Table of Contents

**KHmelnYTSKYI NATIONAL UNIVERSITY**

**APPROVED BY**

|  |  |
| --- | --- |
| Dean of the Faculty of Information Technologies |  |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **Tetyana HOVORUSHCHENKO** |
| Signature | Name, SURNAME |

2025

### WORK PROGRAM OF THE ACADEMIC DISCIPLINE

Conducting Experiments and Processing the Results of Scientific Research

Discipline name

| **Purpose of the Work Program