000
      7      1000
      8      1000
      9      1000

fx >>
```

Fig.1.4 – Coun the number of images per label of imageDatastore

1.2.5.2 hasdata

tf = hasdata(imds) – returns logical 1 (true) if there is data available to read from the datastore specified by *imds*. Otherwise, it returns logical 0 (false).

Example:

```
while hasdata(imds)
  T = read(imds);
end
```

1.2.5.3 partition

subds = partition(ds,n,index) partitions datastore ds into the number of parts specified by n and returns the partition corresponding to the indexindex.

subds = partition(ds,'Files',index) partitions the datastore by files and returns the partition corresponding to the file of index index in the Filesproperty.

subds = partition(ds,'Files',filename) partitions the datastore by files and returns the partition corresponding to the file specified by filename.

For example, the below Matlab code provides partitioning the datastore by files and return the part corresponding to the indicated filename.

```
imds = imageDatastore({'up.jpg','down.png','left.tif','right.jpg'})
sub_imds = partition(imds,'Files','right.jpg')
```

Note: sub_imds contains one file. «right.jpg»

1.2.5.4 numpartitions

n = numpartitions(imds) returns the default number of partitions for datastore *imds*.

n = numpartitions(imds,pool) returns a number of partitions to parallelize datastore access over the parallel pool specified by pool. To parallelize datastore access, Parallel Computing Toolbox™ have to be installed.

By default, there is only one partition in ds because it contains only one small file.

1.2.5.5 preview

data = preview(imds) returns an array of integers corresponding to the first image.in datastore.

```
sub_imds=preview(imds);
imshow(sub_imds);
```

1.2.5.6 read

data = read(imds) returns data from a datastore. Subsequent calls to the read function continue reading from the endpoint of the previous call.

[data,info] = read(imds) also returns information about the extracted data in info, including metadata.

1.2.5.7 readall

data = readall(imds) returns all the data in the datastore specified by *imds*.

If all the data in the datastore does not fit in memory, then readall returns an error. All data in the datastore, returned as a table or cell array. For ImageDatastore bjects, data is a cell array where each element displays the size and data type of the corresponding image data. For example,[480x640x3 uint8] describes the size of uint8 data corresponding to a color image.

1.2.5.8 readimage

img = readimage(imds,I) reads the Ith image file from the datastore *imds* and returns the image data *img*. The size and data type of the *img* array depends on the image formats of the files in the datastore. The image formats supported by readimage function are those formats supported by imread. For more information on the supported formats, see imread.

[img,fileinfo] = readimage(imds,I) also returns a struct fileinfo that contains two file information fields:

- Filename — Name of the file from which the image is read.
- FileSize — Size of the file in bytes.

Example:

```
imds = imageDatastore({'street1.jpg','street2.jpg'});
img = readimage(imds,2);
imshow(img)
```

1.2.5.9 reset

reset(imds) resets the datastore specified by *imds* to the state where no data has been read from it. Resetting allows re-reading from the same datastore.

1.2.5.10 shuffle

imdsrand = shuffle(imds) returns an ImageDatastore object containing a random ordering of the files from *imds*. It create a new datastore containing the same files in random order.

1.2.5.11 splitEachLabel

[imds1,imds2] = splitEachLabel(imds,p) splits the image files in *imds* into two new datastores, *imds1* and *imds2*. The new datastore *imds1* contains the first *p* files from each label and *imds2* contains the remaining files from each label. *p* can be either a number between 0 and 1 indicating the percentage of the files from each label to assign to *imds1*, or an integer indicating the absolute number of files from each label to assign to *imds1*.

[imds1,...,imdsM] = splitEachLabel(imds,p1,...,pN) splits the datastore into N+1 new datastores. The first new datastore *imds1* contains the first *p1* files from each label, the next new datastore *imds2* contains the next *p2* files, and so on. If *p1,...,pN* represent numbers of files, then their sum must be no more than the number of files in the smallest label in the original datastore imds.

___ = *splitEachLabel(___,'randomized')* randomly assigns the specified proportion of files from each label to the new datastores.

___ = *splitEachLabel(___,Name,Value)* specifies the properties of the new datastores using one or more name-value pair arguments. For example, you can specify which labels to split with 'Include','labelname'.

The below example shows how to create two new datastores from the files in *imds* by randomly drawing from each label. The first datastore *imds40* contains 40% of random files. The second datastore *imds60* contains the remaining files.

```
[imds40, imds60] = splitEachLabel(imds,0.4,'randomized')
```

1.3 Tasks to practice

Below locates a list of practical tasks to practice working with image sets (imageDatastore). The presented tasks provide a list of questions that need to be solved on a computer in the Matlab environment for in-depth study of the material. The correct answers are also given for the accomplished tasks. The answers can be used to provide verification, and the best assimilation of the material. All tasks are grouped into the following categories:

– Basics of Image Datastores (Table 1.5);

– Processing Images in a Datastore and preparing inputs for deep neural network (Table 1.6).

Table 1.5 – Basics of Image Datastore

Tasks	**Answer in Matlab**
Create variable ***tempDir*** that include path 'CIFAR-100_part\' to image files. Folder 'CIFAR-100_part\' should be created before starting the labwork and include some images from CIFAR-100 dataset	tempDir='CIFAR-100_part\'
Create a datastore named ***imds*** that refers to the image files in the ***tempDir*** folder and *'DigitDataset*' subfolders.	imds = imageDatastore(fullfile(tempDir,''),... 'IncludeSubfolders',true)
Create a datastore named ***imds*** that refers to the image files with extentions ***"*.png"*** and foldernames labels for the files at the ***tempDir*** folder and *'DigitDataset*' subfolders.	imds = imageDatastore(fullfile(tempDir,'DigitDataset'),... 'IncludeSubfolders',true,'FileExtensions','.png','LabelSource','foldernames')
Display the list of properties of ***imds*** ImageDatasore	properties(imds)
Use the ***Files*** property of the datastore ***imds*** to extract the file names of the images. Store the result in a variable called ***fname_imds***.	fname_imds=imds.Files
Use the ***Labels*** property of the datastore ***imds*** to extract the label names of the images. Store the result in a variable called ***labels_imds***.	labels_imds=imds.Labels
Show the list of methods of ***imds*** imageDatastore using command ***methods()***	methods(imds)
Count and display the number of images per each labels of ***imds*** using ***counrEachLabel*** function	countEachLabel(imds)

Use the ***readimage*** function display the image of the 6th file in the datastore). Store the imported image in a variable called ***im_6***.	image(readimage(imds,6))
	im_6=readimage(imds,6)

Table 1.6 – Processing Images in a Datastore and preparing inputs for deep neural network

Tasks	Answer in Matlab
Use the ***size*** function to view the size of the image ***im_6***. Save the result to a variable ***size_im6***	size_im6=size(im_6)

Loaded ***AlexNet*** as the variable ***net***
Extract the ***InputSize*** property of the first layer of the network.
Store the result in a variable called input_sz
	net = alexnet;
	input_sz = net.Layers(1).InputSize;

Use the ***imresize*** function to resize the image stored in the variable ***im_6*** to be 227-by-227 by 1. Use one color channel of ***im_6*** for this Store the result back in the variable ***im_6_new***
Create variable ***im6_net*** as array with dimension in 227 x 227 x 3 basing on the ***in_6_new.*** The color channels of ***im6_net*** should be equal.
Display the resized ***im6_net*** image and original image ***im_6*** on the same figure
	im_6_new=imresize (im_6(:,:,1), [227 227]);
	im_6_net(:,:,1)=im_6_new;
	im_6_net(:,:,2)=im_6_new;
	im_6_net(:,:,3)=im_6_new;
	figure; subplot(1,2,1); image(im_6);
	subplot(1,2,2); image(im_6_net);

Use AlexNet (loaded as the variable net) to classify the contents of all the images in the data set. Store the results in a variable called prediction
Display what result AlexNet has predicted
	prediction=classify(net,im_6_net);
	disp(prediction)

Divide the imageDatastore ***imds*** into the training train_imds and testing test_imds datastores using ***splitEachLabel*** function. The ***test_imds*** should include only one randomized images for each label category	num_labels=imds.countEachLabel;
	[train_imds,test_imds]=splitEachLabel(imds,num_labels{1,2}-1,'randomized');

Create an image datastore from	augmented_imds =

imds that will resize the images to 227x227x3 using simple processing function ***augmentedImageDatastore([row colums channels],imds)***. Name the new datastore ***augmented_imds***. An ***augmentedImageDatastore*** function can perform simple preprocessing on an entire collection images	augmentedImageDatastore([224 224 3],test_imds,'ColorPreprocessing','gray2rgb');

Classify the images in ***augmented_imds*** using preloaded network in the variable ***net***. Save the predictions to a variable ***prediction***.
Display the images in testing ***test_imds*** datastore and show at the title what ***net*** predicted from variable ***prediction***
	prediction=classify(net,augmented_imds);
	
	numel=length(test_imds.Labels);
	for i=1:numel
	  subplot(1,numel,i);
	  imshow(readimage(test_imds,i));
	  title(sprintf('%s',prediction(i)));
	end

1.4 Test questions

1. What is the image Datastore?

2. What needs in image Datastore?

3. How to access files using image Datastore?

4. How to create a training set for artificial neural network using image Datastore?

5. How to divide the existen Datastore for training and validation subdatastores?

6. How to preview images from Datastore?

7. What parameters of image Datastore?

1.5 Recommended literature

1. "Getting Started with Datastore- MATLAB & Simulink", *Mathworks.com*, 2019. [Online]. Available: https://www.mathworks.com/help/matlab/import_export/what-is-a-datastore.html. [Accessed: 31- Jul- 2019].

2. I. Goodfellow, Y. Bengio, A. Courville, "Deep Learning",

2016, An MIT Press book, 787pp.

3. S. J. Chapman, "MATLAB Programming for Engineers", Published November 1st 2007 by Thomson Learning.

4. APA-style citation: "Warden P. Speech Commands: A public dataset for single-word speech recognition, 2017. Available from http://download.tensorflow.org/data/speech_commands_v0.01.tar.gz".

Laboratory work 2
RECOGNITION OF ALPHANUMERIC INFORMATION BY USING ARTIFICIAL NEURAL NETWORK

Goal and objectives: to obtain the practical experience of applying the artificial neural networks for pattern recognition

Learning objectives:

- studding and obtaining experience of Artificial Neural Network (ANN) using;
- using Matlab for Artificial Neural Network applications that could be used in IoT;

Practical tasks:

- obtaining experience working with Matlab image sets;
- obtaining practical skills in working with Artificial Neural Network architecture;
- obtaining of practical skills in creating training set for ANN;
- obtaining of practical skills in designing of ANN;

Exploring tasks:

- discover structure of ANN.
- investigate the ability of ANN to recognize Alphanumeric Information.
- investigate possibility to implement of ANN for IoT.

Setting up

In preparation for laboratory work it is necessary:

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual, and in references;
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

- computers with OS Windows;
- Matlab software environment;
- access to files at *https://github.com/vkoukr/ERASMUS-ALIOT-2019/blob/master/2_Lab%20work_ALPHANUMERIC_Appendix.zip*;
- installed MS Excel;
- the basic experience in Matlab programming;

2.1 Synopsis

In this laboratory work you will obtain experience of solwing the patterj recgnition problem by artificial neural network. You will study how to design and apply artificial neural network for the tasks that need to recognize alphanumerical information.

2.2 Brief theoretical information

2.2.1. The problem of pattern recognition

"How do we recognize the individual human face, when we see it in different positions: in profile, in three-quarters, or in full-face? How do we know the circle as such, regardless of whether it is large or small, close to it or in the distance? How do we see the faces of animals and geographic maps in the clouds? How do we translate words of birds cry or insects?

To answer these seemingly simple questions, it is necessary to know the processes that occur in the human brain in the perception of external phenomena. Until recently, the mechanisms of these processes seemed completely implicit. Therefore, it is quite understandable that interest caused the announcement of the creation of machines learning to recognize images. In addition to the great practical significance of such "recognizable" machines that can be used in diagnostic installations, for reading and translation, aerial photography, modeling processes of perception will allow to some extent to study the principles of pattern recognition and learning in living organisms.

An essential feature of the process of perception is the presence of an innate mechanism for the classification of certain groups of events or images in their most characteristic features.

Without the ability to classify the phenomena people would have to memorize many specific objects of the world.

Therefore, for reading books it would be necessary to study the fonts of all the printers, because each of them has it own different sets of letters. The production of medical diagnoses would be very complicated because there is almost no identical symptom of one disease.

In reality, not everything is so tragic. People can read the books that published in different places, can recognize handwriting text, predict the nature of the disease and recognize many objects that have been seen for the first time.

This is due to the fact that in the process of perception we learn to break all the phenomena of the surrounding world into groups that have distinctive features. All symbols shown on Fig. 2.1 may be called letter "A", despite them being writing significantly different from each other.

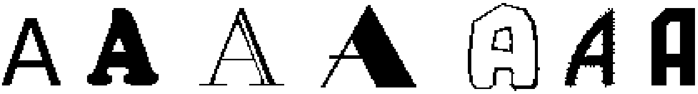


Fig. 2.1 – Examples for different presentation of the symbol “A”

The human brain has the ability to form some general representations that cover a circle of similar in appearance phenomena. Due to this, the process of perception in humans is based on past experience, when at the stage of learning when showing a limited number of objects formed the concept of an image, which includes a lot of similar objects, and which people “recognizes" in the future.

For example, a student who knows the letters will recognizes these letters in the text when the teacher showed him such symbols on the image before.

Thus, the purpose of learning is to successfully recognize the whole set of objects after familiarizing with its part.

Between the photoreceptors and the brain is included a complex neural network that performs some primary processing of visual information. In this network, the motion of the image is allocated through the transmission of signals only at the time of change of illumination and determination of contours of the image. In addition, "directly in the retina, there is a selection process for the simplest configurations contours - the curvature, the angles, the lines etc. This technique is congenital and is carried out by special neurons that associated with groups of receptors excited under certain configurations of the details of the image.

As a result of the researches of the human and animals visual systems, a large number of interesting devices were designed, which allow object classification basing on some simple features independently on their size, orientation in space or perspective changes.

Attempts to find out the mechanism of further processing of information associated with associative processes in the cerebral cortex, encountered significant difficulties.

Therefore, the main role in the study of the recognition of complex phenomena provides by various analytical methods where the data of biological experiments are used to assess the "truth" of the proposed models.

2.2.2. The structure of the perceptron

Using the knowledge about the structure of the elementary nerve cell - the neuron, it is possible to describe its functionality by mathematical equations in order to synthesize a complex network with a set of given properties.

Most interesting in this direction are the work of American scientists – Nicolas Rashevsky [**Error! Reference source not found.**] and later studied by Alan Lloyd Hodgkin [**Error! Reference source not found.**], in which the empirical properties of individual neurons are described by means of differential equations. In these equations, authors tend to take into account all the parameters of signal transmission through a neuron with sufficient accuracy: spatial and temporal summation, constant time, ion permeability, etc. However, adding the resulting expressions to information processes in multi-level networks is impossible due to the implicit complexity of the mathematical dependencies.

Another method is to formalize neural networks by representing of a neuron in the form of a boundary key circuit having two states - "on" or "off", and which works if the algebraic sum of the input signals exceeds the threshold. Input pulses can be either positive (excitatory) or negative (inhibitory). A group of such circuits creates a universal computing element that is used to construct a logical network that implements to any functions of algebra of logic.

For the first time such an approach to the simulation of complex systems was proposed by McCullock and Pitts for solving the logical problems, including learning and recognition. The application of this approach requires basically the writing of a truth table that establishes the connection between the input and output relationships. This requires prior knowledge of the expected behavior of the entire system being constructed, in response to each combination of input signals. It is clear that for complex systems, the compilation of such a table requires significant complexity.

For the first time, the principle of statistical organization was used by the American scientist Frank Rosenblatt to create a model of a recognizable machine that is learning, and received the perceptron title (from perception).

By the perceptron we will call any model of a neural network, which includes receptor, associative and reactive elements, and has some techniques that can affect the relationships between some of these elements.

The perceptron could be presented schematically (Fig. 2.2). The receptor device is presented by a matrix of photocells and is a simplified model of the receptor field of the retina. Images - letters, pictures, photos - are designed in this field, creating a configuration of light and dark places.

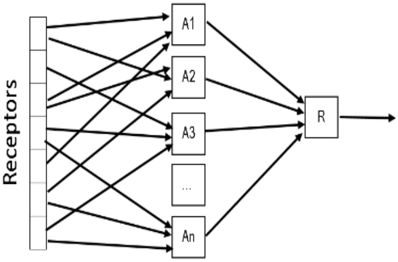


Fig. 2.2 – The structure of Perceptron

Each element in a matrix is a photosensitive sensor. The signal at the sensor output is either exists or absent, depending on whether the contour of the projected figure falls or not on this element.

The next element of the perceptron is a set of boundary elements that form the so-called associative field, in which the allocation of the main features of the image takes place. All elements of this field are the same and represent one of the varieties of the formal model of the neuron. Each such neuron (A-element) has one output and several inputs that receive signals from photoreceptors.

In the process of creating the perceptron, the inputs of each A-element are arbitrarily connected to the outputs of several receptors. In this case, each connection may transmit the positive or negative signal value, depending on the inclusion.

Thus, on each A-element there is a summation of signals from the connected sensors. In case if the input signal of the A-element exceeds the value of its own threshold (the same for all associative field neurons), then this A-element is excited and sends a signal to the special responsive device, (R-element). The inputs of R-element is connected with a defined group of A-elements. It shows, presence or absence of the desired properties of the image. The output signal is +1, if the algebraic sum of the input signals is positive, and -1 if this amount is negative. In the case of a zero signal, the reaction of perceptron is uncertain.

To train the perceptron, the functions of the "teacher" consist in changing of the neuron weights of the connections between the associative field and the reacting element to desired values. In the simplest case, one can imagine that the signals value from the excited A-elements is changed by means of amplifiers with variable gain coefficients, connecting the output of each A-element with the R-element.

Thus, into the R-element comes a total signal from the excited A-elements, each of which has the values of the own weight of the connection (the gain factor λ). The value of the R-element is determined by the following condition:

$$R = \begin{cases} +1, & if \quad \sum_{i=1}^{n} \lambda_i y_i > 0; \\ +1, & if \quad \sum_{i=1}^{n} \lambda_i y_i < 0; \end{cases},$$

where y_i is the output signal from the A-element, excited in the event that the algebraic sum of signals from photoreceptors exceeds its own threshold θ.

The described above perceptron includes only one R-element. It can recognize only two classes of patterns from the set of objects that has been shown to it. The perceptron output corresponds to two possible states of the R-element: +1 and -1.

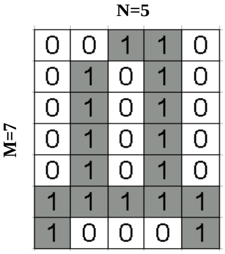
The images that belonging to one of two patterns sequentially are projected to the perceptron device. For example, different symbols of the letters "A" and "B". The training system, depending on the correct or wrong reaction, affects the weight of the connections. In such way iteratively found needed weights that provide generating of the output signal value, basing on the receipted input signal value.

In order the perceptron could recognize more numbers of images, such as all letters of the alphabet, you need to increase the number of R-elements.

2.3 Task definition

2.3.1 The problem of pattern recognition

The alphanumeric information is represented by a binary image with resolution MxN (Fig. 2.3).



0	0	1	1	0
0	1	0	1	0
0	1	0	1	0
0	1	0	1	0
0	1	0	1	0
1	1	1	1	1
1	0	0	0	1

Fig. 2.3 – Image of the alphanumeric character with dimension 7x5

The practical task consists in applying of the neural network to recognize the symbols from the noised image.

2.3.2 Selection of the neural network structure

To solve the above task, it will be used a three-level neural network that contains MxN neurons at the input layer, 10 neurons of the hidden layer and 10 neurons of the output layer. Such relation is formed basing on the generalized rule of constructing neural networks:

1. The neural network with one hidden layer allows transformation of any input signals to output values;
2. The number of neurons in the hidden layer of neural network should be less than the size of the training set;

3. The power of the neural network could be increased by increasing the number of neurons in a layer as well as increasing the number of layers in the neural network. When the neural network is restricted by applying of rule point 2 and it still can’t solve the task, that means it is necessary to increase the number of hidden layers of the neural network;

4. The random initialization of the weighting coefficients of the neural network should take place in a rather narrow range of values.

Based on the above rules, let match a specific alphanumeric symbol for each neuron of the output layer according to its position in the list. Then the output neuron, that has the maximum output value, will determine the corresponding alphanumeric character. Within this laboratory work, the size of the training sample is ten. The structure of the neural network is shown on Fig. 2.4.

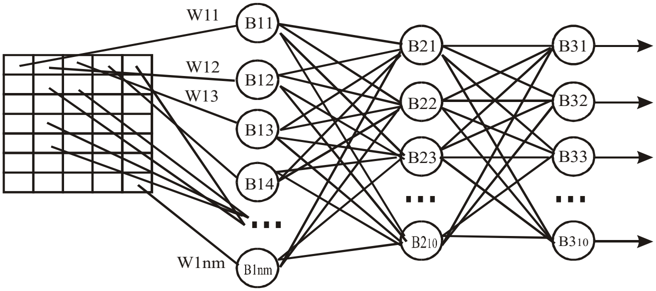


Fig. 2.4 – Structure of the neural network

2.4 Execution order and discovery questions

1. Read the presented above theoretical information.

2. Obtain a task from a lecturer according to the list of variants that is given at the end of this laboratory work.

3. Create a matrix of alphanumeric information as training set for the neural network (the number of characters and Cyrillic are selected according to the variant information) by following the next steps:

3.1. Copy the file "alph8x8.xls" or "alph7x5.xls" (hereinafter referred as "Alphabet.xls") depending on the variant information to your working directory.

3.2. Open the "Alphabet.xls" file using MS Excel. At the same time, leave the included macros as “active” in the file, which is intended to automate the process for creating of the binary input matrices.

3.3. Activate "Sheet1".

3.4. Fill the matrix of alphanumeric information by entering zeros and ones values into the corresponding cells (the one value is an indication that the corresponding pixel of the image has black color).

3.5. After filling the alphanumeric matrices, move the cursor to the position of the upper left pixel of each formed character. In MS Excel it is necessary to execute te the macro using the shortcut Alt +F8 (but it might vary in different version of MS Excel). At the end of the formation, save the file with matrices with the appropriate name.

3.6. Activate "Sheet2". In this case, the binary matrix will be displayed. The values from this matrix will be applied to the input of the neural network. Therefore, save the binary matrix to file as the format "CSV (MS-DOS) * .csv". To do this, select "Save as" in the MS Excel file menu, specify the file name and type: "CSV (MS-DOS) (* .csv)".

3.7. Edit the created * .csv file as text file MS-DOS (for example, using Total Commander or Notepad) by removing the first row that consists of ";" separators.

4. Present in the report (screenshot) btained at point 3.3 binary matrix as training set and inputs of the neural network.

5. Develop a software unit for training of the neural network based on the binary matrix created above and recognizing of the alphanumeric information. As a result, the following information should be added to the report:

- the structure of the crated neural network;
- the parameters for neural network training;
- training set in the format of matrices;
- the neural network training progress (for example, in Fig. 2.5).

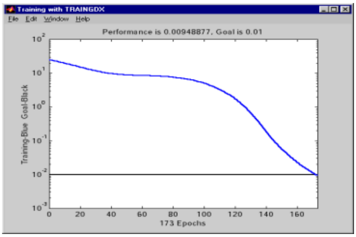


Fig. 2.5 – The result neural network training progress

6. Develop the software unit for noising of the symbol images (binary matrices) in a random way in the range 0-100% of the image pixels (for example, in Fig. 2.6).

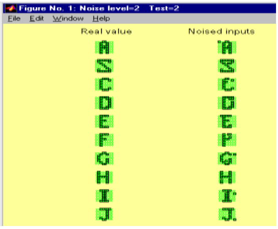


Fig. 2.6 – Noising of the neural network inputs

In this case, in new sets of noisy symbols, include several values (up to 10) for one level of noise. Represent one copy of the symbol images for one noise level to the report.

7. Develop a software unit for recognizing of incoming noisy images by the trained neural network (for example, Fig. 2.7). Provide displaying of the output results and include it into report.

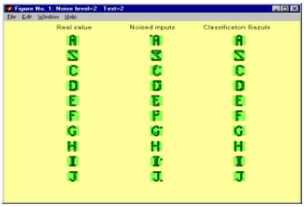


Fig. 2.7 – The result of the character recognition by the ANN

8. Explore the level of character recognition of alphanumeric information by the neural network, depending on the level of noise of the images (for example, Fig. 2.8). At the report it should be displayed a function of the wrong pattern recognitions by the neural network depending on the level of noise. Provide analysis of the obtained result at the report.

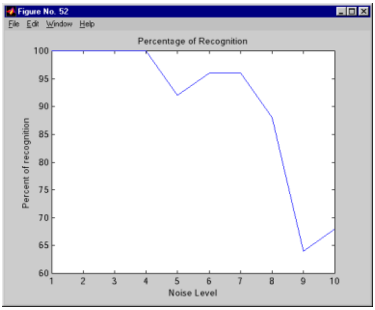


Fig. 2.8 – The result of image recognition by the neural network, in all experiments with the different level of noise.

8. Provide the report with the results and defend it.

2.5 Requirements to the content of the report

1. The subject and purpose of laboratory work.
2. The brief theoretical information.
3. Print out a program with comments.
4. The Print Screens with the results
5. Description of Print Screens obtained on point 4 with analysis.
6. Conclusions on the obtained results in laboratory work.

2.6 Test questions

1. What is the pattern recognition problem?
2. What is a perceptron?
3. What are the neural network training methods?
4. How to choose the neural network structure?
5. Main parameters of the neural network?
6. What should be structure of neural network for recognizing of the rotated image of symbols?

2.7 Recommended literature

1. "Nicolas Rashevsky", *En.wikipedia.org*, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Nicolas_Rashevsky. [Accessed: 31- Jul- 2019].

2. "Alan Hodgkin", *En.wikipedia.org*, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Alan_Lloyd_Hodgkin. [Accessed: 31- Jul- 2019].

3. "Manchester Mark 1", *En.wikipedia.org*, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Manchester_Mark_1. [Accessed: 31- Jul- 2019].

4. E. Alpaydin, "Introduction to Machine Learning, Third Edition", 2014 Massachusetts Institute of Technology. – 640 pp.

5. I. Goodfellow, Y. Bengio, A. Courville, "Deep Learning", 2016, An MIT Press book, 787pp.

6. Stephen J. Chapman, "MATLAB Programming for Engineers", Published November 1st 2007 by Thomson Learning.

2.8 The variants for laboratory work

Variant	Chars	Input Symbols	Neural Network Structure	Training parameters		Symbol size	Noise level
				Training error	Epoches		
1	Ukr	1-10	35-5-10	0,01	5000	7x5	1:10
2	Eng	1-10	35-6-10	0,1	5000	7x5	1:10
3	Digitals	0-9	35-10-10	0,001	5000	7x5	1:10
4	Ukr	1-10	35-5-10	0,0001	100	7x5	1:10
5	Ukr	11-20	35-8-10	0,1	5000	7x5	1:10
6	Eng	10-19	35-9-10	0,001	5000	7x5	1:10
7	Digitals	10-20	35-5-10	0,1	5000	7x5	1:10
8	Ukr	11-20	35-5-10	0,1	5000	7x5	1:10
9	Ukr	21-30	35-6-10	0,01	5000	7x5	1:10
10	Eng	20-26	35-8-10	0,0001	5000	7x5	1:10
11	Digitals	20-29	35-5-10	0,001	5000	7x5	1:10
12	Ukr	21-30	35-10-10	0,1	5000	7x5	1:10
13	Eng	1-10	64-10-10	0,01	5000	8x8	1:10
14	Eng	10-19	64-5-10	0,001	5000	8x8	1:10
15	Digitals	10-20	64-5-10	0,00001	5000	8x8	1:10
16	Eng	11-20	64-6-10	0,1	5000	8x8	1:10
17	Ukr	21-30	64-10-10	0,01	5000	8x8	1:10
18	Eng	10-19	64-5-10	0,01	5000	8x8	1:10
19	Digitals	10-20	64-10-10	0,001	5000	8x8	1:10
20	Eng	11-20	64-9-10	0,0001	5000	8x8	1:10
21	Ukr	21-30	64-5-10	0,1	5000	8x8	1:10
22	Eng	10-19	64-5-10	0,001	5000	8x8	1:10
23	Digitals	10-20	64-6-10	0,1	5000	8x8	1:10
24	Eng	11-20	64-10-10	0,01	5000	8x8	1:10
25	Ukr	21-30	10-5-10	0,0001	5000	7x5	1:10
26	Eng	10-19	64-5-10	0,001	5000	8x8	1:10
27	Digitals	10-20	35-11-10	0,1	5000	7x5	1:10
28	Eng	11-20	64-5-10	0,001	5000	8x8	1:10
29	Ukr	2-11	64-7-10	0,01	5000	8x8	1:10
30	Eng	5-15	35-6-10	0,1	5000	7x5	1:10
31	Digitals	5-15	64-10-10	0,001	5000	8x8	1:10
32	Eng	15-25	64-5-10	0,0001	5000	8x8	1:10

Laboratory work 3
IMAGE CLASSIFICATION FOR IoT DEVICES BY USING DEEP NEURAL NETWORK

Goal and objectives: obtaining practical skills in Deep Neural Networks architecture design for Image classification problems at the Matlab environment

Learning objectives:

– studying the principles of Deep Neural Network (DNN);

– studying the ways for creating the training sets of DNN;

– studying of DNN architectures for image classification task;

– studying a method for image classification using DNN;

– studying the principles of IoT.

Practical tasks:

– designing of DNN convolutional architectures in Matlab environment;

– researching the quality of image classification by DNN in Matlab;

Exploring tasks:

– discover classification possibilities of DNN;

– investigate the behaviors of different DNN architectures forimage classifications.

– investigate structure of DNN.

Setting up

In preparation for laboratory work it is necessary:

– to clear the goals and mission of the research;

– to study theoretical material contained in this manual, and in references;

– to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

– Matlab software (version 2017b or above, RAM - not less than 4Gb);

– for increasing the training efficiency of DNN, it is suggested (not required for executing the laboratory work) to use the following approaches:

a) GPU-based approach. The CUDA technology (ver. 8 or higher) should be used for NVIDIA GPU and Deep Learning Primitives (cuDNN)**:** High-performance building blocks for DNN applications including convolutions, activation functions, and tensor transformations;

b) CPU-based approach. Installing Matlab Parallel Processing Toolbox for data processing on multiprocessors platforms or computer cluster organization using Matlab;

c) Cloud-based computational approach. The subscription and license for Internet Cloud is necessary for Matlab (for example: Microsoft Azure, Amazon AWS, Thingspeak or other);

– programming experience in Matlab;

3.1 Synopsis

In this laboratory work you will investigate classification possibilities of Deep Neural Networok. You will obtain experience in designing of the architectures of DNN for image classification task. Such task is very popular in IoT devices and has practical and scientific importance.

3.2 Brief theoretical information

Internet of Things (IoT) is a network concept consisting of interconnected physical devices with built-in sensors and software. It allows the transmission of data between the physical world and computer systems, using standard communication protocols. In addition to sensors, the network may have actuators built into physical objects and interconnected via wired or wireless networks. These interconnected devices have the ability to eliminate the need for human participation through the intelligent interfaces [**Error! Reference source not found.-Error! Reference source not found.**].

Concepts of IoT

The concept of IoT is the ability to connect different objects (things) for using by persons in everyday life. For example, a refrigerator, air conditioner, a car, a bike and even sneakers. All these objects (things) must be equipped with built-in sensors or sensors that have the ability to process information from the environment, exchange it and perform various actions depending on the received information. An example of implementing such a concept is the "smart house" or "smart farm" system [**Error! Reference source not found.**].

For practical implementation, all surrounding objects and devices must be provided with miniature identifying and sensory devices [**Error! Reference source not found.**, **Error! Reference source not found.**, **Error! Reference source not found.**]. Then, it will be possible to control and track these objects and their parameters in space and time as well as introduce information. A network channels are necessary for communication with them. In the general form from the Internet of things can be written in the form of the following symbolic formula:

IoT = Sensors + Data + Network + Services.

3.3 Task definition

3.3.1 Task description

Pattern recognition is one of the most fundamental problems in the theory of intelligent systems. On the other hand, the task of pattern recognition has a huge practical value.

The "classification" term is often used instead of the "recognition". These two terms are in many cases considered synonyms, but are not completely interchangeable. Each of them has its own application areas, and the interpretation of both terms often depends on the specifics of a particular task.

Here are some typical statements of recognition tasks.

1. The **identification task**, which is used to identify a particular object among its similar (for example, to identify wife among the other people).

2. **The classification task** is assignment of an object to a particular class. For example, the task of recognizing objects or deciding whether

to have a defect in some technical detail. Assignment an object to a particular class reflects the most common classification problem.

3. **Cluster analysis**, consists in the division of a given set of objects into classes - groups of objects that are similar among themselves by one or another criterion. This task is often called a classification without a teacher, because, unlike the previous task 2, the priori classes are not given.

Let consider the applications that are closest and similar in content for research (prototypes) among the set of tasks that are solved using IoT:

a) online search for the product properties by its image during purchases;

b) security systems using the recognizing of people or other objects in video stream;

c) searching people in the video stream by face image (at the airport, train station, etc.);

d) the conveyor lines that transporting various objects depending on the classified object on images;

e) etc.

The following problem statement is considered within the IoT for all of the above-mentioned application areas (Fig. 3.1).

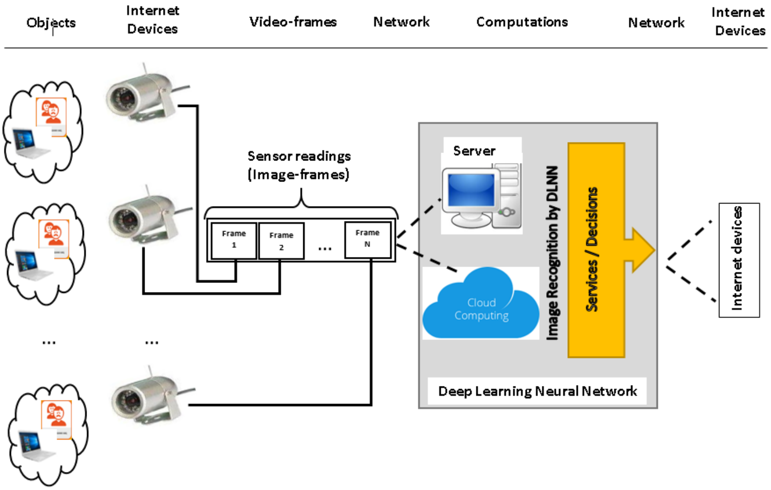


Fig. 3.1 – Task description

It is considered a technical system with Internet devices (video cameras) for the classification of video images. The video camera observes objects and generate stream of video frames. Through the communication channels video frames are sent for processing into a network cloud, or to a data processing server. After this, a decision about IoT service is made by using DNN image classification techniques (or the implementation of the DNN service in the cloud). Such services through network communication channels (Internet) are provided to users or IoT devices.

Thus, it is important to implement the IoT technology to ensure a qualitative classification of images. According to the purpose, the subject of the study is image classification methods using DNN.

3.3.2 Deep Neural Network principles

The artificial neural network represents by list of neurons distributed at the list of layers (Fig. 3.2). About the Deep Neural Network talks when the artificial neural network that has more than three layers. One of the DNN type is Convolutional Neural Network (Fig. 3.3). It consists the feature detection and classification layers. The feature detection layers provide convolution, rectification and pooling processing operation (Fig.3.4). The structure of classification layers is the same as structure of feedforward neural network. It includes fully connected layers to support classification (Fig. 3.4).

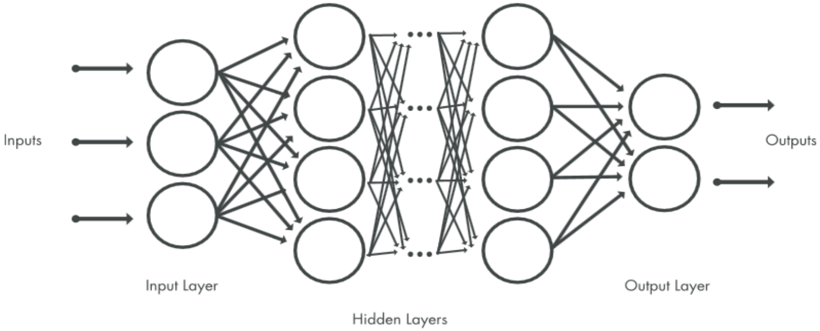


Fig. 3.2 – Artificial Neural Network Structure

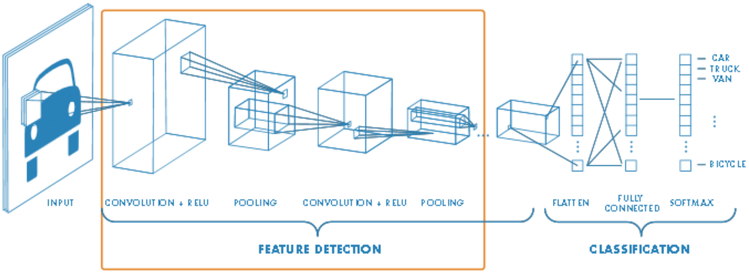


Fig. 3.3 – Convolutional Neural Network

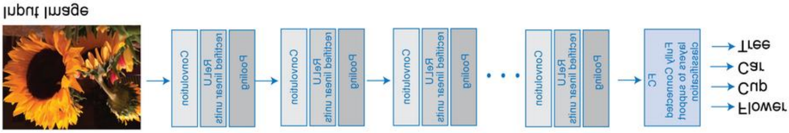

Input image
Convolution
ReLU rectified linear units
Pooling
FC Fully Connected layers of output classification
Tree
Car
Cup
Flower

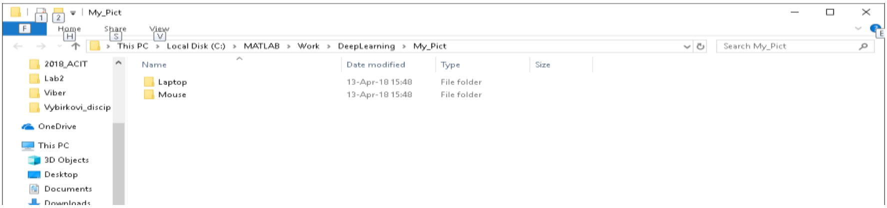
Fig. 3.4 – Convolutional Neural Network units

3.4 Execution order and discovery questions

1. Organization of input data

For input data organization, it is necessary to ensure that at least 50 images are taken for each of the classification objects. The image files of objects must be stored in separate directories.

For example, there are two categories of objects (Laptop, Mouse) stored in the *My_Pict* directory (fig. 3.5). A set of files for each of the specified object categories is also grouped and stored in separate directories with a generic category name.


My_Pict

This PC › Local Disk (C:) › MATLAB › Work › DeepLearning › My_Pict

Search My_Pict

Name	Date modified	Type	Size
Laptop	13-Apr-18 15:48	File folder	
Mouse	13-Apr-18 15:48	File folder	

Fig. 3.5 – The folder with image patterns

2. Designing of DNN structure

The special structure of artificial neural network is necessary to use for DNN application. To perform laboratory work, it is necessary to investigate two structures of the neural network:

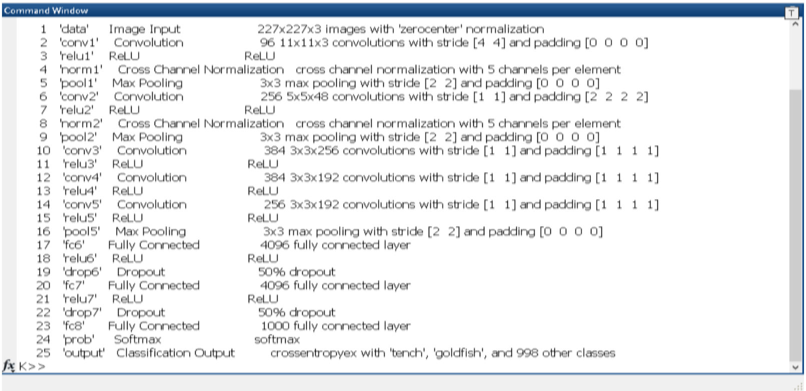
– modified AlexNet;

– personal.

2.1. To download pre-trained AlexNet neural network you can use the following code of Matlab:

```
%% Load a pre-trained, deep, convolutional network
alex = alexnet;
layers  Alex = alex.Layers
```

After executing of the above code, the architecture of the neural network will be displayed (Fig. 3.6). It consist from of 25 layers.



```
 1 'data'    Image Input              227x227x3 images with 'zerocenter' normalization
 2 'conv1'    Convolution              96 11x11x3 convolutions with stride [4  4] and padding [0  0  0  0]
 3 'relu1'   ReLU                     ReLU
 4 'norm1'    Cross Channel Normalization   cross channel normalization with 5 channels per element
 5 'pool1'   Max Pooling              3x3 max pooling with stride [2  2] and padding [0  0  0  0]
 6 'conv2'    Convolution              256 5x5x48 convolutions with stride [1  1] and padding [2  2  2  2]
 7 'relu2'   ReLU                     ReLU
 8 'norm2'    Cross Channel Normalization   cross channel normalization with 5 channels per element
 9 'pool2'   Max Pooling              3x3 max pooling with stride [2  2] and padding [0  0  0  0]
10 'conv3'    Convolution              384 3x3x256 convolutions with stride [1  1] and padding [1  1  1  1]
11 'relu3'   ReLU                     ReLU
12 'conv4'    Convolution              384 3x3x192 convolutions with stride [1  1] and padding [1  1  1  1]
13 'relu4'   ReLU                     ReLU
14 'conv5'    Convolution              256 3x3x192 convolutions with stride [1  1] and padding [1  1  1  1]
15 'relu5'   ReLU                     ReLU
16 'pool5'    Max Pooling              3x3 max pooling with stride [2  2] and padding [0  0  0  0]
17 'fc6'     Fully Connected          4096 fully connected layer
18 'relu6'   ReLU                     ReLU
19 'drop6'    Dropout                  50% dropout
20 'fc7'     Fully Connected          4096 fully connected layer
21 'relu7'   ReLU                     ReLU
22 'drop7'    Dropout                  50% dropout
23 'fc8'     Fully Connected          1000 fully connected layer
24 'prob'    Softmax                  softmax
25 'output'   Classification Output      crossentropyex with 'tench', 'goldfish', and 998 other classes
fx K>>
```

Fig. 3.6 – AlexNet structure

In order to reduce the computational consumption of the training procedures for the AlexNet neural network, it is advisable to modify only the last, classification layers, and leave the rest unchanged.

For example, the modification of 23 and 25 layers of AlexNet is provided by the following Matlab code.

```
%% Modify the network to use two categories
layers_Alex(23) = fullyConnectedLayer(2);
layers_Alex(25) = classificationLayer;
```

2.2. To create your own DNN structure, you need to create a data structure with a list of required types of DNN layers and their parameters in the Matlab environment. For example, the following Matlab code is used to create the proposed on Fig. 3.7 DNN as example:

```
%%---My Network
layers_my=[imageInputLayer([227 227 3]),
convolution2dLayer(5,20), reluLayer,
maxPooling2dLayer(2,'Stride',2), fullyConnectedLayer(2),
softmaxLayer,classificationLayer]
```

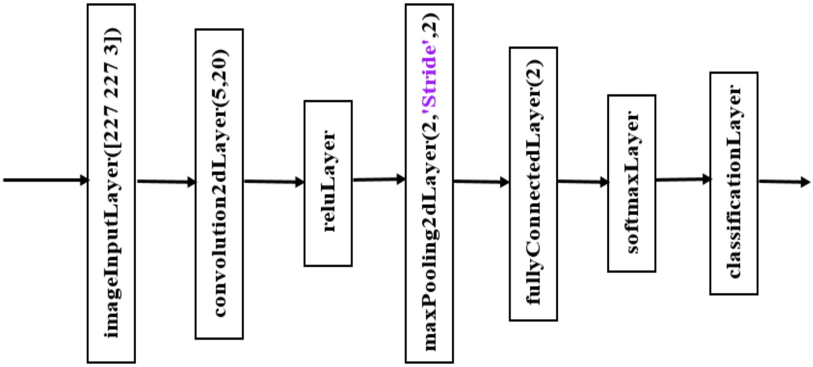
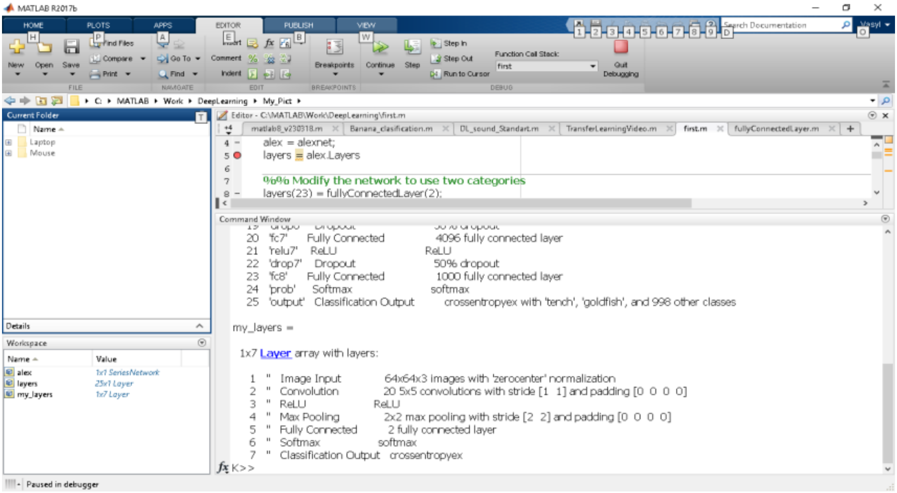


Fig. 3.7 – The proposed DNN architecture for image recognition

As the result obtained DNN in Matlab is shown on Fig. 3.8.



Editor - C:\MATLAB\Work\DeepLearning\first.m

matlab8_v230318.m | Banana_clasification.m | DL_sound_Standart.m | TransferLearningVideo.m | first.m | fullyConnectedLayer.m

```
alex = alexnet;
layers = alex.Layers

%%% Modify the network to use two categories
layers(23) = fullyConnectedLayer(2);
```

Command Window

```
   20  'fc7'     Fully Connected           4096 fully connected layer
   21  'relu7'    ReLU                    ReLU
   22  'drop7'    Dropout                  50% dropout
   23  'fc8'     Fully Connected           1000 fully connected layer
   24  'prob'     Softmax                  softmax
   25  'output'   Classification Output      crossentropyex with 'tench', 'goldfish', and 998 other classes

my_layers =

  1x7 Layer array with layers:

    1   ''   Image Input            64x64x3 images with 'zerocenter' normalization
    2   ''   Convolution            20 5x5 convolutions with stride [1  1] and padding [0  0  0  0]
    3   ''   ReLU                 ReLU
    4   ''   Max Pooling            2x2 max pooling with stride [2  2] and padding [0  0  0  0]
    5   ''   Fully Connected         2 fully connected layer
    6   ''   Softmax               softmax
    7   ''   Classification Output   crossentropyex
fx K>>
```

Name	Value
alex	1x1 SeriesNetwork
layers	25x1 Layer
my_layers	1x7 Layer

Fig. 3.8 – The proposed DNN structure

3. Preparing of DNN training set

To prepare training set for neural network, it is necessary to use the *imageDatastore* tool in Matlab. The *imds* object creates in Matlab for this purpose. It specifies the path to the files executed in step 1. Matlab operators that create the *imageDatastore* are listed below.

```
%% Data preparation and Pre-processing
imageSize_Alex = alex.Layers(1).InputSize(1:2);
imageSize_my = layers_my(1).InputSize(1:2);



 path='C:\MATLAB\Work\DeepLearning\My_Pict';
 imds_Alex=imageDatastore(path, 'IncludeSubfolders', true,
 'LabelSource', 'foldernames');
 imds_Alex.ReadFcn =
 @(path)imresize(imread(path),imageSize_Alex);
 [TrainingImages_Alex, TestImages_Alex] =
 imds_Alex.splitEachLabel(18, 'randomize');


imds_my=imageDatastore(path, 'IncludeSubfolders', true,
'LabelSource', 'foldernames');
imds_my.ReadFcn = @(path)imresize(imread(path),imageSize_my);
[TrainingImages_my, TestImages_my] =
imds_my.splitEachLabel(18, 'randomize');
```

In this case, all input images are divided into two parts: training (*TrainingImages*) and those that will take part in the test (*TestImages*).

After creating the *ImageDatastore* in Matlab, let study the set of other properties and methods of the specified object by executing the following commands. Let determine the number of images per category

```
methods(imds_Alex); properties(imds_Alex);
```

To display a random image example for each category, let follow these steps:

```
figure
subplot(1,2,1);
imshow(imds_Alex.readimage(find(imds_Alex.Labels =='Laptop',
1)))
subplot(1,2,2);
imshow(imds_Alex.readimage(find(imds_Alex.Labels =='Mouse',
1)))
```

The result of displaying the random image per each category is shown on Fig. 3.9.

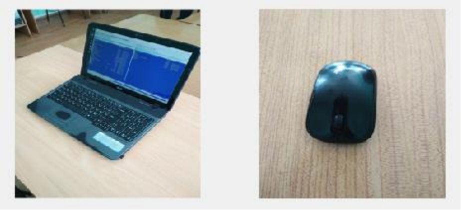


Fig. 3.9 – One image of each category

4. Training DNN

The parameters for training of DNN could be set by using *trainingOptions*

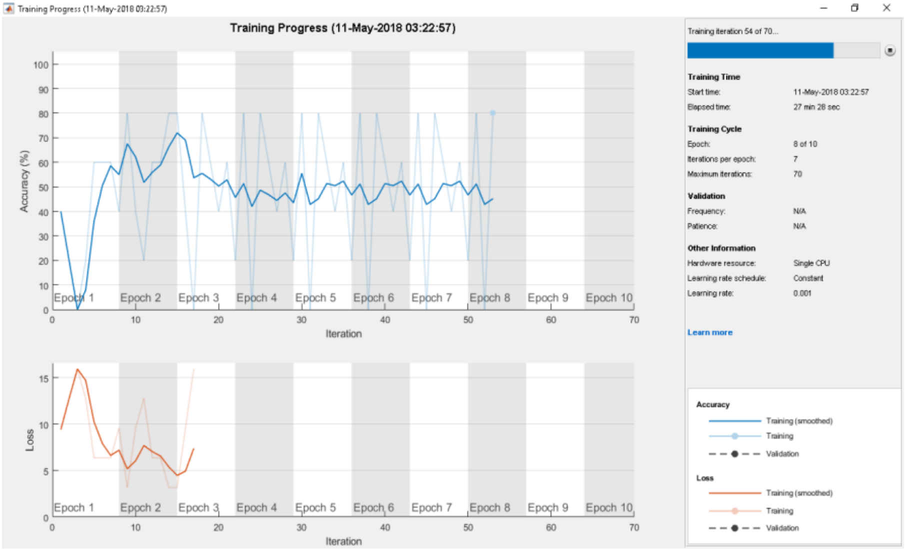
```
%% Training options for DLNN
opts = trainingOptions('sgdm', 'InitialLearnRate', 0.001, 'MaxEpochs',
10, 'MiniBatchSize', 5,'Plots','training-progress');
```

Let provide training for the AlexNet Neural Network after you have set up training options. The training vectors (*TrainingImages_Alex*), the DLNN structure (*layers_Alex*), and the learning options (*opts*) is specifying for this reason.

```
%% Training of DLNN
Net_Alex =trainNetwork(TrainingImages_Alex,layers_Alex, opts);
```

Provide training for the proposed DNN structure in a similar way.

The DNN process is accompanied by graphical and table maintenance, as shown on Fig. 3.10-3.11. The progress results of the DNN training provide in the report.



Training iteration 54 of 70...

Training Time

Start time: 11-May-2018 03:22:57
Elapsed time: 27 min 28 sec

Training Cycle

Epoch: 8 of 10
Iterations per epoch: 7
Maximum iterations: 70

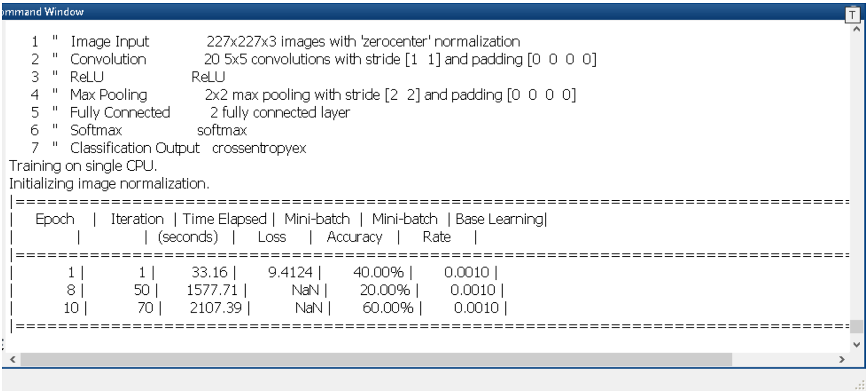
Validation

Frequency: N/A
Patience: N/A

Other Information

Hardware resource: Single CPU
Learning rate schedule: Constant
Learning rate: 0.001

Fig. 3.10 – The progress of DNN training



```
   1   "   Image Input          227x227x3 images with 'zerocenter' normalization
   2   "   Convolution          20 5x5 convolutions with stride [1  1] and padding [0  0  0  0]
   3   "   ReLU                 ReLU
   4   "   Max Pooling          2x2 max pooling with stride [2  2] and padding [0  0  0  0]
   5   "   Fully Connected      2 fully connected layer
   6   "   Softmax              softmax
   7   "   Classification Output   crossentropyex
Training on single CPU.
Initializing image normalization.
|==========================================================================================
|    Epoch    |   Iteration   | Time Elapsed |  Mini-batch  |  Mini-batch  | Base Learning|
|             |               |  (seconds)   |     Loss     |   Accuracy   |     Rate     |
|==========================================================================================
|           1 |             1 |        33.16 |       9.4124 |       40.00% |       0.0010 |
|           8 |            50 |      1577.71 |          NaN |       20.00% |       0.0010 |
|          10 |            70 |      2107.39 |          NaN |       60.00% |       0.0010 |
|==========================================================================================
```

Fig. 3.11 – The training progress of DNN in Matlab environment

5. Assessment of the quality

Use the test set of images created in step 3 to evaluate the quality of the DNN. Average the result and calculate the percentage of the correct classification.

Let use the test set of images created in step 3 To evaluate the quality of the trained DNN. Show at the report the average value of the classification results and determine the percentage of the correct classification of test images.

```
%% Measure network accuracy
predictedLabels_Alex = classify(Net_Alex, TestImages_Alex);
accuracy_Alex = mean(predictedLabels_Alex ==
TestImages_Alex.Labels)
```

Provide the similar calculations for the proposed DNN structure. Compare the results with AlexNet and report their analysis.

3.5 Requirements to the content of the report

1. The subject and purpose of laboratory work.
2. The brief theoretical information.
3. The Information about the task from variant.

4. Provide the input data for training and classification DNN using a digital camera. Present in the report examples of images from each category, according to the task in the variant.

5. Divide the incoming images into two categories: learning and testing. In this case, ensure the uniform distribution of the training sample. Change the resolution of the images. Introduce a Matlab script to run the specified operations in the report.

6. Create two structures of neural networks to solve the image classification problem (according to the variant): by modifying the layers of an existing pre-trained DNN, and by proposing your personal DNN structure. Show the used structures of neural networks in the report. Argue the proposed architectures.

7. Define the training parameters for the two neural network structures and show them in the report.

8. Train the two created neural networks in accordance with the prepared training vectors. The training performance of the DNN show in the report and explain the training quality of each created DNN.

9. Provide a classification of test images that are formed in point 3.5 bu neural networks. Conduct an analysis and compare the results of various DNN architectures. Present the analysis results in the report.

10. Matlab script with comments.

11. Provide conclusions in the report.

3.6 Test questions

1. What is the Deep Learning Neural Network?

2. What inside in DLNN?

3. What are the neural network training methods?

4. How to choose the neural network structure?

5. Main parameters of the neural network?

6. What should be structure of neural network for recognizing of the rotated image of symbols?

3.7 Recommended literature

1. "Nicolas Rashevsky", *En.wikipedia.org*, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Nicolas_Rashevsky. [Accessed: 31- Jul- 2019].

2. "Alan Hodgkin", *En.wikipedia.org*, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Alan_Lloyd_Hodgkin. [Accessed: 31-

Jul- 2019].

3. "Manchester Mark 1", *En.wikipedia.org*, 2019. [Online]. Available: https://en.wikipedia.org/wiki/Manchester_Mark_1. [Accessed: 31- Jul- 2019].

4. E. Alpaydin, "Introduction to Machine Learning, Third Edition", 2014 Massachusetts Institute of Technology. – 640 pp.

5. I. Goodfellow, Y. Bengio, A. Courville, "Deep Learning", 2016, An MIT Press book, 787pp.

6. Stephen J. Chapman, "MATLAB Programming for Engineers", Published November 1st 2007 by Thomson Learning.

3.8 The variants for laboratory work

Variant	**Object types**	**Number of images**	**Known DNN**
1.	Laptop, Mouse, PC	50x50x50	AlexNet
2.	Amchair, Table, Door	50x50x50	GoogLeNet
3.	Lamp, Desk, Window	50x50x50	VGG-16
4.	Pen or Pencil, Copybook, Laptop	50x50x50	VGG-19
5.	Flower, Animal, People	50x50x50	AlexNet
6.	Man, Woman, Computer	50x50x50	GoogLeNet
7.	Cellphone, Notes, Book	50x50x50	VGG-16
8.	Keyboard, Webcam, Microphone	50x50x50	VGG-19
9.	Smile, Distress, Surprise	50x50x50	AlexNet
10.	Hard drive, CD, Flash	50x50x50	AlexNet
11.	Laptop, Mouse, PC	50x50x50	GoogLeNet
12.	Amchair, Table, Door	50x50x50	VGG-19
13.	Lamp, Desk, Window	50x50x50	AlexNet
14.	Pen or Pencil, Copybook, Laptop	50x50x50	AlexNet
15.	Flower, Animal, People	50x50x50	AlexNet
16.	Man, Woman, Computer	50x50x50	VGG-19
17.	Cellphone, Notes, Book	50x50x50	AlexNet
18.	Keyboard, Webcam, Microphone	50x50x50	AlexNet
19.	Smile, Distress, Surprise	50x50x50	GoogLeNet
20.	Hard drive, CD, Flash	50x50x50	GoogLeNet

Laboratory work 4
DEEP LEARNING SPEECH RECOGNITION FOR IoT

Goal and objectives: getting Deep Neural Networks skills for Voice Command Recognition in the Matlab environment

Learning objectives:

– studying the principles of Deep Neural Network (DNN);

– studying Matlab application for sound recognizing task;

– studying the principles of IoT.

Practical tasks:

– obtaining experience working with IoT for recognition of sound commands;

– obtaining experience working with sound recording devices;

– obtaining experience in sound processing;

– practical application of DNN in Matlab environment for recognition of sound commands.

Exploring tasks:

- discover the possibility of DNN to recognize speach.
- investigate the possibility to train DNN.
- investigate efficiency of the different DNN architectures.

Setting up

In preparation for laboratory work it is necessary:

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual, and in references;
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

– access to files at https://github.com/vkoukr/ERASMUS-ALIOT-2019/blob/master/4_Lab_Work_SPEECH_RECOGNITION_Appendix.zip;

– computers with soundcard;

– microphone and loudspeakers (headphones);

– Matlab software (version 2018b or above, RAM – from 4Gb);

– for increasing the training efficiency of DNN, it is suggested (not required for executing the lab) to use the following approaches:

a) GPU-based approach. The CUDA technology (ver. 8 or higher) should be used for NVIDIA GPU and Deep Learning Primitives (cuDNN)**:** High-performance building blocks for DLNN applications including convolutions, activation functions, and tensor transformations;

b) CPU-based approach. Installing Matlab Parallel Processing Toolbox for data processing on multiprocessors platforms or computer cluster organization using Matlab;

c) Cloud-based computational approach. The subscription and license for Internet Cloud is necessary for Matlab (for example: Microsoft Azure, Amazon AWS, Thingspeak or other);

- programming experience in Matlab.

4.1 Synopsis

In this laboratory work you will discover the possibility of Deep Neural Networks to recognize Voice Command. This knowledge allow you to design the IoT systems based on natural language processing approach using DNN.

4.2 Brief theoretical information

4.2.1. IoT principles

The modern concept of IoT involves the interacting objects that use Internet technology and the surrounding environment. The most important features of IoT include artificial intelligence (AI), connectivity, sensors, active engagement, and small device use. One of the possible things that combine IoT and AI is Smart Things [**Error! Reference source not found.**]. This concept is the combination of three technologies: the Internet of Things, Artificial Intelligence and Semantic Web (Fig. 4.1)

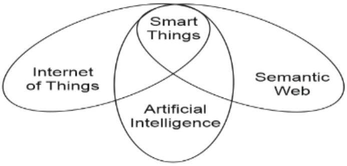


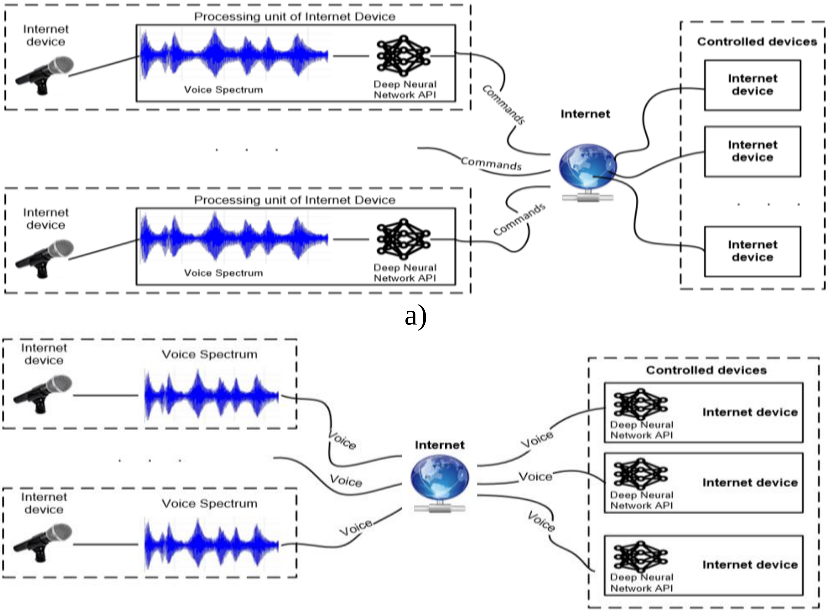
Fig. 4.1 – Concepts of Smart Things

Implementation of such concept allows to provide functioning of a lot of applications without human intervention.

4.2.2. Task description

Let consider Deep Learning Speech Recognition among the set of tasks that are solved using IoT. In the laboratory work it will be considered the DNN for detecting of voice commands in audio stream.

In general, this task has practical importance and can be represented by several applications depending on the location of the using data processing module of DNN (Fig. 4.2).



a)

b)

Fig. 4.2 – Task definition (different location of DNN API)

Let consider some Internet-based device that is controlleb by commands through the Internet. Such device is a controled object in the technical system. On the other hand, the controlling entity that control by the Internet device provides transmission of commands to an object (Fig. 4.2). It is possible to create a new type of interface between the object and subject in some technical system using natural language. For example, there is possibility to control by a TV, lighting, heating, or other device through Internet using sound commands. Also, the robotic complexes or recognition may be controlled by using voice commands or recognizing of people by sound phrase.

Thus, as it is evident from the above presented tasks, it is important to realize the technology of the IoT to ensure the sound recognition of sound commas against at the audio stream. According to the goal, the subject of the study is the methods for recognizing voice commands using DNN.

The main principles for designing of DNN.

A convolutional neural network can be used for recognizing of voice commands (Fig. 4.3). The main principles of DNN could be find at [2]. A lot of solutions for applications of DNN with practical examples presents at Matlab [3].

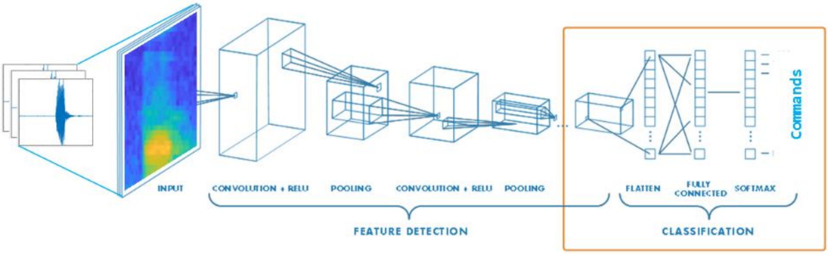


Fig. 4.3 – Convolutional Neural Networks

4.3 Execution order and discovery questions

1. Organization of input data.

Load Background Data

In order to recognize voice commands in a sound stream, you must save background sound files with the extension "* .wav" in a user-defined directory (for example, *"My_sounds"*). Such audio files can be generated from microphone or downloaded from the Internet *http://download.tensorflow.org/data/speech_commands_v0.01.tar.gz* [4], or copied from the directory *"background noise"*, at the accompanying files to the laboratory work at *https://github.com/vkoukr/ERASMUS-ALIOT-*

2019/blob/master/4_Lab_Work_SPEECH_RECOGNITION_Appendix.zip.

Choose Phrases to Recognize

To form the training set for DNN it is necessary to form a set of vocabulary phrases. The voice phrases for labwork it is necessary to create according to the options given at the variants that are given in the table at the end of laboratory work. To ensure the quality of DNN training, it is necessary to provide a training set containing about 1000 audio images of one vocabulary phrase.

Recording of the sound pattern of one voice command must be provided by using a microphone as well as an available audio recording program (for example, "Voice Recorder", which is typical in OS Windows10). In this case, the parameters of the recorded sound image must be:

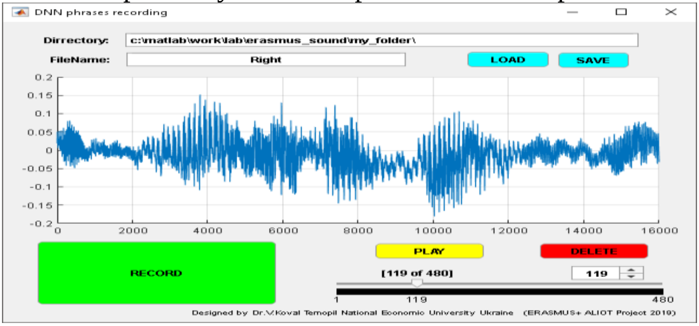
a) Sampling rate in Hz – 16000;

b) Bits per sample – 16;

c) The number of channels – 1 (mono);

d) Time Duration per Sample– 1 sec.;

For more comfortable usage of voice phrases, it is advisable to use the Matlab environment. It is proposed to apply the developed software (*Reccord_voice.mlapp*), that is accompanying at the supported folder for laboratory work (*https://github.com/vkoukr/ERASMUS-ALIOT-2019/blob/master/4_Lab_Work_SPEECH_RECOGNITION_Appendix.zip*). To do this, you need to run the "*Reccord_voice.mlapp*" file in Matlab. A user-friendly interface was created (Fig. 4.4), and allows you to create and manipulate by the audio patterns of voice phrases.



Dirrectory: c:\matlab\work\lab\erasmus_sound\my_folder\

FileName: Right

[119 of 480]

119

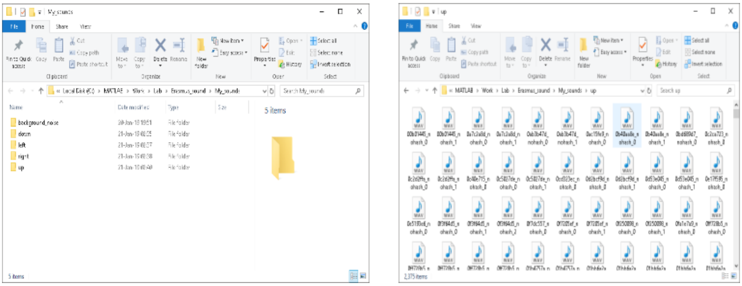
Designed by Dr.V.Koval Ternopil National Economic University Ukraine (ERASMUS+ ALIOT Project 2019)

Fig. 4.4 – Matlab Software for creating sound patterns

In order to decrease the time for creation of DNN training set it is necessary to obtain a lot of voice patterns in the dataset. The variety of voice patterns it is suggested to create by all students in the studding group. In this case, one voice phrase it is necessary to record several times of by all students in the group. For example, if there are 20 students in a group, then having recorded the voice command 50 times, you can get a sample that will consist of 20 * 50 = 1000 patterns. This procedure should be performed for each voice phrase of all variants in the laboratory work. After each student prepares and save records, it is necessary to combine them by reading from separate files. Finally, the set of voice phrase should be save to filename containing the name of the voice phrase (for example, "right"). This procedure is provided by Matlab software "*Reccord_voice.mlapp*" (Fig. 4.4).

Composing of The Working Folder

A folder data structure must be created on the basis of the above actions, for further actions in laboratory work. For this purpose, it is necessary to determine the folder where the files will be located. For example, on Fig. 4.5 shows a directory structure that contains 5 folders with files corresponding to voice phrase as well as background sound.



```
background_noise   20-Jan-19 19:51   File folder
down               21-Jan-19 02:05   File folder
left               21-Jan-19 02:07   File folder
right              21-Jan-19 02:08   File folder
up                 21-Jan-19 02:40   File folder
```

a) b)

Fig. 4.5 – Structure of the directory:
(a) the root directory; (b) subdirectory

At the same time, the folder "*background_noise*" (Fig. 4.5a) containing sound of background noise, and at least one folder containing explored voice phrase (for example, the folder "up") must be mandatory. It is recommended to create additional folders that contain patterns of other voice phrases (for example, formed, by students from the other variants, or downloaded from [4]). In the last cases, the quality of recognition voice phrases by the artificial neural network will be significantly higher. The structure of the directory with sound patterns that contain 4 voice commands ("up", "down", "left", "right"), as well as the background sound is shown on Fig. 4.5a.

2. Audio Data Store Creation

Let execute the following Matlab code to create the audio Data Store with sound patterns.

```
tempdir='c:\matlab\work\lab\erasmus_sound\';
datafolder = fullfile(tempdir,'My_sounds');

ads = audioDatastore(datafolder, ...
   'IncludeSubfolders',true, ...
   'FileExtensions','.wav', ...
   'LabelSource','foldernames')
ads0 = copy(ads);
```

The specified code captures patterns from sound recordings in *“My_sounds”* folder that have been created at the previous stage.

3. Choosing Voice Phrases to Recognize

Let define the list of voice phrases to study (variable *isCommands*) for crating the training sets of neural network. All other sound images in the Data Set will be defined as "*isUknown.*"

To reduce the class imbalance between the known and unknown words and speed up processing, only include a fraction (includeFraction variable in the code) of the unknown words in the training set. Do not include the longer files with background noise in the training set yet. Background noise will be added in a separate step later.

```
commands = categorical(["up"]);

isCommand = ismember(ads.Labels,commands);
isUnknown = ~ismember(ads.Labels,[commands,"background_noise"]);

includeFraction = 0.2;
mask = rand(numel(ads.Labels),1) < includeFraction;
isUnknown = isUnknown & mask;
```

```
ads.Labels(isUnknown) = categorical("unknown");

ads = subset(ads,isCommand|isUnknown);
countEachLabel(ads)
```

4. Split Data into Training, Validation, and Test Sets

You need to create three subsets of data: training sample, checking and test in order to form training sets of the neural network. Three separate audio Data Set are formed for this (*adsTrain, adsValidation, adsTest*).

```
    [adsTrain,adsValidation,adsTest] =
splitEachLabel(ads,0.25,0.25,'randomized');
```

In this case, the formation of training sets provides by random sampling of data from the existing proportion: 25% - training set, 25% - validate set, 50% - test set.

5. Compute Speech Spectrograms

For efficient training of a deep neural network, let convert the speech waveforms to log-bark auditory spectrograms. For this reason let define the parameters of the spectrogram calculation.

- «segmentDuration», is the duration of each speech clip (in seconds);
- «frameDuration», is the duration of each frame for spectrogram calculation;
- «hopDuration», is the time step between each column of the spectrogram.
- «numBands», is the number of log-bark filters and equals the height of each spectrogram.

```
segmentDuration = 1;
frameDuration = 0.025;
hopDuration = 0.010;
numBands = 40;
```

Compute the spectrograms for the training, validation, and test sets by using the supporting function «speechSpectrograms». This function uses auditorySpectrogram for the spectrogram calculations. To obtain data with a smoother distribution, take the logarithm of the spectrograms using a small offset epsil.

```
epsil = 1e-6;
XTrain= speechSpectrograms(adsTrain,segmentDuration,frameDuration,hopDuration,numBands);
XTrain = log10(XTrain + epsil);
XValidation= speechSpectrograms(adsValidation,segmentDuration,frameDuration,hopDuration,numBands);
XValidation = log10(XValidation + epsil);

XTest = speechSpectrograms(adsTest,segmentDuration,frameDuration,hopDuration,numBands);
XTest = log10(XTest + epsil);

YTrain = adsTrain.Labels;
YValidation = adsValidation.Labels;
YTest = adsTest.Labels;
```

To visualize the formed spectrograms, it is necessary to run the following Matlab code.

```
specMin = min(XTrain(:));
specMax = max(XTrain(:));
idx = randperm(size(XTrain,4),3);
figure('Units','normalized','Position',[0.2 0.2 0.6 0.6]);
for i = 1:3
   [x,fs] = audioread(adsTrain.Files{idx(i)});
   subplot(2,3,i)
   plot(x)
   axis tight
   title(string(adsTrain.Labels(idx(i))))

   subplot(2,3,i+3)
   spect = XTrain(:,:,1,idx(i));
   pcolor(spect)
   caxis([specMin+2 specMax])
   shading flat
```

```
        sound(x,fs)
        pause(2)
    end
```

The screenshots of spectrographs (Fig. 4.6) it is necessary to present at the report of laboratory work execution.

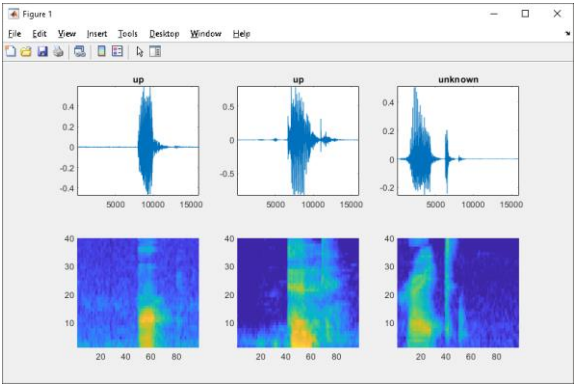


Fig. 4.6 – Examples of spectrograms of sound patterns

6. Add Background Noise to Data

The network must be able not only to recognize different spoken phrases but also to detect if the input contains silence or background noise. Use the audio files in the «*background_noise*» folder to create samples of one-second clips of background noise. It is possible to create your own recordings of background noise. If so, it is necessary to locate them to the «*background_noise*» folder. To calculate *numBkgClips* spectrograms of background clips taken from the audio files in the *adsBkg* datastore, use the supporting function «*backgroundSpectrograms*». Before calculating the spectrograms, the function rescales each audio clip with a factor sampled from a log-uniform distribution in the range given by volumeRange.

Create 4000 background clips and rescale each by a number between 1e-4 and 1. *XBkg* contains spectrograms of background noise with volumes ranging from practically silent to loud.

```
    adsBkg = subset(ads0, ads0.Labels=="background_noise");
    numBkgClips = 4000;
    volumeRange = [1e-4,1];

    XBkg =
backgroundSpectrograms(adsBkg,numBkgClips,volumeRange,segmentDurati
on,frameDuration,hopDuration,numBands);
    XBkg = log10(XBkg + epsil);
```

Split the spectrograms of background noise between the training, validation, and test sets. To increase the robustness of the network to noise, you can also try mixing background noise into the speech files.

```
    numTrainBkg = floor(0.8*numBkgClips);
    numValidationBkg = floor(0.1*numBkgClips);
    numTestBkg = floor(0.1*numBkgClips);

    XTrain(:,:,:,end+1:end+numTrainBkg) = XBkg(:,:,:,1:numTrainBkg);
    XBkg(:,:,:,1:numTrainBkg) = [];
    YTrain(end+1:end+numTrainBkg) = "background";

    XValidation(:,:,:,end+1:end+numValidationBkg) =
XBkg(:,:,:,1:numValidationBkg);
    XBkg(:,:,:,1:numValidationBkg) = [];
    YValidation(end+1:end+numValidationBkg) = "background";

    XTest(:,:,:,end+1:end+numTestBkg) = XBkg(:,:,:,1: numTestBkg);
    clear XBkg;
    YTest(end+1:end+numTestBkg) = "background";

    YTrain = removecats(YTrain);
    YValidation = removecats(YValidation);
    YTest = removecats(YTest);
```

Plot the distribution of the different class labels in the training and validation sets and include it to the report (Fig. 4.7).

```
    figure('Units','normalized','Position',[0.2 0.2 0.5 0.5]);
    subplot(2,1,1)
    histogram(YTrain)
    title("Training Label Distribution")
    subplot(2,1,2)
    histogram(YValidation)
    title("Validation Label Distribution")
```

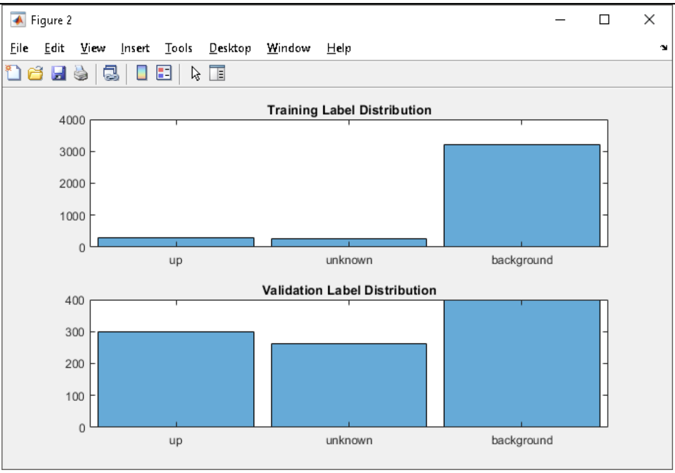


Figure 2
File Edit View Insert Tools Desktop Window Help
Training Label Distribution
4000
3000
2000
1000
0
up
unknown
background
Validation Label Distribution
400
300
200
100
0
up
unknown
background

Fig. 4.7 – Ddistribution of the different class labels in the training and validation sets

7. Add Data Augmentation

Create an augmented image datastore for automatic augmentation and resizing of the spectrograms. Translate the spectrogram randomly up to 10 frames (100 ms) forwards or backwards in time, and scale the spectrograms along the time axis up or down by 20 percent. Augmenting the data can increase the effective size of the training data and help prevent the network from overfitting. The augmented image datastore creates augmented images in real time during training and inputs them to the network. No augmented spectrograms are saved in memory.

```
    sz = size(XTrain);
    specSize = sz(1:2);
    imageSize = [specSize 1];
    augmenter = imageDataAugmenter( ...
       'RandXTranslation',[-10 10], ...
       'RandXScale',[0.8 1.2], ...
       'FillValue',log10(epsil));
    augimdsTrain = augmentedImageDatastore(imageSize,XTrain,YTrain, ...
    'DataAugmentation',augmenter);
```

8. Working with DNN

8.1. Defining of DNN structure

The DNN structure should be defined before applications. The Matlab code for defining of DNN structure is shown below. The defined DNN include only five convolutional layers with few filters. The variable «*numF*» controls the number of filters in the convolutional layers. Within the lab work, try to increasing the network depth by adding identical blocks of convolutional, batch normalization, and ReLU layers. You can also try to increase the number of convolutional filters by increasing *numF*. This will increase the accuracy of the network.

The DNN structure use a weighted cross entropy classification loss. The «*weightedClassificationLayer(classWeights)*» creates a custom classification layer that calculates the cross entropy loss with observations weighted by *classWeights*. Specify the class weights in the same order as the classes appear in categories (*YTrain*). To give each class equal total weight in the loss, use class weights that are inversely proportional to the number of training examples in each class. When using the Adam optimizer to train the network, the training algorithm is independent of the overall normalization of the class weights.

```
classWeights = 1./countcats(YTrain);
classWeights = classWeights'/mean(classWeights);
numClasses = numel(categories(YTrain));

dropoutProb = 0.2;
numF = 12;
layers = [
    imageInputLayer(imageSize)

    convolution2dLayer(3,numF,'Padding','same')
    batchNormalizationLayer
    reluLayer

    maxPooling2dLayer(3,'Stride',2,'Padding','same')

    convolution2dLayer(3,2*numF,'Padding','same')
```

```
        batchNormalizationLayer
        reluLayer
        maxPooling2dLayer(3,'Stride',2,'Padding','same')

        convolution2dLayer(3,4*numF,'Padding','same')
        batchNormalizationLayer
        reluLayer

        maxPooling2dLayer(3,'Stride',2,'Padding','same')

        convolution2dLayer(3,4*numF,'Padding','same')
        batchNormalizationLayer
        reluLayer
        convolution2dLayer(3,4*numF,'Padding','same')
        batchNormalizationLayer
        reluLayer

        maxPooling2dLayer([1 13])

        dropoutLayer(dropoutProb)
        fullyConnectedLayer(numClasses)
        softmaxLayer
        weightedClassificationLayer(classWeights)];
```

Explore the possibility of using AlexNet's modified network n the same way by applying transfer-learning technique. It is also encouraged to create its own structure of the DNN.

```
options('sgdm', 'InitialLearnRate', 0.001, 'MaxEpochs', 10,
'MiniBatchSize', 5,'Plots','training-progress');
%% Training of DLNN
      Net_Alex          =trainNetwork(TrainingImages_Alex,layers_Alex,
options);
```

8.2 Training of DNN

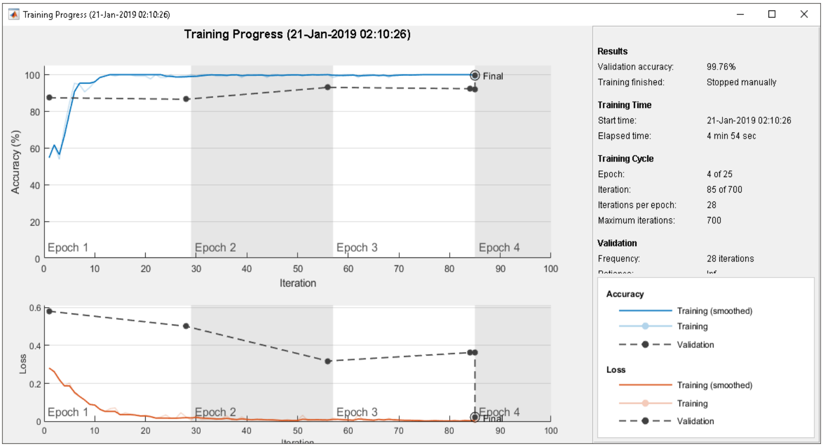
Let define parameters for training of DNN by using *trainingOptions*. Use the Adam optimizer with a mini-batch size of 128. Train for 15 epochs and reduce the learning rate by a factor of 10 after 20 epochs.

```
miniBatchSize = 128;
validationFrequency = floor(numel(YTrain)/miniBatchSize);
options = trainingOptions('adam', ...
    'InitialLearnRate',3e-4, ...
    'MaxEpochs',15, ...
    'MiniBatchSize',miniBatchSize, ...
    'Shuffle','every-epoch', ...
    'Plots','training-progress', ...
    'Verbose',false, ...
    'ValidationData',{XValidation,YValidation}, ...
    'ValidationFrequency',validationFrequency, ...
    'LearnRateSchedule','piecewise', ...
    'LearnRateDropFactor',0.1, ...
    'LearnRateDropPeriod',20);
```

Provide training of the Deep Neural Network by specifying the training vectors (*augimdsTrain*), the DNN structure (*layers*) and training options (*options*). If you do not have a GPU, then training the network can take time.

```
trainedNet = trainNetwork(augimdsTrain,layers,options);
```

The training process of DNN is accompanied by graphical and tabular representation, shown in Fig. 4.8. The results of training the DNN it is necessary to include into labwork report.


Training Progress (21-Jan-2019 02:10:26)

Training Progress (21-Jan-2019 02:10:26)

Results
Validation accuracy: 99.76%
Training finished: Stopped manually

Training Time
Start time: 21-Jan-2019 02:10:26
Elapsed time: 4 min 54 sec

Training Cycle
Epoch: 4 of 25
Iteration: 85 of 700
Iterations per epoch: 28
Maximum iterations: 700

Validation
Frequency: 28 iterations
Patience: Inf

Fig. 4.8 – Graphical representation of training the DNN in Matlab

8.3 Estimation of DNN quality

To evaluate the quality of the trained on the previous stage DNN, use the test set of patterns created on step 4. Average the results of the classification and determine the percentage of correct voice pattern classification.

```
YValPred = classify(trainedNet,XValidation);
validationError = mean(YValPred ~= YValidation);
YTrainPred = classify(trainedNet,XTrain);
trainError = mean(YTrainPred ~= YTrain);
disp("Training error: " + trainError*100 + "%")
disp("Validation error: " + validationError*100 + "%")
```

Make similar calculations for the modifyed DNN. According to the purpose of laboratory work, it is necessary to investigate the possibility of recognizing voice commands using DNN. Therefore, it is necessary to provide experimental research, which will ensure the selection of such a structure of DNN that will ensure error-free recognition of sound images. To do this, select the three DNN structures, and compare their quality for recognizing the voice phrases in the report.

9. DNN Online Application For Detection Commands Using Streaming Audio from Microphone

Test your best trained DNN on streaming audio from the microphone. Try saying one of the phrases, saving in the source folder. Then, try saying any unknown words. Look at the behavior of DNN. Create statistics of recognizing commands per 100 saying words and include it to the report.

Run the below presented code-routines to request audio-flow from microphone.

```
%Specify the audio sampling rate and classification rate in Hz
% and create an audio device reader that can read audio from your microphone.

fs = 16e3;
classificationRate = 20;
audioIn = audioDeviceReader('SampleRate',fs, ...
   'SamplesPerFrame',floor(fs/classificationRate));
%Specify parameters for the streaming spectrogram computations and initialize an audio
buffer.
%Extract the classification labels of the network. Initialize buffers of half a second for the
labels
%and classification probabilities of the streaming audio. Use these buffers to compare the
% classification results over a longer period of time and by that build 'agreement' over when a
%command is detected.

frameLength = frameDuration*fs;
hopLength = hopDuration*fs;
waveBuffer = zeros([fs,1]);
labels = trainedNet.Layers(end).Classes;
YBuffer(1:classificationRate/2) = categorical("background");
probBuffer = zeros([numel(labels),classificationRate/2]);

%Create a figure and detect commands as long as the created figure exists. To stop the live
detection, simply close the figure.
h = figure('Units','normalized','Position',[0.2 0.1 0.6 0.8]);

while ishandle(h)
   % Extract audio samples from the audio device and add the samples to the buffer.
   x = audioIn();
   waveBuffer(1:end-numel(x)) = waveBuffer(numel(x)+1:end);
   waveBuffer(end-numel(x)+1:end) = x;

   % Compute the spectrogram of the latest audio samples.
   spec = auditorySpectrogram(waveBuffer,fs, ...
      'WindowLength',frameLength, ...
      'OverlapLength',frameLength-hopLength, ...
      'NumBands',numBands, ...
      'Range',[50,7000], ...
      'WindowType','Hann', ...
      'WarpType','Bark', ...
      'SumExponent',2);
```

```
    spec = log10(spec + epsil);

    % Classify the current spectrogram, save the label to the label buffer,
    % and save the predicted probabilities to the probability buffer.
    [YPredicted,probs] = classify(trainedNet,spec,'ExecutionEnvironment','cpu');
    YBuffer(1:end-1)= YBuffer(2:end);
    YBuffer(end) = YPredicted;
    probBuffer(:,1:end-1) = probBuffer(:,2:end);
    probBuffer(:,end) = probs';

    % Plot the current waveform and spectrogram.
    subplot(2,1,1);
    plot(waveBuffer)
    axis tight
    ylim([-0.2,0.2])
    subplot(2,1,2)
    pcolor(spec)
    caxis([specMin+2 specMax])
    shading flat

    % Now do the actual command detection by performing a very simple
    % thresholding operation. Declare a detection and display it in the
    % figure title if all of the following hold:
    % 1) The most common label is not |background|.
    % 2) At least |countThreshold| of the latest frame labels agree.
    % 3) The maximum predicted probability of the predicted label is at
    % least |probThreshold|. Otherwise, do not declare a detection.
    [YMode,count] = mode(YBuffer);
    countThreshold = ceil(classificationRate*0.2);
    maxProb = max(probBuffer(labels == YMode,:));
    probThreshold = 0.7;
    subplot(2,1,1);
    if YMode == "background" || count<countThreshold || maxProb < probThreshold
        title(" ")
    else
        title(string(YMode),'FontSize',20)
    end
    drawnow
end
```

The graphical representation of the streaming audio from microphone is given on Fig. 4.9. The result of speech command recognition represents at the title position by text.

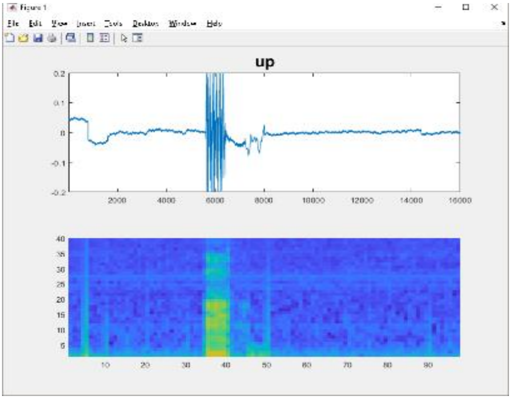


Fig .4.9 – Streaming audio from microphone

4.4 Requirements to the content of the report

1. The subject and purpose of laboratory work.

2. The brief theoretical information.

3. The Information about the task from variant.

4. Provide creating of the training sets for DNN using the microphone. It should contain the audio files with voice phrases. Also create audio Datastore. Choose the voice phrase to recognize by DNN.

Present in the report examples of sound wave from each category, according to the variant and the screenshot with the folder structure for audio Datastore.

5. Provide splitting of voice patterns into three categories: training, validation and testing with proportion 25%, 25% and 50% from the input audio Dataset. Provide a Matlab script for the implementation of the specified operations in the report.

6. Compute speech spectrograms for the training, validation, and test sets. The screenshots of spectrographs from each category of Dataset include to the report.

7. Add a Background Noise to Data and include screenshot to the report with a distribution of the different class labels in the training and validation sets.

8. Provide data augmentation according to the needs of DNN and in indicate them into report

9. Define two structures of DNN for voice phrase recognition (first - presented at the section 2 and second proposed by yourself). Describe and show the structures of DNN at the report.

10. Define options and provide training of the DNNs using training dataset. Show the training options and training progress at the report.

11. Investigate the possibility of recognizing voice commands using DNN. In order to do this, evaluate the quality of the trained DNNs by using testing Dataset. The results of classification quality by each DNN presents at the report.

12. Test the best trained DNN on streaming audio from the microphone. Look at the classification quality of DNN for the online inputs from microphone. Create statistics of recognizing commands per 100 saying words and include it to the report.

13. Matlab script for voice phrase recognition using DNN with comments.

14. Provide conclusions in the report.

4.5 Test questions

1. What is the Deep Learning Neural Network?

3. What pattern recognition means?

4. How to choose the neural network structure?

5. What parameters has Deep Neural Network?

6. What spectrogram means?

7. What are the application of DNN in IoT?

8. What are the differences between training, validation and testing sets?

9. What are the possibilities to estimate quality of DNN?

10. What information incudes Matlab’s audioDatastore?

4.6 Recommended literature

1. "Internet of Things and the Prelude to Artificial Intelligence", *The Internet of Things,* 2019. [Online]. Available: http://www.infiniteinformationtechnology.com/the-internet-of-things-prelude-to-artificial-intelligence. [Accessed: 31- Jul- 2019].

2. Ian Goodfellow, Yoshua Bengio and Aaron Courville, Deep Learning, 2016, An MIT Press book, 787pp.

3. "Deep Learning Applications- MATLAB & Simulink", *Mathworks.com,* 2019. [Online]. Available:

https://www.mathworks.com/help/deeplearning/deep-learning-applications.html;jsessionid=de95c95c08b7a0eecabba8c48446.
[Accessed: 31- Jul- 2019].

4. APA-style citation: "Warden P. Speech Commands: A public dataset for single-word speech recognition, 2017. Available from http://download.tensorflow.org/data/speech_commands_v0.01.tar.gz".

4.7 The variants for laboratory work

Variant	Voice phrase to recognize	Other voice patterns
1.	Up	Down, Left, Right,
2.	Down	Up, Left, Right
3.	Left	Up, Down, Right
4.	Right	Up, Down, Left
5.	Top	Left, Right, Bottom
6.	Bottom	Up, Down, Top
7.	One	Two, Three, Four
8.	Two	Three, Four, Five
9.	Three	Four, Five, Six
10.	Four	Five, Six, Seven
11.	Five	Six, Seven, Eight
12.	Six	Seven, Eight, Nine
13.	Seven	Eight, Nine, Ten
14.	Eight	Five, Six, Seven
15.	Nine	Six, Seven, Eight
16.	Ten	Seven, Eight, Nine
17.	Sunday	Thursday, Friday, Saturday
18.	Monday	Saturday, Tuesday, Wednesday
19.	Tuesday	Sunday, Monday, Wednesday
20.	Wednesday	Tuesday, Thursday, Friday
21.	Thursday	Monday, Tuesday, Wednesday
22.	Friday	Monday, Tuesday, Wednesday
23.	Saturday	Sunday, Monday, Tuesday

Big Data Benchmarking for IoT Based Systems

A.V. Gorbenko and O.M. Tarasyuk

Laboratory work 1
DEPLOY AN HDINSIGHT HADOOP CLUSTER ON LINUX

Goal and objectives: this work is designed to acquaint students with the process of deploying and running Hadoop clusters provisioned by HDInsight on Linux VMs to process big data.

Learning objectives: in this lab, you will learn how to create an HDInsight cluster running Linux.

Practical tasks: deploying a cloud HDInsight cluster which is Microsoft Azure’s implementation of Hadoop.

Exploring tasks: no

Setting up

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual and listed in the Recommended literature section;
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

Prerequisites

The following are required to complete this hands-on lab:

– An active Microsoft Azure subscription. If you don't have one, sign up for a free trial or use your pay-as-you-go subscription;

1.1 Synopsis

In this lab work you will learn how to create and deploy a Hadoop cluster on Microsoft Azure Cloud (Azure HDInsight).

1.2 Brief theoretical information:

When you consider that there are more than 20 billion devices connected to the Internet today, most all of them generating data, and then think of the massive amounts of data being produced by Web sites, social networks, and other sources, you begin to understand the true implications of *Big Data*. Data is being collected in ever-escalating volumes, at increasingly high velocities, and in a widening variety of formats, and it's being used in increasingly diverse contexts. "Data" used to be something stored in a table in a database, but today it can be a sensor reading, a tweet, a GPS location, or almost anything else. The challenge for information scientists is to make sense of all that data.

A popular tool for analyzing big data is Apache Hadoop. Hadoop is “a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models”. It is frequently combined with other open-source frameworks such as Apache Spark, Apache HBase, and Apache Storm to increase its capabilities and performance. Azure HDInsight is the Azure implementation of Hadoop, Spark, HBase, and Storm, with other tools such as Apache Pig and Apache Hive thrown in to provide a comprehensive and high-performance solution for advanced analytics. HDInsight can spin up Hadoop clusters for you using either Linux or Windows as the underlying operating system, and it integrates with popular business-intelligence tools such as Microsoft Excel and SQL Server Analysis Services.

1.3 Execution order and discovery questions:

In this exercise, you will use the Azure Portal to deploy an HDInsight Hadoop cluster with Linux installed on the cluster’s nodes.

1. Open the Azure Portal in your browser. If you are asked to log in, do so using your university email account. Alternatively, you can create a new account using one of your private e-mails.

2. Click *“****+Create a resource***” in the upper-left corner of the portal. Then click “***Data + Analytics***”, followed by HDInsight.

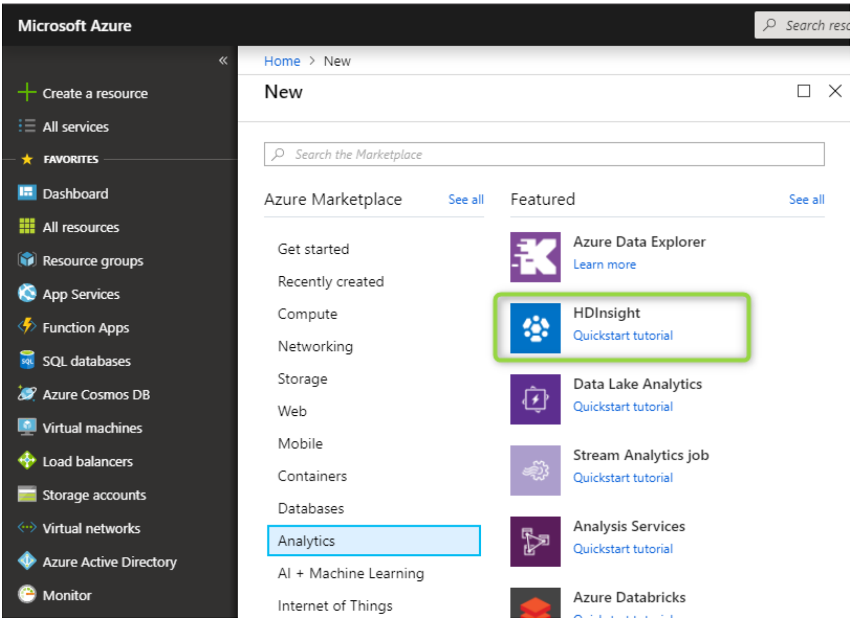
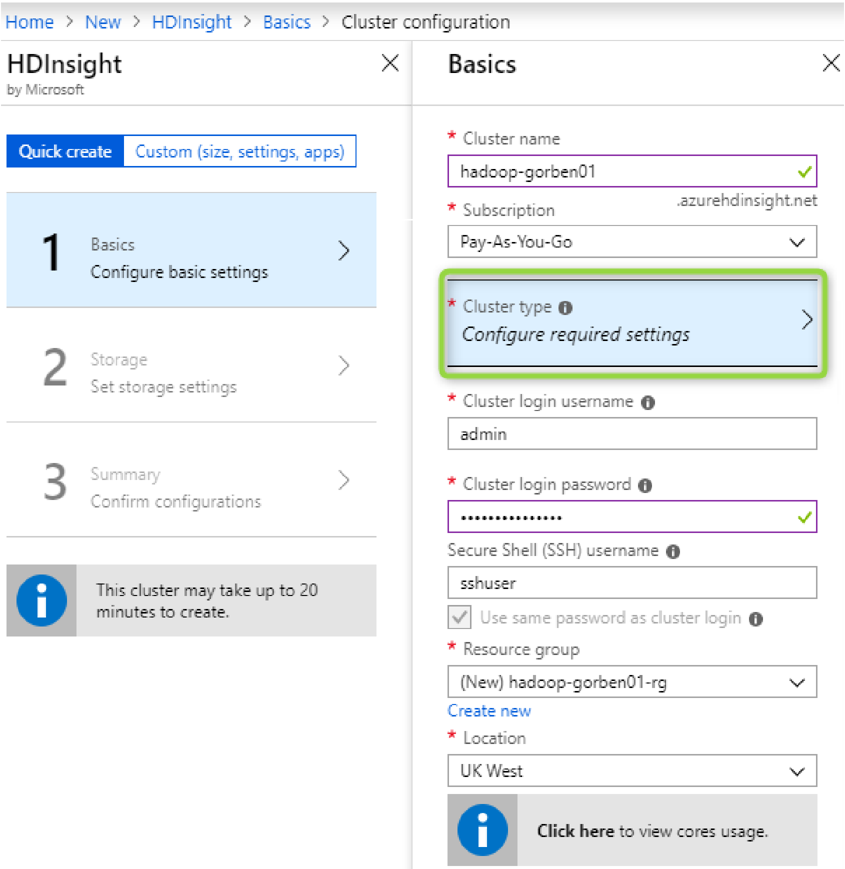


Fig. 1.1 – Creating an HDInsight cluster

2. In the **Cluster Name** box, enter a unique DNS name for the cluster and make sure a green check mark appears next to it indicating that the name is valid and unique.

In case someone else in the lab selects the same cluster name, make it as unique as possible by including birth dates, initials, and anything else you care to add. The name you entered may be unique right now, but it might NOT be unique a few minutes into the deployment.

Enter *Azure4Research!* for **Cluster login password** or you can set your own password. Select **Create new** under **Resource group** and enter the resource-group name “HadoopLabResourceGroup”. Select the **Location** nearest you, and then click **Cluster type** to open a “Cluster configuration” blade.



Home > New > HDInsight > Basics > Cluster configuration

HDInsight
by Microsoft

Quick create | Custom (size, settings, apps)

1 Basics
Configure basic settings

2 Storage
Set storage settings

3 Summary
Confirm configurations

This cluster may take up to 20 minutes to create.

Basics

* Cluster name
hadoop-gorben01 ✓
.azurehdinsight.net

* Subscription
Pay-As-You-Go

* Cluster type
Configure required settings

* Cluster login username
admin

* Cluster login password
•••••••••••••••• ✓

Secure Shell (SSH) username
sshuser

☑ Use same password as cluster login

* Resource group
(New) hadoop-gorben01-rg
Create new

* Location
UK West

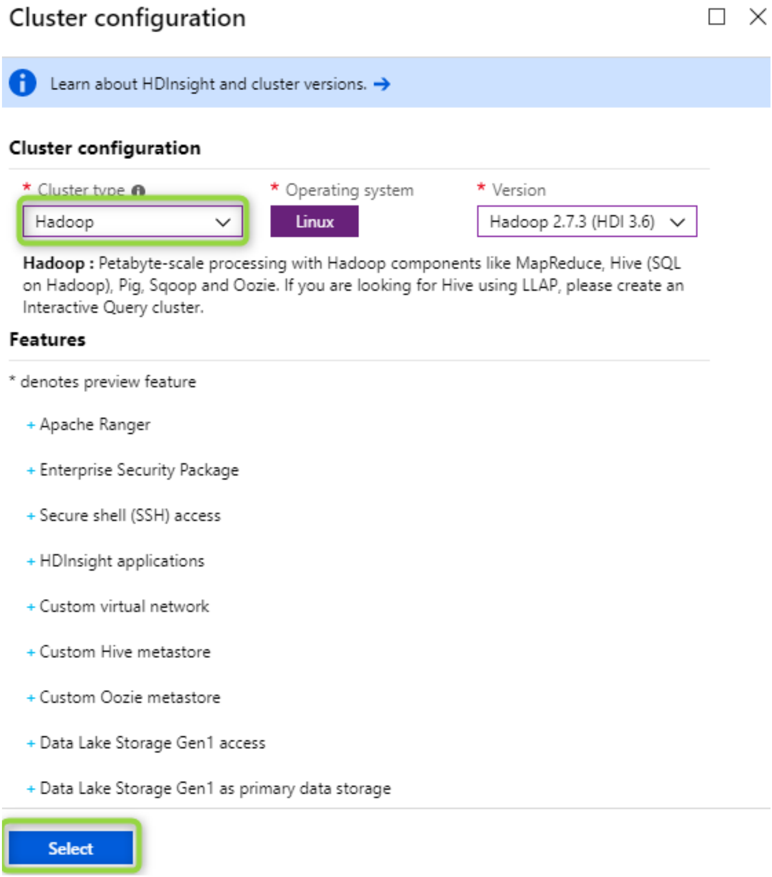
Click here to view cores usage.

Fig. 1.2 – Entering basic cluster settings

3. Select **Hadoop** as the **Cluster type**. Make sure **Linux** is selected as the **Operating system**, and accept the default Hadoop version on the right.

Make sure **Cluster tier** is set to **Standard**.

Click the **Select** button at the bottom of the blade, and then finish up by clicking the **Next** button at the bottom of the “Basics” blade.



Cluster configuration

Learn about HDInsight and cluster versions. →

Cluster configuration

* Cluster type: Hadoop

* Operating system: Linux

* Version: Hadoop 2.7.3 (HDI 3.6)

Hadoop : Petabyte-scale processing with Hadoop components like MapReduce, Hive (SQL on Hadoop), Pig, Sqoop and Oozie. If you are looking for Hive using LLAP, please create an Interactive Query cluster.

Features

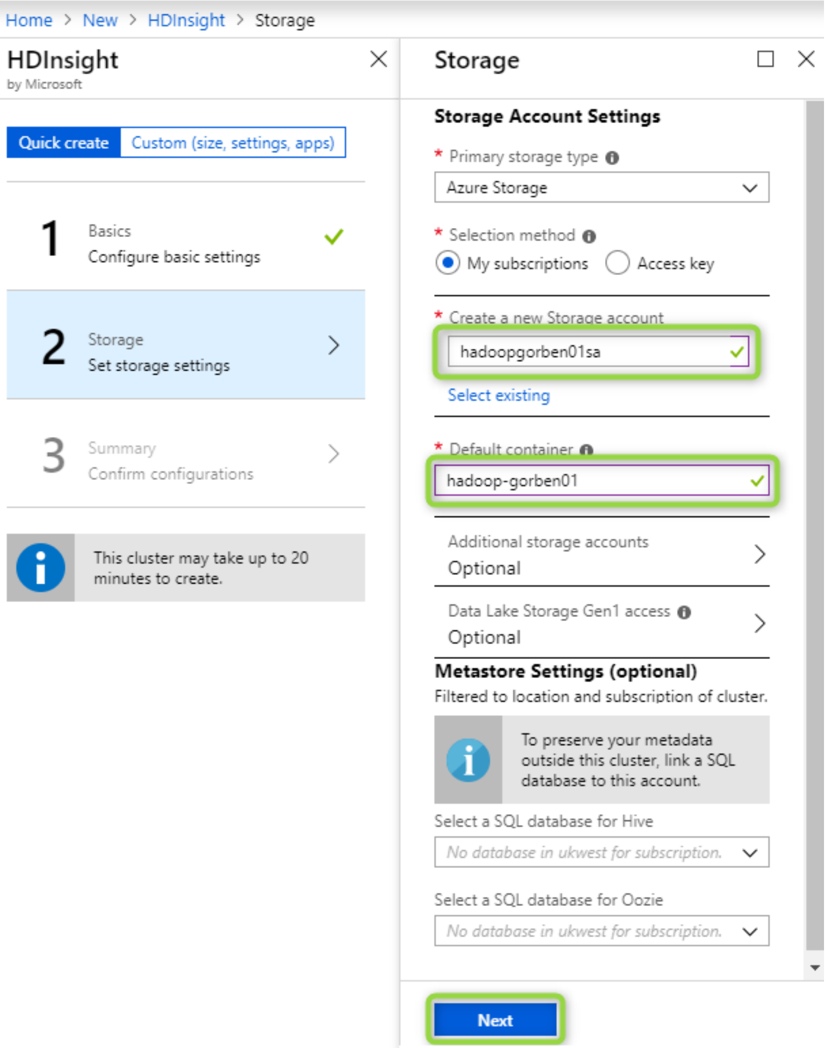
* denotes preview feature

- + Apache Ranger
- + Enterprise Security Package
- + Secure shell (SSH) access
- + HDInsight applications
- + Custom virtual network
- + Custom Hive metastore
- + Custom Oozie metastore
- + Data Lake Storage Gen1 access
- + Data Lake Storage Gen1 as primary data storage

Fig. 1.3 – Entering cluster-configuration settings

4. Make sure **Primary storage type** is set to **Azure Storage** and **Selection method** is set to **My subscriptions**. Then enter a unique storage account name, once more making it as unique as possible by including birth dates or other information. (If the portal selects an existing storage account by default, click **Create new** to create a new storage account). Storage account name must be between 3 and 24 characters in length and use numbers and lower-case letters ONLY. Type a unique name into the **Default container** box (e.g. “hadooplab”), and then click **Next**.

These settings apply to the storage account that is provisioned along with the cluster. The storage account contains the cluster's file system and the software installed inside.



Home > New > HDInsight > Storage

HDInsight
by Microsoft

Quick create | Custom (size, settings, apps)

1 Basics
Configure basic settings ✓

2 Storage
Set storage settings >

3 Summary
Confirm configurations >

This cluster may take up to 20 minutes to create.

Storage

Storage Account Settings

* Primary storage type
Azure Storage

* Selection method
● My subscriptions ○ Access key

* Create a new Storage account
hadoopgorben01sa ✓
Select existing

* Default container
hadoop-gorben01 ✓

Additional storage accounts
Optional >

Data Lake Storage Gen1 access
Optional >

Metastore Settings (optional)
Filtered to location and subscription of cluster.

To preserve your metadata outside this cluster, link a SQL database to this account.

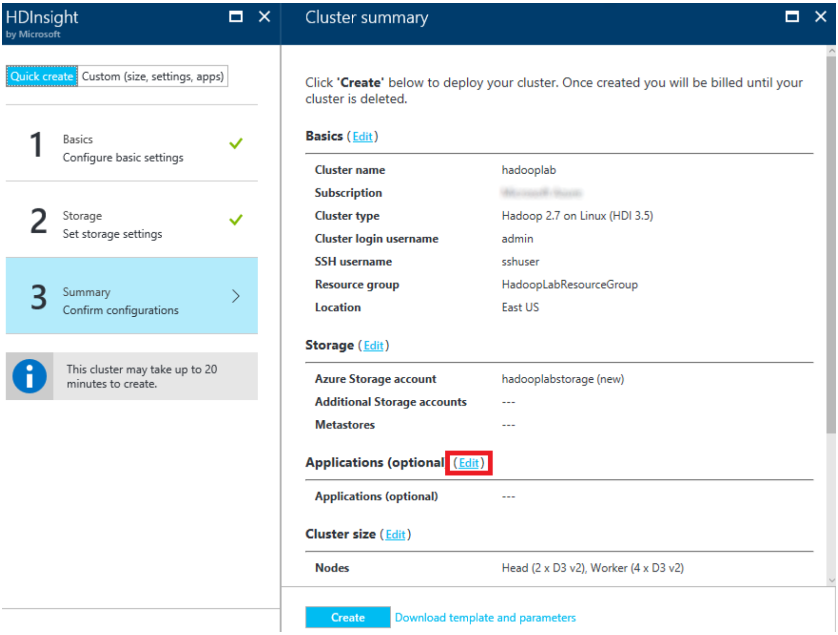
Select a SQL database for Hive
No database in ukwest for subscription.

Select a SQL database for Oozie
No database in ukwest for subscription.

Next

Fig. 1.4 – Entering cluster-storage settings

6. Click the **Edit** link next to **Applications**.



Click **'Create'** below to deploy your cluster. Once created you will be billed until your cluster is deleted.

Basics (Edit)

Cluster name	hadooplab
Subscription	[redacted]
Cluster type	Hadoop 2.7 on Linux (HDI 3.5)
Cluster login username	admin
SSH username	sshuser
Resource group	HadoopLabResourceGroup
Location	East US

Storage (Edit)

Azure Storage account	hadooplabstorage (new)
Additional Storage accounts	---
Metastores	---

Applications (optional) (Edit)

Applications (optional)	---

Cluster size (Edit)

Nodes	Head (2 x D3 v2), Worker (4 x D3 v2)

Fig. 1.5 – Editing installed applications

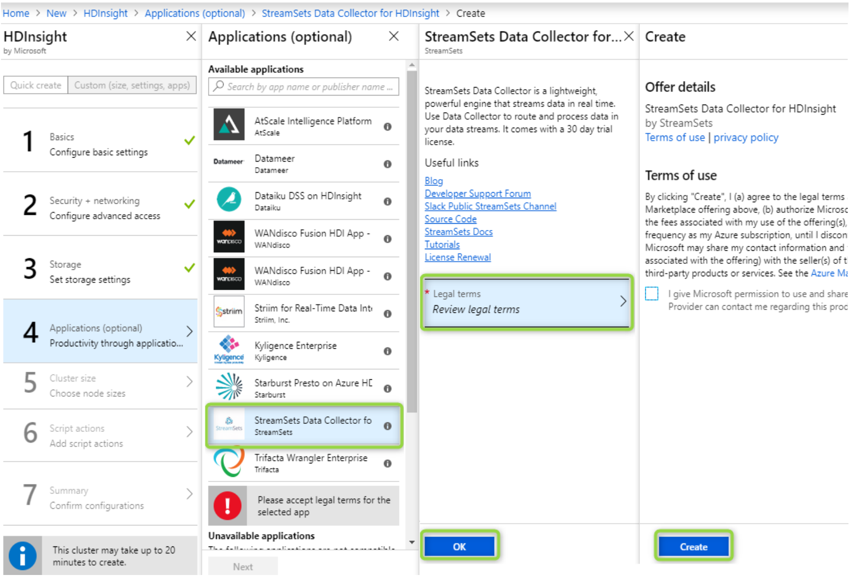
7. Select **StreamSets Data Collector for HI** (see Fig. 1.6). Then click **Legal terms** in the “Streamsets Data Collector” blade and the **Purchase** button in the “Purchase” blade. Finish up by clicking **OK** at the bottom of the “Streamsets Data Collector” blade and **Next** at the bottom of the “Applications” blade.

8. Select Click **Next** at the bottom of the “Cluster size” blade. By accepting the defaults, you are deploying a cluster that contains two head nodes and four worker nodes.

If you would like to change the number of nodes or the sizes of the nodes, you may do so in this blade. Your settings here will define amount of money you will be charged (see Fig. 1.7).

9. Click **Next** at the bottom of the “Advanced settings” blade. It is optional, so, simple skip this.

10. Review the cluster settings and make sure everything is correct. Then click **Create** to begin deploying the cluster (see Fig. 1.8).



Home > New > HDInsight > Applications (optional) > StreamSets Data Collector for HDInsight > Create

HDInsight by Microsoft

Quick create | Custom (size, settings, apps)

1 Basics – Configure basic settings ✓

2 Security + networking – Configure advanced access ✓

3 Storage – Set storage settings ✓

4 Applications (optional) – Productivity through applicatio...

5 Cluster size – Choose node sizes

6 Script actions – Add script actions

7 Summary – Confirm configurations

This cluster may take up to 20 minutes to create.

Applications (optional)

Available applications

Search by app name or publisher name ...

- AtScale Intelligence Platform – AtScale
- Datameer – Datameer
- Dataiku DSS on HDInsight – Dataiku
- WANdisco Fusion HDI App - – WANdisco
- WANdisco Fusion HDI App - – WANdisco
- Striim for Real-Time Data Int – Striim, Inc.
- Kyligence Enterprise – Kyligence
- Starburst Presto on Azure HD – Starburst
- StreamSets Data Collector fo – StreamSets
- Trifacta Wrangler Enterprise – Trifacta

Please accept legal terms for the selected app

Unavailable applications

Next

StreamSets Data Collector for... StreamSets

StreamSets Data Collector is a lightweight, powerful engine that streams data in real time. Use Data Collector to route and process data in your data streams. It comes with a 30 day trial license.

Useful links

Blog
Developer Support Forum
Slack Public StreamSets Channel
Source Code
StreamSets Docs
Tutorials
License Renewal

* Legal terms – Review legal terms

OK

Create

Offer details

StreamSets Data Collector for HDInsight
by StreamSets
Terms of use | privacy policy

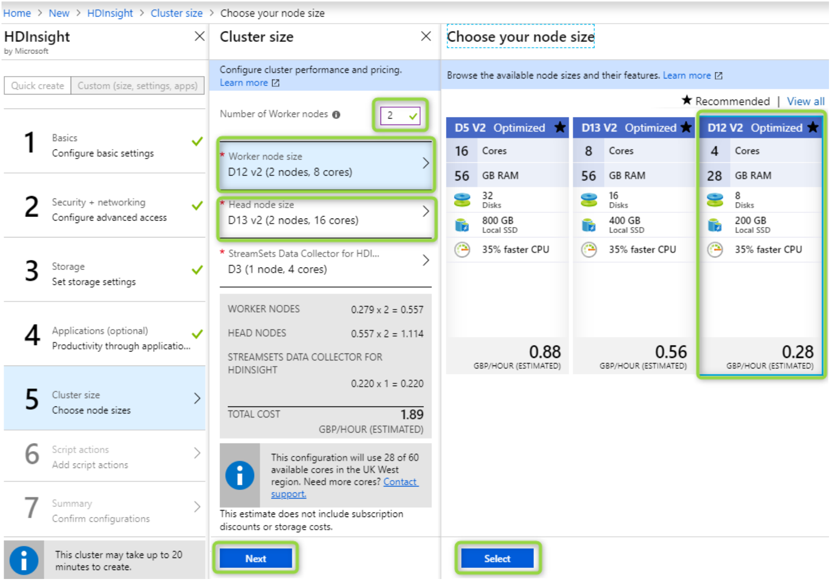
Terms of use

By clicking "Create", I (a) agree to the legal terms Marketplace offering above, (b) authorize Microso the fees associated with my use of the offering(s), frequency as my Azure subscription, until I discon Microsoft may share my contact information and associated with the offering) with the seller(s) of t third-party products or services. See the Azure Ma

□ I give Microsoft permission to use and share Provider can contact me regarding this proc

Create

Fig. 1.6 – Installing Streamsets Data Collector



Home > New > HDInsight > Cluster size > Choose your node size

HDInsight by Microsoft

Quick create | Custom (size, settings, apps)

1 Basics – Configure basic settings ✓

2 Security + networking – Configure advanced access ✓

3 Storage – Set storage settings ✓

4 Applications (optional) – Productivity through applicatio... ✓

5 Cluster size – Choose node sizes

6 Script actions – Add script actions

7 Summary – Confirm configurations

This cluster may take up to 20 minutes to create.

Cluster size

Configure cluster performance and pricing. Learn more

Number of Worker nodes: 2 ✓

* Worker node size – D12 v2 (2 nodes, 8 cores)

* Head node size – D13 v2 (2 nodes, 16 cores)

* StreamSets Data Collector for HDI... – D3 (1 node, 4 cores)

WORKER NODES	0.279 x 2 = 0.557
HEAD NODES	0.557 x 2 = 1.114
STREAMSETS DATA COLLECTOR FOR HDINSIGHT	0.220 x 1 = 0.220
TOTAL COST	1.89 GBP/HOUR (ESTIMATED)

This configuration will use 28 of 60 available cores in the UK West region. Need more cores? Contact support.

This estimate does not include subscription discounts or storage costs.

Next

Choose your node size

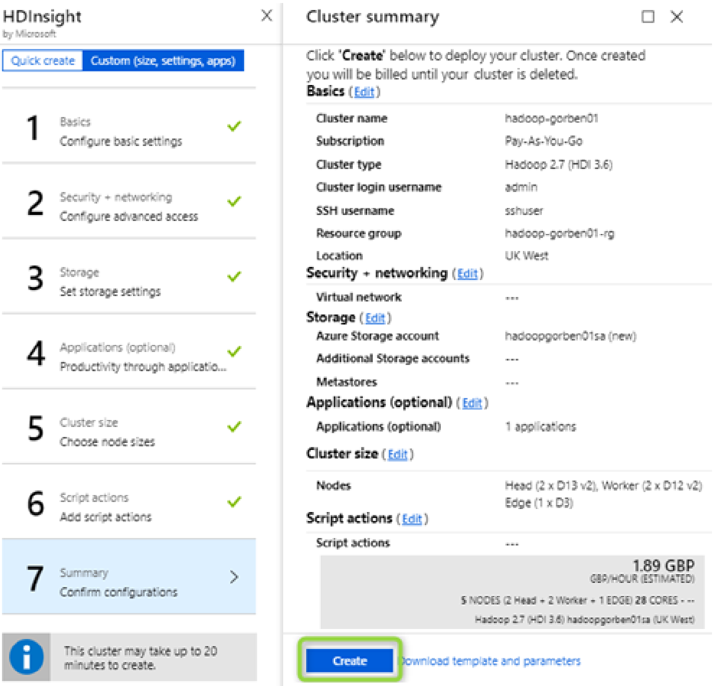
Browse the available node sizes and their features. Learn more

★ Recommended | View all

D5 V2 Optimized ★	D13 V2 Optimized ★	D12 V2 Optimized ★
16 Cores	8 Cores	4 Cores
56 GB RAM	56 GB RAM	28 GB RAM
32 Disks	16 Disks	8 Disks
800 GB Local SSD	400 GB Local SSD	200 GB Local SSD
35% faster CPU	35% faster CPU	35% faster CPU
0.88 GBP/HOUR (ESTIMATED)	0.56 GBP/HOUR (ESTIMATED)	0.28 GBP/HOUR (ESTIMATED)

Select

Fig. 1.7 – Specifying the cluster size



HDInsight
by Microsoft

Quick create | Custom (size, settings, apps)

1 Basics
Configure basic settings ✓

2 Security + networking
Configure advanced access ✓

3 Storage
Set storage settings ✓

4 Applications (optional)
Productivity through applicatio... ✓

5 Cluster size
Choose node sizes ✓

6 Script actions
Add script actions ✓

7 Summary
Confirm configurations >

This cluster may take up to 20 minutes to create.

Cluster summary

Click '**Create**' below to deploy your cluster. Once created you will be billed until your cluster is deleted.

Basics (Edit)

Cluster name	hadoop-gorben01
Subscription	Pay-As-You-Go
Cluster type	Hadoop 2.7 (HDI 3.6)
Cluster login username	admin
SSH username	sshuser
Resource group	hadoop-gorben01-rg
Location	UK West

Security + networking (Edit)

Virtual network	---

Storage (Edit)

Azure Storage account	hadoopgorben01sa (new)
Additional Storage accounts	---
Metastores	---

Applications (optional) (Edit)

Applications (optional)	1 applications

Cluster size (Edit)

Nodes	Head (2 x D13 v2), Worker (2 x D12 v2) Edge (1 x D3)

Script actions (Edit)

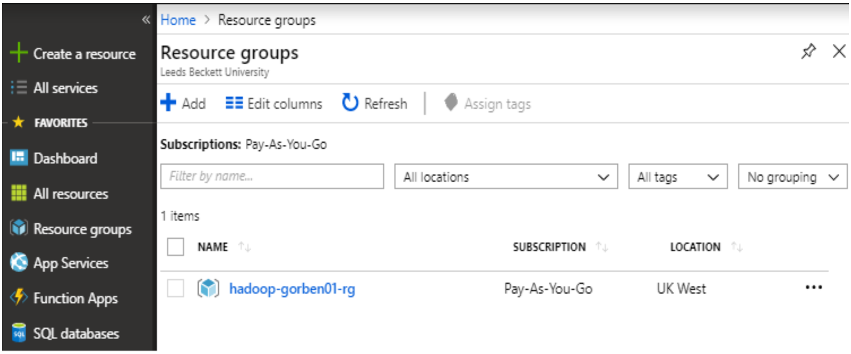
Script actions	---

1.89 GBP
GBP/HOUR (ESTIMATED)
5 NODES (2 Head + 2 Worker + 1 EDGE) 28 CORES - --
Hadoop 2.7 (HDI 3.6) hadoopgorben01sa (UK West)

Create | Download template and parameters

Fig. 1.8 – Reviewing cluster settings

11. Click **Resource groups** in the ribbon on the left side of the portal, and then click the resource group created for the HDInsight cluster.



Home > Resource groups

Resource groups
Leeds Beckett University

+ Add | Edit columns | Refresh | Assign tags

Subscriptions: Pay-As-You-Go

Filter by name... | All locations | All tags | No grouping

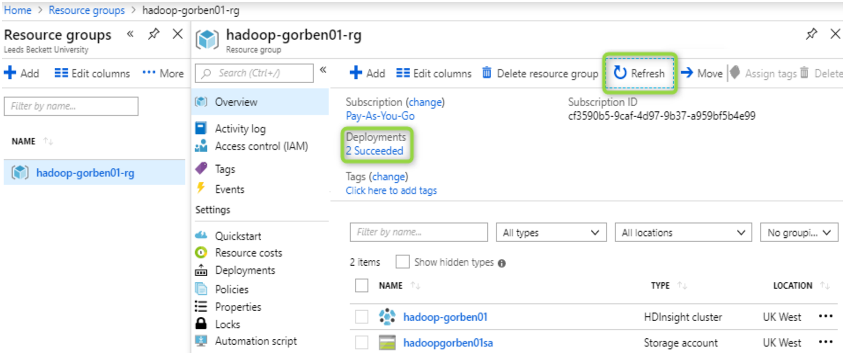
1 items

□ NAME	SUBSCRIPTION	LOCATION	
□ hadoop-gorben01-rg	Pay-As-You-Go	UK West	•••

Fig. 1.9 – Opening the resource group

12. Deploying an HDInsight cluster can take 15 minutes or more. Wait until “Deploying” changes to “Succeeded”, indicating that the cluster has been deployed. You can click the **Refresh** button at the top of the blade to refresh the deployment status.

Wait for the deployment to finish, and then proceed to the next exercise.



Subscription (change)
Pay-As-You-Go

Subscription ID
cf3590b5-9caf-4d97-9b37-a959bf5b4e99

Deployments
2 Succeeded

Tags (change)
Click here to add tags

2 items

NAME	TYPE	LOCATION
hadoop-gorben01	HDInsight cluster	UK West
hadoopgorben01sa	Storage account	UK West

Fig. 1.10 – Viewing the deployment status

1.4 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, figure / scheme of testing environment, procedures
- (R): narrate (like a story), tables, indicate final results;
- (D): answers on discovery questions, explanation of anomalies, conclusion / summary

1.5 Test questions:

1. What is Microsoft Azure?
2. What is Apache Hadoop? Which tools are included into Hadoop?
3. What is HDInsight? How to deploy HDInsight cluster on Microsoft Azure?

1.6 Recommended literature:

1. T. White, ″Hadoop: The Definitive Guide, 4th Edition″, O’Reilly Media, Inc., 2015. 756 p. [Online]. Available: library/view/hadoop-the-definitive/9781491901687/. [Accessed: 22-June-2019].

2. M. Zaharia, B. Chambers, ″Spark: The Definitive Guide″, O’Reilly Media, Inc., 2018. 606 p. [Online]. Available : library/view/spark-the-definitive/9781491912201/. [Accessed: 22-June-2019].

3. A. Chauhan, V. Fontama, M. Hart, W.-H. Tok, B. Woody, ″Introducing Microsoft Azure HDInsight″, Microsoft Press, 2014. 94 p. [Online]. Available: https://download.microsoft.com/download/e/7/b/e7b25440-1569-40b5-989e-3951fc178214/microsoft_press_ebook_introducing_ hdinsight_pdf.pdf . [Accessed: 22- June-2019].

4. ″Microsoft Azure Tutorial″. *Edureka.co*, 2019. [Online]. Available: https://www.edureka.co/blog/ microsoft-azure-tutorial. [Accessed: 22- June-2019].

Laboratory work 2
CONNECT TO THE CLUSTER VIA SSH

Goal and objectives: this work is designed to acquaint students with the process of deploying and running Hadoop clusters provisioned by HDInsight on Linux VMs to process big data.

Learning objectives: in this lab, you will learn how to connect to the HDInsight cluster using SSH.

Practical tasks: configuring HDInsight Hadoop clusters from a Windows or Linux PC using SSH.

Exploring tasks: no

Setting up

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual and listed in the Recommended literature section;
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

Prerequisites

The following are required to complete this hands-on lab:

– An active Microsoft Azure subscription. If you don't have one, sign up for a free trial or use your pay-as-you-go subscription;

– PuTTY (Windows users only). Install the latest full package using the MSI installer.

2.1 Synopsis

In this lab work you will learn how to connect to the Hadoop cluster deployed on Microsoft Azure Cloud remotely from a Windows or Linux PC using SSH.

2.2 Brief theoretical information

Before you can run jobs on the Hadoop cluster, you need to open an SSH connection to it so you can execute commands on the cluster. In this exercise, you will remote into the cluster via SSH using the **ssh** command if you are running macOS or Linux, or PuTTY if you are running Windows. If you are a Windows user and haven't installed PuTTY, take the time to install it now.

*If you are running Windows, skip to Step 2**. Otherwise, proceed to Step 1.***

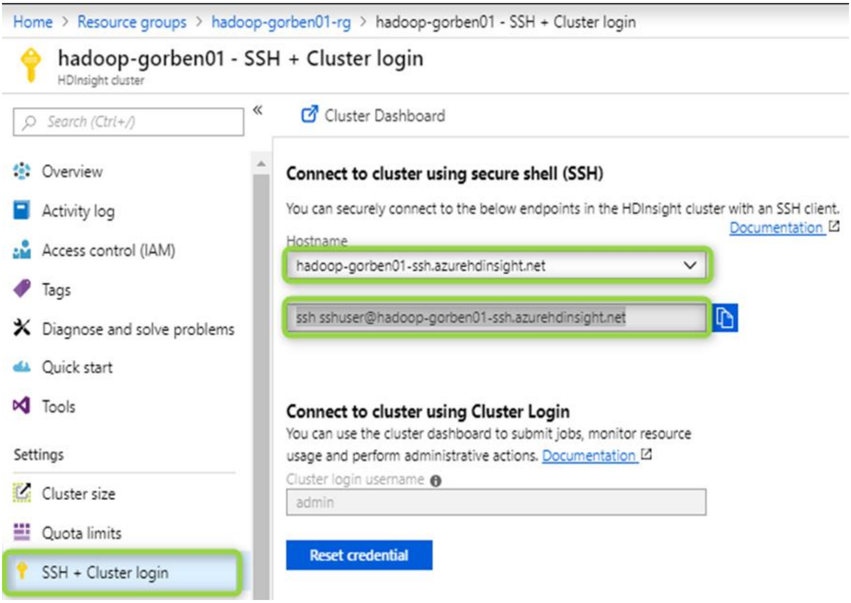
2.3 Execution order and discovery questions

1. ***Linux and macOS users only***: Open SSH+Cluster login tab of your Hadoop cluster and copy the connection string (see Fig. 2.1).

Open a terminal window so you can use the **ssh** command to establish a connection. Execute the following command in the terminal window, replacing *clustername* with the cluster name you entered in Exercise 1, Step 3 (or simply paste the connection string copied from SSH+Cluster login tab):

```
ssh sshuser@clustername-ssh.azurehdinsight.net
```

Enter the SSH password *Azure4Research!* (or the password you specified) when prompted. **Now proceed to the next lab**.



Home > Resource groups > hadoop-gorben01-rg > hadoop-gorben01 - SSH + Cluster login

hadoop-gorben01 - SSH + Cluster login
HDInsight cluster

Connect to cluster using secure shell (SSH)

You can securely connect to the below endpoints in the HDInsight cluster with an SSH client. Documentation

Hostname: hadoop-gorben01-ssh.azurehdinsight.net

ssh sshuser@hadoop-gorben01-ssh.azurehdinsight.net

Connect to cluster using Cluster Login

You can use the cluster dashboard to submit jobs, monitor resource usage and perform administrative actions. Documentation

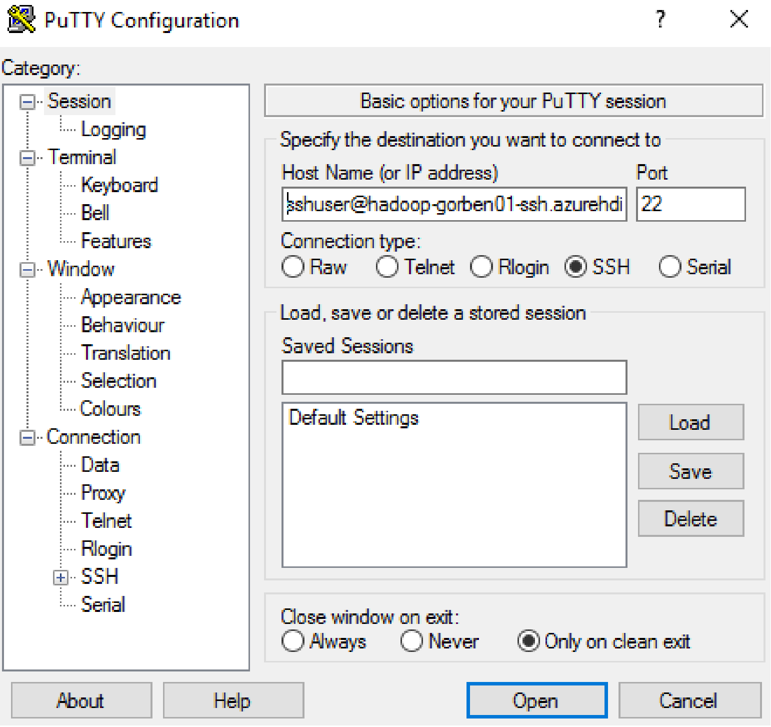
Cluster login username: admin

Fig. 2.1 – Establishing a connection with PuTTY

2. ***Windows users only***: Start PuTTY. In the **Host Name (or IP address)** field, type "sshuser@*clustername*-ssh.azurehdinsight.net" without quotation marks, replacing *clustername* with the cluster name you entered in Exercise 1, Step 3. Then click the **Open** button to open an SSH connection.

> *Because this is the first time you have connected to the master node, you will be prompted with a warning dialog asking if you trust this host. Since the host is one you created, click* ***Yes****.*

A PuTTY terminal window will appear and prompt you for a password. Enter the SSH password *Azure4Research!* (or the password you specified when you created the cluster) and press **Enter**.



Host Name (or IP address): sshuser@hadoop-gorben01-ssh.azurehdi

Port: 22

Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

Saved Sessions:

Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit

Fig. 2.2 – Establishing a connection with PuTTY

2.4 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, figure / scheme of testing environment, procedures
- (R): narrate (like a story), tables, indicate final results;
- (D): answers on discovery questions, explanation of anomalies, conclusion / summary

2.5 Test questions:

1. What is SSH?
2. How to connect to a Linux machine from a Windows PC via SSH?
3. Which tools are included into PuTTY.

2.6 Recommended literature:

1. T. White, ″Hadoop: The Definitive Guide, 4th Edition″, O’Reilly Media, Inc., 2015. 756 p. [Online]. Available: library/view/hadoop-the-definitive/9781491901687/. [Accessed: 22-June-2019].
2. M. Zaharia, B. Chambers, ″Spark: The Definitive Guide″, O’Reilly Media, Inc., 2018. 606 p. [Online]. Available : library/view/spark-the-definitive/9781491912201/. [Accessed: 22-June-2019].
3. A. Chauhan, V. Fontama, M. Hart, W.-H. Tok, B. Woody, ″Introducing Microsoft Azure HDInsight″, Microsoft Press, 2014. 94 p. [Online]. Available: https://download.microsoft.com/download/e/7/b/e7b25440-1569-40b5-989e-3951fc178214/microsoft_press_ebook_introducing_ hdinsight_pdf.pdf . [Accessed: 22- June-2019].
4. ″Microsoft Azure Tutorial″. *Edureka.co*, 2019. [Online]. Available: https://www.edureka.co/blog/ microsoft-azure-tutorial. [Accessed: 22- June-2019].

Laboratory work 3
ANALYZE AN APACHE LOG FILE WITH HIVE

Goal and objectives: this work is designed to acquaint students with the process of deploying and running Hadoop clusters provisioned by HDInsight on Linux VMs to process big data.

Learning objectives: in this lab, you will learn how to use Hive on the cluster to query datasets.

Practical tasks: using Hive on HDInsight Hadoop clusters to query and analyze data.

Exploring tasks: analyzing log files using Hive.

Setting up

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual and listed in the Recommended literature section;
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

Prerequisites

The following are required to complete this hands-on lab:

– An active Microsoft Azure subscription. If you don't have one, sign up for a free trial or use your pay-as-you-go subscription;

– PuTTY (Windows users only). Install the latest full package using the MSI installer.

3.1 Synopsis

In this lab work you will learn how to use Hadoop Hive (a part of Azure HDInsight) to analyze textual log files.

3.2 Brief theoretical information:

In this task, students will use *Apache Hive* and the HiveQL query language to query a sample Apache log4j log file that was created along with the cluster.

Apache Hive is a data-warehouse infrastructure built on top of Hadoop that facilitates summarizing, querying, and analyzing data. It supports a SQL-like interface for querying data stores that integrate with Hadoop, and it allows you to project structure onto data that lacks structure.

3.3 Execution order and discovery questions:

1. In the terminal window you opened in the previous exercise, start the Hive command-line interface by executing the following command:

```
   hive
```

2. Wait for a Hive prompt to appear. Then enter the following commands to create a new table named "log4jlogs" using sample data stored in a blob that was created along with your cluster.

You can paste a command into a PuTTY terminal window by right-clicking in the window.

```
   DROP TABLE log4jLogs;

   CREATE EXTERNAL TABLE log4jLogs(t1 string, t2
string, t3 string, t4 string, t5 string, t6 string,
t7 string) ROW FORMAT DELIMITED FIELDS TERMINATED BY
' ' STORED AS TEXTFILE LOCATION
'wasb:///example/data/';

   SELECT t4 AS sev, COUNT(*) AS cnt FROM log4jLogs
WHERE t4 = '[ERROR]' GROUP BY t4;
```

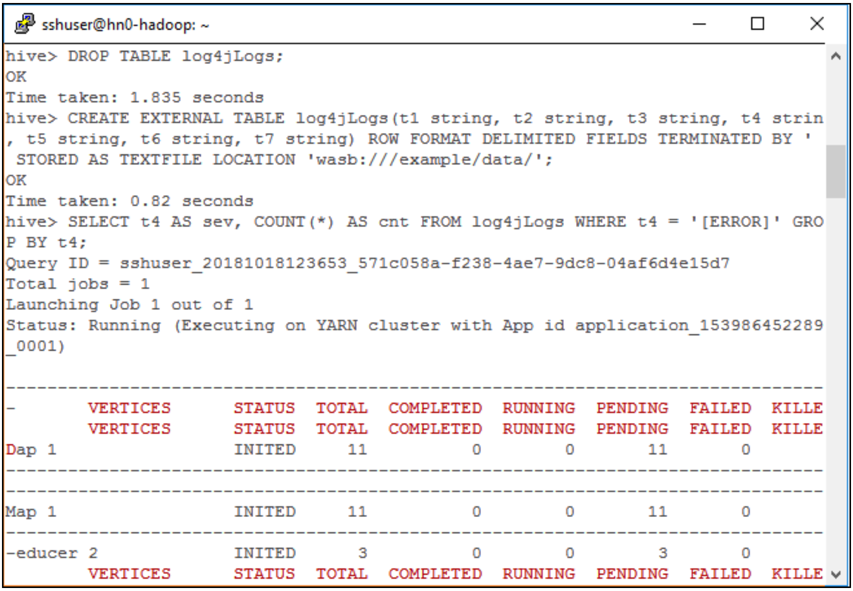
*The **DROP TABLE** command removes any existing table named “log4jLogs”. **CREATE EXTERNAL TABLE** creates a new external table. External tables store only the table definitions in Hive; the data is left in the original location. **STORED AS TEXTFILE LOCATION** tells Hive that the data is stored in a text file and where the file is located. “wasb” is the protocol prefix; it stands for “Windows Azure Storage Blob”. (HDInsight transparently maps HDFS to Azure blob storage.) Finally, the **SELECT** statement counts all the rows in which column t4 contains the value “[ERROR]”.*

After the final command is entered, you will see statements similar to the following at the end of the output (see Fig. 3.1).

Note that the output contains "*[ERROR] 3*" as there are three rows that contain this value.

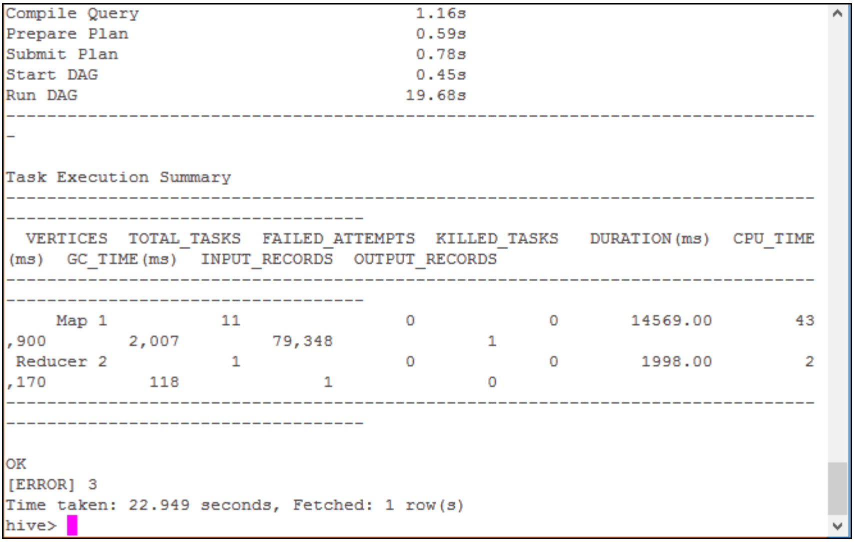
3. Execute the following commands to create a new internal table named "*errorLogs*" and see the output (Fig. 3.2):

```
   CREATE TABLE IF NOT EXISTS errorLogs (t1 string, t2 string, t3 string, t4 string, t5 string, t6 string, t7 string) STORED AS ORC;
   INSERT OVERWRITE TABLE errorLogs SELECT t1, t2, t3, t4, t5, t6, t7 FROM log4jLogs WHERE t4 = '[ERROR]';
```



```
hive> DROP TABLE log4jLogs;
OK
Time taken: 1.835 seconds
hive> CREATE EXTERNAL TABLE log4jLogs(t1 string, t2 string, t3 string, t4 strin
, t5 string, t6 string, t7 string) ROW FORMAT DELIMITED FIELDS TERMINATED BY '
 STORED AS TEXTFILE LOCATION 'wasb:///example/data/';
OK
Time taken: 0.82 seconds
hive> SELECT t4 AS sev, COUNT(*) AS cnt FROM log4jLogs WHERE t4 = '[ERROR]' GRO
P BY t4;
Query ID = sshuser_20181018123653_571c058a-f238-4ae7-9dc8-04af6d4e15d7
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_153986452289
_0001)

-------------------------------------------------------------------------------
-       VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLE
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLE
Dap 1                 INITED     11          0        0       11       0
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
Map 1                 INITED     11          0        0       11       0
-------------------------------------------------------------------------------
-educer 2             INITED      3          0        0        3       0
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLE
```

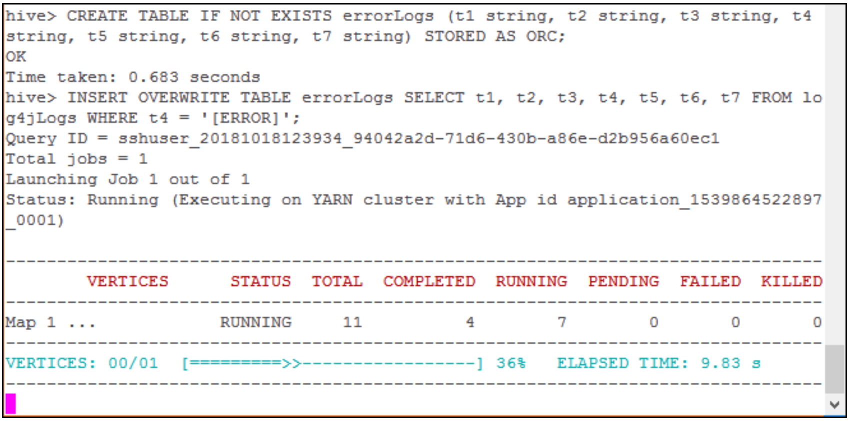
…



```
Compile Query                              1.16s
Prepare Plan                               0.59s
Submit Plan                                0.78s
Start DAG                                  0.45s
Run DAG                                   19.68s
--------------------------------------------------------------------------------
-

Task Execution Summary
--------------------------------------------------------------------------------
-----------------------------------
  VERTICES  TOTAL_TASKS  FAILED_ATTEMPTS  KILLED_TASKS   DURATION(ms)   CPU_TIME
(ms)   GC_TIME(ms)   INPUT_RECORDS  OUTPUT_RECORDS
--------------------------------------------------------------------------------
-----------------------------------
     Map 1           11                0             0        14569.00        43
,900         2,007         79,348                   1
 Reducer 2            1                0             0         1998.00         2
,170           118              1                   0
--------------------------------------------------------------------------------
-----------------------------------

OK
[ERROR] 3
Time taken: 22.949 seconds, Fetched: 1 row(s)
hive>
```

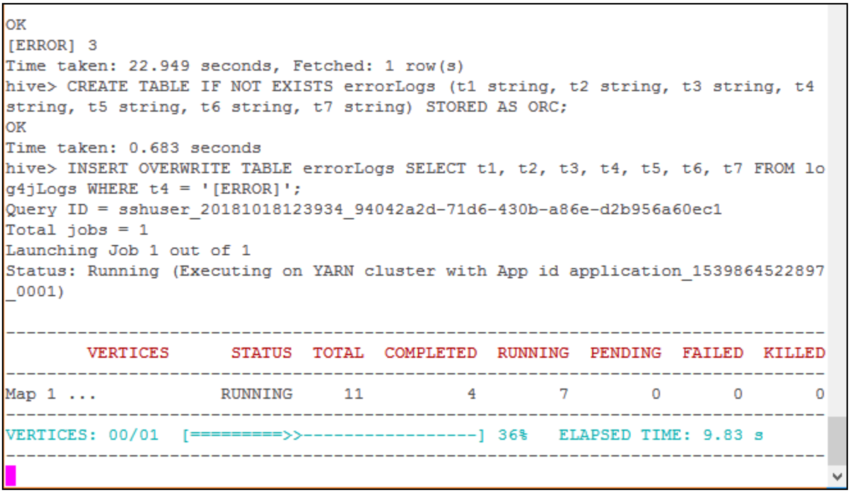
Fig. 3.1 – creating a table in Hive and executing select query

***CREATE TABLE IF NOT EXISTS** creates a table if it does not already exist. Because the EXTERNAL keyword is not specified, this is an internal table that is stored in the Hive data warehouse and is managed completely by Hive. Unlike dropping an external table, dropping an internal table deletes the underlying data as well. **STORED AS ORC** says to store the data in Optimized Row Columnar (ORC) format. This is a highly optimized and efficient format for storing Hive data. **INSERT OVERWRITE...SELECT** selects rows from the "log4jLogs" table that contain the string "[ERROR]," and then inserts them into the "errorLogs" table.*



```
hive> CREATE TABLE IF NOT EXISTS errorLogs (t1 string, t2 string, t3 string, t4
string, t5 string, t6 string, t7 string) STORED AS ORC;
OK
Time taken: 0.683 seconds
hive> INSERT OVERWRITE TABLE errorLogs SELECT t1, t2, t3, t4, t5, t6, t7 FROM lo
g4jLogs WHERE t4 = '[ERROR]';
Query ID = sshuser_20181018123934_94042a2d-71d6-430b-a86e-d2b956a60ec1
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1539864522897
_0001)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ...             RUNNING     11          4        7        0       0       0
--------------------------------------------------------------------------------
VERTICES: 00/01  [=========>>-----------------] 36%   ELAPSED TIME: 9.83 s
--------------------------------------------------------------------------------
```

…



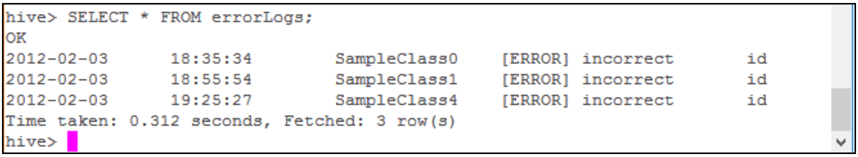
```
OK
[ERROR] 3
Time taken: 22.949 seconds, Fetched: 1 row(s)
hive> CREATE TABLE IF NOT EXISTS errorLogs (t1 string, t2 string, t3 string, t4
string, t5 string, t6 string, t7 string) STORED AS ORC;
OK
Time taken: 0.683 seconds
hive> INSERT OVERWRITE TABLE errorLogs SELECT t1, t2, t3, t4, t5, t6, t7 FROM lo
g4jLogs WHERE t4 = '[ERROR]';
Query ID = sshuser_20181018123934_94042a2d-71d6-430b-a86e-d2b956a60ec1
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1539864522897
_0001)

--------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ...             RUNNING     11          4        7        0       0       0
--------------------------------------------------------------------------------
VERTICES: 00/01  [=========>>-----------------] 36%   ELAPSED TIME: 9.83 s
--------------------------------------------------------------------------------
```

Fig. 3.2 – Inserting data into Hive’s table resulted from SELECT query

4. The final step is to verify that only rows containing “[ERROR]” in column t4 were stored in the “errorLogs” table. To do that, use the following command to return all rows from “errorLogs”:

```
SELECT * FROM errorLogs;
```

The output will look like this:



```
hive> SELECT * FROM errorLogs;
OK
2012-02-03      18:35:34        SampleClass0    [ERROR] incorrect       id
2012-02-03      18:55:54        SampleClass1    [ERROR] incorrect       id
2012-02-03      19:25:27        SampleClass4    [ERROR] incorrect       id
Time taken: 0.312 seconds, Fetched: 3 row(s)
hive>
```

Fig. 3.3 – Executing SELECT query in Hive

5. Use the following command to close the Hive session:

```
quit;
```

Leave your SSH session open, because you will use it again in the next exercise.

3.4 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, figure / scheme of testing environment, procedures
- (R): narrate (like a story), tables, indicate final results;
- (D): answers on discovery questions, explanation of anomalies, conclusion / summary

3.5 Test questions:

1. What is Apache Hive? What are differences between Hive and SQL?
2. What is optimized Row Columnar (ORC) format?
3. What is Windows Azure Storage Blob (wasb)?

3.6 Recommended literature:

1. T. White, ″Hadoop: The Definitive Guide, 4th Edition″, O’Reilly Media, Inc., 2015. 756 p. [Online]. Available: library/view/hadoop-the-definitive/9781491901687/. [Accessed: 22-June-2019].

2. M. Zaharia, B. Chambers, ″Spark: The Definitive Guide″, O’Reilly Media, Inc., 2018. 606 p. [Online]. Available : library/view/spark-the-definitive/9781491912201/. [Accessed: 22-June-2019].

3. A. Chauhan, V. Fontama, M. Hart, W.-H. Tok, B. Woody, ″Introducing Microsoft Azure HDInsight″, Microsoft Press, 2014. 94 p. [Online]. Available: https://download.microsoft.com/download/e/7/b/e7b25440-1569-40b5-989e-3951fc178214/microsoft_press_ebook_introducing_ hdinsight_pdf.pdf . [Accessed: 22- June-2019].

4. ″Microsoft Azure Tutorial″. *Edureka.co*, 2019. [Online]. Available: https://www.edureka.co/blog/ microsoft-azure-tutorial. [Accessed: 22- June-2019].

Laboratory work 4
USE MAP-REDUCE TO ANALYZE A TEXT FILE WITH PYTHON

Goal and objectives: this work is designed to acquaint students with the process of deploying and running Hadoop clusters provisioned by HDInsight on Linux VMs to process big data.

Learning objectives: in this lab, you will learn how to use Python to perform MapReduce operations.

Practical tasks: performing MapReduce operations coded in Python or other languages on HDInsight cluster to analyze text file.

Exploring tasks: using MapReduce to count the appearance of each word in a set of documents.

Setting up

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual and listed in the Recommended literature section;
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

Prerequisites

The following are required to complete this hands-on lab:

– An active Microsoft Azure subscription. If you don't have one, sign up for a free trial or use your pay-as-you-go subscription;

– PuTTY (Windows users only). Install the latest full package using the MSI installer.

Resources

You will need to save program codes given in the lab as Python’s.py scripts and copy them into a folder on your hard disk.

4.1 Synopsis

In this lab work you will learn how to use MapReduce operations to count the appearance of each word in a set of documents.

4.2 Brief theoretical information:

One of the most important algorithms introduced in recent years is Google’s MapReduce, which facilitates the processing of very large data sets. MapReduce is a two-stage algorithm that relies on a pair of functions: the map function, which transforms a set of input data to produce a result, and the reduce function, which reduces the results of a map to a scalar value. What makes MapReduce so relevant for big data is that operations can be executed in parallel and independent of the data source. The parallelism facilitates handling massive amounts of data, and the data-source independence means you are not locked into a particular data store such as MySQL or Microsoft SQL Server.

HDInsight, with its underlying Hadoop implementation, allows you to write MapReduce functions in Java, Python, C#, and even Apache Pig. In this exercise, you will use Python since it is widely used in the data-processing community. The Python code reads a text file and counts the frequency of the words in it.

4.3 Execution order and discovery questions:

1. Before you start, take a moment to read over the Python code for the ***mapper*** you will be using:

```
#!/usr/bin/env python
"""mapper.py"""
import sys
# input comes from STDIN (standard input)
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()
    # split the line into words
    words = line.split()
    # increase counters
    for word in words:
    # write the results to STDOUT (standard output);
    # what we output here will be the input for the
    # Reduce step, i.e. the input for reducer.py
    #
    # tab-delimited; the trivial word count is 1
        print '%s\t%s' % (word, 1)
```

The mapper reads a file from standard input (STDIN) and outputs each word in the file, followed by a tab character and the value 1, on a separate line.

2. Now take a moment to examine the ***reducer***:

```
#!/usr/bin/env python
"""reducer.py"""
from operator import itemgetter
import sys
current_word = None
current_count = 0
word = None
# input comes from STDIN
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()
    # parse the input we got from mapper.py
    word, count = line.split('\t', 1)
    # convert count (currently a string) to int
    try:
        count = int(count)
    except ValueError:
        # count was not a number, so silently
        # ignore/discard this line
        continue
# this IF-switch works because Hadoop sorts map output
# by key (here: word) before it is passed to the reducer
    if current_word == word:
        current_count += count
    else:
        if current_word:
            # write result to STDOUT
            print '%s\t%s' % (current_word,
current_count)
        current_count = count
        current_word = word
# do not forget to output the last word if needed!
if current_word == word:
    print '%s\t%s' % (current_word, current_count)
```

The reducer reads each word output by the mapper, looks it up in the list of word groups it compiles, and adds the number of instances found to the total number of instances, writing the data to standard output (STDOUT).

3. The two Python scripts containing the mapper and the reducer are provided for you in the resources that accompany this lab. The next step is to copy the two files, which are named **mapper.py** and **reducer.py**, to the cluster. **If you're using Windows, skip to Step 5**. Otherwise, proceed to the next step.

4. **Linux and macOS users only**: Open a terminal window and navigate to the directory where you copied the lab resources. Then execute the following command to copy **mapper.py** and **reduce.py** to the HDInsight cluster, replacing *clustername* with the cluster name you specified in Exercise 1, Step 3. When prompted for a password, enter the cluster's SSH password ("Azure4Research!").

```
scp *.py sshuser@clustername-ssh.azurehdinsight.net:
```

Now skip to Step 6. Step 5 is for Windows users only.

5. **Windows users only**: Open a Command Prompt window and navigate to the directory where you copied the lab resources. Then execute the following command to copy **mapper.py** and **reduce.py** to the HDInsight cluster, replacing *clustername* with the cluster name you specified in Exercise 1, Step 3. When prompted for a password, enter the cluster's SSH password ("Azure4Research!").

```
pscp *.py sshuser@clustername-ssh.azurehdinsight.net:
```

pscp.exe is part of PuTTY. This command assumes that pscp.exe is in the PATH. If it's not, preface the command with the path to pscp.exe.

6. Return to the SSH session that you established in Exercise 2 (if you closed the session, or if it timed out, follow the instructions in Exercise 2 to establish a new SSH connection).

7. To be certain that the Python files you uploaded contain Linux-style line endings ("/r" rather than "/r/n"), execute the following commands in the terminal window to install and run the dos2unix conversion program:

```
sudo apt-get install dos2unix
dos2unix -k -o *.py
```

8. Now execute the following command to run the Hadoop job:

```
hadoop jar /usr/hdp/current/hadoop-mapreduce-
client/hadoop-streaming.jar -files
mapper.py,reducer.py -mapper mapper.py -reducer
reducer.py -input
wasb:///example/data/gutenberg/davinci.txt -output
wasb:///example/wordcountout
```

There is a lot going on in that command. Here is a quick synopsis of each part:

hadoop: Launches the Hadoop program

jar /usr/hdp/current/hadoop-mapreduce-client/hadoop-streaming.jar: Tells Hadoop you want to run a specific jar (Java ARchive). In this case, it is the program that interfaces Hadoop with the MapReduce code.

-files mapper.py,reducer.py: Specifies that these files should be copied to all the nodes in the cluster

-mapper mapper.py: Specifies which file contains the mapper

-reducer reducer.py: Specifies which file contains the reducer

-input wasb:///example/data/gutenberg/davinci.txt: Specifies the input file. In this case, it is the file named davinci.txt, which was copied into blob storage when the cluster was created.

-output wasb:///example/wordcountout: Specifies the output file

9. To see the files that Hadoop created, execute the following command:

```
hadoop fs -ls /example/wordcountout
```

The output will show that two files were created:

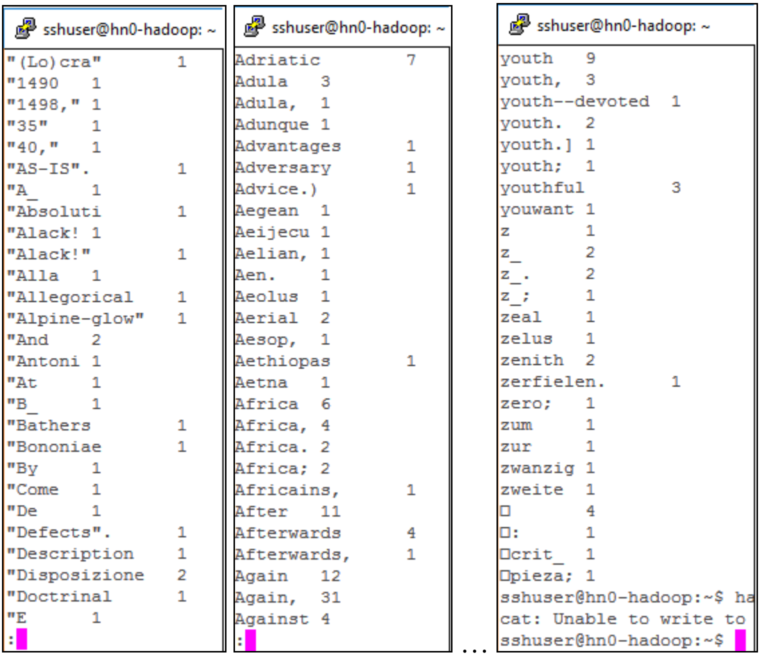
```
Found 2 items
-rw-r--r--   1 sshuser supergroup          0 2016-
12-09 14:04 /example/wordcountout/_SUCCESS
-rw-r--r--   1 sshuser supergroup     337623 2016-
12-09 14:04 /example/wordcountout/part-00000
```

The _SUCCESS file, which is zero bytes in length, indicates that the job was a success. The part-00000 file contains the list of words and their counts. To view the contents of that file, use the following command:

```
hadoop fs -cat /example/wordcountout/part-00000 | less
```

As you can see, mapper.py does not separate words that contain punctuation characters. It is left as an exercise for you to consider how you would change the code to strip off extra punctuation marks when harvesting words.

*Use **PgUP**, **PgDn** to go through the output. Press **q** to exit.*



```
"(Lo)cra"       1
"1490   1
"1498," 1
"35"    1
"40,"   1
"AS-IS".        1
"A_     1
"Absoluti       1
"Alack! 1
"Alack!"        1
"Alla   1
"Allegorical    1
"Alpine-glow"   1
"And    2
"Antoni 1
"At     1
"B_     1
"Bathers        1
"Bononiae       1
"By     1
"Come   1
"De     1
"Defects".      1
"Description    1
"Disposizione   2
"Doctrinal      1
"E      1
:
```

```
Adriatic        7
Adula   3
Adula,  1
Adunque 1
Advantages      1
Adversary       1
Advice.)        1
Aegean  1
Aeijecu 1
Aelian, 1
Aen.    1
Aeolus  1
Aerial  2
Aesop,  1
Aethiopas       1
Aetna   1
Africa  6
Africa, 4
Africa. 2
Africa; 2
Africains,      1
After   11
Afterwards      4
Afterwards,     1
Again   12
Again,  31
Against 4
:
```

```
youth   9
youth,  3
youth--devoted  1
youth.  2
youth.] 1
youth;  1
youthful        3
youwant 1
z       1
z_      2
z_.     2
z_;     1
zeal    1
zelus   1
zenith  2
zerfielen.      1
zero;   1
zum     1
zur     1
zwanzig 1
zweite  1
□       4
□:      1
□crit_  1
□pieza; 1
sshuser@hn0-hadoop:~$ ha
cat: Unable to write to
sshuser@hn0-hadoop:~$
```

Fig. 4.1 – MapReduce output

If you want to run the job again, you will either have to change the output directory specified in the **hadoop** command, or delete the output directory with the following command:

```
hadoop fs -rm -r /example/wordcountout
```

This exercise showed how to execute streaming MapReduce jobs with HDInsight using a widely used programming language, Python. The next and most important step is to delete the HDInsight cluster so you are not billed for it when it is not being used.

4.4 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, figure / scheme of testing environment, procedures
- (R): narrate (like a story), tables, indicate final results;
- (D): answers on discovery questions, explanation of anomalies, conclusion / summary

4.5 Test questions:

1. Describe MapReduce paradigm to process Big data. How does Map Reduce work?
2. Describe in details Python code for the ***mapper***, presented on page 17.
3. Describe in details Python code for the ***reducer***, presented on pages 17-18.
4. What is Apache Spark? What are differences between Apache Spark and the common MapReduce cluster computing paradigm?
5. What are differences between Spark and Spark SQL?

4.6 Recommended literature:

1. T. White, ″Hadoop: The Definitive Guide, 4th Edition″, O’Reilly Media, Inc., 2015. 756 p. [Online]. Available: library/view/hadoop-the-definitive/9781491901687/. [Accessed: 22-June-2019].

2. M. Zaharia, B. Chambers, ″Spark: The Definitive Guide″, O’Reilly Media, Inc., 2018. 606 p. [Online]. Available : library/view/spark-the-definitive/9781491912201/. [Accessed: 22-June-2019].

3. A. Chauhan, V. Fontama, M. Hart, W.-H. Tok, B. Woody, ″Introducing Microsoft Azure HDInsight″, Microsoft Press, 2014. 94 p. [Online]. Available: https://download.microsoft.com/download/e/7/b/e7b25440-1569-40b5-989e-3951fc178214/microsoft_press_ebook_introducing_ hdinsight_pdf.pdf . [Accessed: 22- June-2019].

4. ″Microsoft Azure Tutorial″. *Edureka.co*, 2019. [Online]. Available: https://www.edureka.co/blog/ microsoft-azure-tutorial. [Accessed: 22- June-2019].

Laboratory work 5
DELETE THE HADOOP CLUSTER

Goal and objectives: this work is designed to acquaint students with the process of deploying and running Hadoop clusters provisioned by HDInsight on Linux VMs to process big data.

Learning objectives: in this lab, you will learn how to delete a cluster when it is no longer needed.

Practical tasks: deleting a HDInsight cluster should when it is not being used to avoid unnecessary charges.

Exploring tasks: no

Setting up

- to clear the goals and mission of the research;
- to study theoretical material contained in this manual and listed in the Recommended literature section;
- to familiarize oneself with the main procedures and specify the exploration program according to defined task.

Recommended software and resources:

Prerequisites

The following are required to complete this hands-on lab:

– An active Microsoft Azure subscription. If you don't have one, sign up for a free trial or use your pay-as-you-go subscription;

5.1 Synopsis

In this lab work you will learn how to delete aAzure HDInsight cluster.

5.2 Brief theoretical information

When you are finished using an HDInsight Hadoop cluster, you should delete it because you are charged for it while it exists, regardless of whether it's doing any work. In this exercise, you will delete the resource group created in Task 1 when you created the cluster. Deleting the resource group deletes everything in it and prevents any further charges from being incurred for it.

5.3 Execution order and discovery questions

1. In the Azure Portal, open the blade for the resource group that holds your lab cluster. Then click the **Delete** button at the top of the blade.

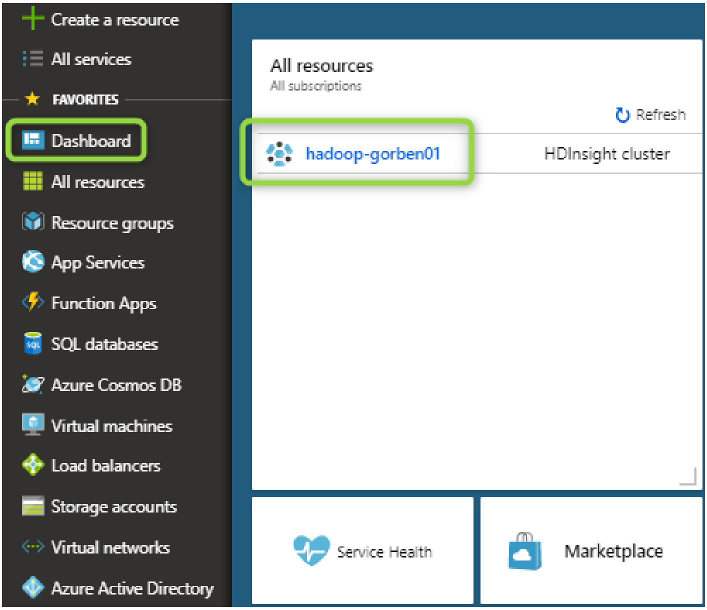
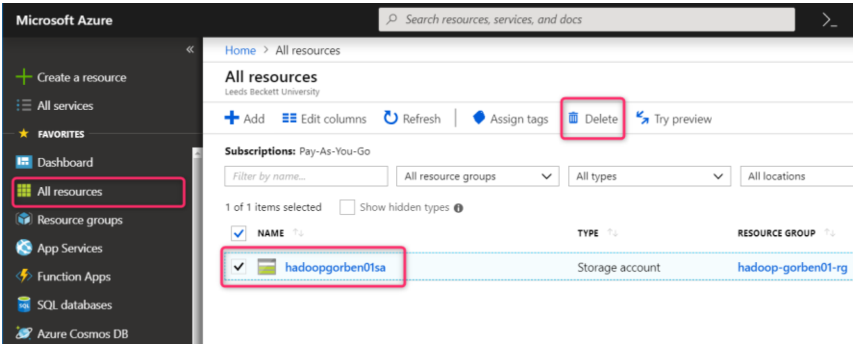


Fig. 5.1 – Deleting a resource group

2. For safety, you are required to type in the resource group's name. (Once deleted, a resource group cannot be recovered.) Type the name of the resource group. Then click the **Delete** button to remove all traces of this lab from your account.

After a few minutes, the cluster and all of its resources will be deleted. Billing stops when you click the **Delete** button, so you're not charged for the time required to delete the cluster. Similarly, bulling doesn't start until a cluster is fully and successfully deployed.

At the final stage you also need to delete your storage account.



Subscriptions: Pay-As-You-Go

1 of 1 items selected

NAME	TYPE	RESOURCE GROUP
☑ hadoopgorben01sa	Storage account	hadoop-gorben01-rg

Fig. 5.2 – Deleting a storage account

5.4 Requirements to the content of the report

Report should contain 5 sections: Introduction (I), Methods (M), Results (R), and Discussion (D)

- (I): background / theory, purpose and discovery questions
- (M): complete description of the software, and procedures which was followed in the experiment, experiment overview, figure / scheme of testing environment, procedures
- (R): narrate (like a story), tables, indicate final results;
- (D): answers on discovery questions, explanation of anomalies, conclusion / summary

5.5 Test questions:

1. What are core features of Microsoft Azure cloud platform?
2. How to run Hadoop on Microsoft Azure cloud platform?
3. How to establish SSH connection with the remote Linux/Unix server deployed at Microsoft Azure? How to establish SSH connection from Windows desktop to Linux/Unix virtual server?

5.6 Recommended literature:

1. T. White, ″Hadoop: The Definitive Guide, 4th Edition″, O'Reilly Media, Inc., 2015. 756 p. [Online]. Available: library/view/hadoop-the-definitive/9781491901687/. [Accessed: 22-June-2019].

2. M. Zaharia, B. Chambers, ″Spark: The Definitive Guide″, O'Reilly Media, Inc., 2018. 606 p. [Online]. Available :

library/view/spark-the-definitive/9781491912201/. [Accessed: 22-June-2019].

3. A. Chauhan, V. Fontama, M. Hart, W.-H. Tok, B. Woody, ″Introducing Microsoft Azure HDInsight″, Microsoft Press, 2014. 94 p. [Online]. Available: https://download.microsoft.com/download/e/7/b/e7b25440-1569-40b5-989e-3951fc178214/microsoft_press_ebook_introducing_ hdinsight_pdf.pdf . [Accessed: 22- June-2019].

4. Microsoft Azure Tutorial″. *Edureka.co*, 2019. [Online]. Available: https://www.edureka.co/blog/ microsoft-azure-tutorial. [Accessed: 22- June-2019].

APPENDIX A. TEACHING PROGRAM

DESCRIPTION OF THE MODULE

TITLE OF THE MODULE	Code
Data Science for IoT and IoE	**MS2**

Teacher(s)	Department
Coordinating: Prof. I. Skarga-Bandurova, Prof. A. Gorbenko **Others:** Assoc. Prof., Dr. T. Biloborodova, Prof. DrS. A. Sachenkko, Assoc. Prof., Dr. V.Koval, Assoc Prof. Dr. O. Tarasiuk	Computer science and Engineering, V. Dahl EUNU, Computer systems and cybersecurity, NAU “KhAI” Information Computing Systems and Control, TNEU

Study cycle	Level of the module	Type of the module
MSc	A	Full-time tuition

Form of delivery	Duration	Langage(s)
Full-time tuition	One semester	English

Prerequisites	
Prerequisites: Machine Learning, Artificial Intelligence, Modelling, Basics of Programming; Theory of algorithms	**Co-requisites (if necessary):**

Credits of the module	Total student workload	Contact hours	Individual work hours
1	30	20	10

Aim of the module (course unit): competences foreseeen by the study programme		
The aim of study is giving to students the knowledges in artificial intelligence, neural networks and application of deep neural network for IoT.		
Learning outcomes of module (course unit)	**Teaching/learning methods**	**Assessment methods**
At the end of course, the successful student will be able	Interactive lectures, Learning in	Module Evaluation Questionnaire

to: 1. To differentiate the architectures of artificial neural network for the tasks of IoT systems	laboratories, Just-in-Time Teaching	
2. To design specialized architecture of deep neural network for different IoT solutions	Interactive lectures, Learning in laboratories, Just-in-Time Teaching	Module Evaluation Questionnaire
3. To provide correct learning and simulating of the artificial neural network for IoT task solutions	Interactive lectures, Learning in laboratories, Just-in-Time Teaching	Module Evaluation Questionnaire
4. To use Matlab and other toolboxes of deep neural network for implementation and simulating of IoT systems	Interactive lectures, Learning in laboratories, Just-in-Time Teaching	Module Evaluation Questionnaire

Themes	Contact **work hours**							**Time and tasks for individual work**	
	Lectures	Consultations	Seminars	Practiacl work	Laboratory work	Placements	**Total contact work**	**Individual work**	**Tasks**
1. IoT and IoE ecosystem 1.1 Data science for IoT vs traditional data science 1.2 The IoT ecosystem and IoT problems in data science (Overview: time series data, enterprise IoT edge computing, real--time processing, cognitive computing, image processing, introduction to deep	2				4		6	4	1.4 Extracting meaning from data 1.5 Handling large scale data, handling poor data quality

learning algorithms, geospatial analysis for IoT/managing massive geographic scale, strategies for integration with hardware, sensor fusion) 1.3 IoT datasets (IoT datasets by application: social networks, healthcare, smart cities, enterprises and manufacturing, energy etc. Data from IoT devices) **Lab:** Discovering of data processing technique for time series									
2. Scientific analytics models used in IoT verticals 2.1 Supervised algorithms, unsupervised algorithms (classification, regression, clustering, dimensionality reduction etc) applicable to IoT datasets 2.2. Data fusion and time series data processing from IoT devices **Lab:** Exploring of signal segmentation and using of window function	2				4		6	5	2.3 Applying predictive learning algorithms to IoT datasets 2.4 Enterprise IoT edge computing and Manufacturing: Predictive maintenance, anomaly detection, forecasting and missing event interpolation.
3 Data fusion and data processing from IoT device 3.1 Data Fusion Challenges 3.2 Mathematical methods of Data Fusion **Lab:** Anomaly detection in time series using clustering	2			2			4	5	3.3 ARIIMA: Real IoT implementation of a machine learning architecture for reducing energy consumption
3. Data mining models for IoT 3.1 Basic idea of using data mining for IoT 3.2 Classification for IoT 3.3 Clustering for IoT 3.4 Frequent Pattern Mining for IoT 3.5 Association analysis **Lab:** Exploratory data analysis and	2				4		6	4	4.6 Outlier detection 4.7 Spatial and temporal patterns mining for IoT

data visualization for IoT **Lab:** Classification and prediction modeling in IoT systems									
4. Mining of Massive Datasets 4.1 CRISP-DM data mining process methodology for IoT domain 4.2 Map reduce 4.3 Finding Similar Items **Lab:** Association analysis for frequent pattern mining in IoT systems	2				4		6	4	5.4 Similarity-preserving summaries of sets 5.5 Methods for High Degrees of Similarity
5. Stream mining 5.1 Stream Processing and Streaming Analytics: introduction and motivation 5.2 Real-Time Data-Stream Analysis 5.3 Data Streaming Models & Basic Mathematical Tools **Lab:** Analysis of interestingness measures in frequent pattern of IoT data	2				4		6	4	6.4 Kafka Streams 6.6 Streaming models (Landmark Streams, Sliding Window)
6.Basics of Machine Learning and Neural Networks 7.1 Introduction to machine learning and artificial intelligence 7.2 The model of neuron 7.3 The classification of artificial neural networks **Lab:** Recognition of alphanumeric information by using artificial neural network	2				4		6	4	7.4 Modern trends of Machine learning and Neural Networks
7.Deep Learning Neural Networks 8.1 The specifics of deep learning neural networks architecture 8.2 The training methods of deep learning neural network **Seminar:** Data preparation for deep neural network **Lab:** Image classification for IoT devices by using deep neural network	2		2		4		8	4	8.3 Specifics of deep neural network design and training
8.Deep Learning Neural Network Applications for IoT 9.1 Attack detection scheme using deep learning approach for IoT 9.2 Deep learning for the real-time embedded systems for IoT	2				4		6	2	9.4 Implementation of Deep Neural Networks for IoT applications

9.3 Pattern recognition for IoT **Lab:** Deep learning speech recognition for IoT									
9. Big data and NoSQL databases 9.1 A concept of Big data 9.2 NoSQL databases 10.3 Big data trade-offs between consistency, availability and latency **Lab:** Deploy an HDInsight Hadoop cluster on Linux	2				4		6	2	10.4 Database decisions for the Internet of Things
10. Big data modelling using Cassandra NoSQL data storage 11.1 The Cassandra NoSQL database 11..2 Cassandra consistency model 11..3 Cassandra data model 11..4 Basic rules of Cassandra data modeling 11.5 An example of Cassandra database design for IoT system **Lab:** Connect to the cluster via SSH **Lab:** Analyze an Apache log file with Hive	2				4		6	1	11.5 Big data analytics using Cassandra
11. Cassandra performance benchmarking 12.1 Experimental setup and benchmarking scenario 12.2 Raw data analysis and the cold start phenomenon 12.3 Read/write latency and throughput statistics 12.4 Theoretical regressions of the Cassandra performance **Lab:** Use MapReduce to analyze a text file with Python	2				4		6	2	12.5 Real-time data analysis with Cassandra
12. Methodology of optimal consistency setup 12.1 Finding the optimal consistency settings 13.2 Experimental-based methodology for optimal coordination of consistency settings **Lab:** Delete the Hadoop cluster	2				4		6	1	13.3 Core features of Microsoft Azure cloud platform
In total	26		2	2	48		78	42	

Assessment strategy	Weight in %	Deadlines	Assessment criteria
Lecture activity, including fulfilling special hometask	10	2,4	90–100% (A) – Outstanding work, showing a full grasp of all the questions answered. 85–89% (B) – Perfect or near perfect answers to a high proportion of the questions answered. There should be a thorough understanding and appreciation of the material. 75–84% (C) – A very good knowledge of much of the important material, possibly excellent in places, but with a limited account of some significant topics. 65–74% (D) – There should be a good grasp of several important topics, but with only a limited understanding or ability in places. There may be significant omissions. 60–64% (E) – Students will show some relevant knowledge of some of the issues involved, but with a good grasp of only a minority of the material. Some topics may be answered well, but others will be either omitted or incorrect. 35–59% (F) – There should be substantial deficiencies, or no answers, across large parts of the topics set, but with a little relevant and correct material in places 1–34% (FX) – Very little or nothing that is correct and relevant.
Learning in laboratories	30	3,4	90% – 100% An outstanding piece of work, superbly organised and presented, excellent achievement of the objectives, evidence of original thought. 85-89% (B) - Students will show a thorough understanding and appreciation of the material, producing

			work without significant error or omission. Objectives achieved well. Excellent organisation and presentation. 75-84% (C) -. Students will show a thorough understanding and appreciation of the material, producing work without significant error or omission. When performing laboratory work student completely provide results and make the right conclusions, but admits inaccuracies or errors. 65-74 (D) – The student can solve simple problems and has the ability to perform basic operations and basic transformation and some algorithmic, but is not able to independently formulate task and determine the solution. When performing laboratory work the student execute it with errors. He draws conclusions, but not sufficient to understand the purpose of work. 60-64% (E) – The student can solve simple problems by using teacher support to perform basic operations and basic transformation, but not able to formulate the problem by verbal description to determine the solution. 35-59% (F) – When performing practical (laboratory) work the student knows how to use computers, but is unable to complete the task. The work may contain some correct and relevant material, but most issues are neglected or are covered incorrectly. 1-34% (FX) – The student does not perform the tasks. Very little or nothing that is correct and relevant and no real appreciation of the laboratory work requirements.
Module Evaluation Quest	60	4	The score corresponds to the percentage of correct answers to the test questions

Author	Year of issue	Title	No of periodical or volume	Place of printing. Printing house or intrenet link
Compulsory literature				
Ajit Jaokar, Jean-Jacques Bernard	2016	Data Science for Internet of Things		http://www.opengardensblog.futuretext.com/archives/2016/07/a-methodology-for-solving-problems-with-datascience-for-internet-of-things.html
Qingchun Jiang, Sharma Chakravarthy	2009	Stream Data Processing: A Quality of Service Perspective: Modeling, Scheduling, Load Shedding, and Complex Event Processing	Vol. 36	Springer; Advances in Database Systems
Ian Goodfellow, Yoshua Bengio and Aaron Courville	2016	Deep Learning	book	An MIT Press book www.deeplearningbook.org
Ethem Alpaydin	2014	Introduction to Machine Learning, Third Edition	book	2014 Massachusetts Institute of Technology https://mitpress.mit.edu/books/introduction-machine-learning-0
Michael A. Nielsen	2015	Neural Networks and Deep Learning	online book	Determination Press: http://neuralnetwo

				rksanddeeplearning.com/index.html
Additional literature				
Vincent C. Müller (Ed.)	2016	Fundamental Issues of Artificial Intelligence	563 p.	Springer
Russell, S. and Norvig, P	2004	Artificial Intelligence-A Modern Approach	book	2nd Edition, Pearson Education / Prentice Hall of India
Ionut Danaila, Pascal Joly, Sidi Mahmoud Kaber	2006	An Introduction to Scientific Computing: Twelve Computational Projects Solved with MATLAB	book	Published November 27th 2006 by Springer
Stephen J. Chapman	2007	MATLAB Programming for Engineers	book	Published November 1st 2007 by Thomson Learning

АНОТАЦІЯ

УДК 004.415/.416.04(076.5)=111

Скарга-Бандурова І.С., Горбенко А.В., Білобородова Т.О., Коваль В.С., Саченко А.О., Тарасюк О.М. **Наука про дані для Інтернету Речей та Інтернету Всього** / За ред. І.С. Скарга-Бандурової та А.В. Горбенко – МОН України, Східноукраїнський університет ім. Володимира Даля, Національний аерокосмічний університет ім. М. Є. Жуковського «ХАІ». – 169 с.

Викладено матеріали практичної частини курсу “MC2. Data Science for IoT and IoE”, підготовленого в рамках проекту ERASMUS+ ALIOT “Internet of Things: Emerging Curriculum for Industry and Human Applications” (573818-EPP-1-2016-1-UK-EPPKA2-CBHE-JP).

Наведена структура робіт з перевірки знань з курсу, відповідний тренінговий матеріал, приклади виконання завдань та критерії оцінювання. В процесі навчання надаються теоретичні аспекти розроблення та впровадження IoT – систем. Наведені базові концепції та підходи науки про дані для IoT – систем.

Призначено для інженерів, розробників та науковців, які займаються розробкою та впровадженням IoT для промислових систем, для аспірантів університетів, які навчаються за напрямом комп’ютерних наук, комп’ютерної та програмної інженерії, а також для викладачів відповідних курсів.

Бібл. – 76, рисунків – 60, таблиць – 7.

ЗМІСТ

CONTENTS

Скарга-Бандурова Інна Сергіївна
Горбенко Анатолій Вікторович
Білобородова Тетяна Олександрівна
Коваль Василь Сергійович
Саченко Анатолій Олексійович
Тарасюк Ольга Михайлівна

Наука про дані для Інтернету Речей та Інтернету Всього

Практикум

(англійською мовою)

Редактори Скарга-Бандурова І.С., Горбенко А.В.

Комп'ютерна верстка: *В.С. Харченко, О.О. Іляшенко*

Зв. план, 2019
Підписаний до друку 22.08.2019
Формат 60х84 1/16. Папір офс. No2. Офс. друк.
Умов. друк. арк. 9,82. Уч.-вид. л. 10,56. Наклад 150 прим.
Замовлення 220819-3

Національний аерокосмічний університет ім. М. Є. Жуковського
"Харківський авіаційний інститут"
61070, Харків-70, вул. Чкалова, 17
http://www.khai.edu

Випускаючий редактор: ФОП Голембовська О.О.
03049, Київ, Повітрофлотський пр-кт, б. 3, к. 32.
Свідоцтво про внесення суб’єкта видавничої справи до державного реєстру видавців,
виготовлювачів і розповсюджувачів видавничої продукції
серія ДК No 5120 від 08.06.2016 р.

Видавець: ТОВ «Видавництво «Юстон»
01034, м. Київ, вул.. О. Гончара, 36-а, тел.: +38 044 360 22 66
www.yuston.com.ua
Свідоцтво про внесення суб’єкта видавничої справи до державного реєстру видавців,
виготовлювачів і розповсюджувачів видавничої продукції
серія ДК No 497 від 09.09.2015 р.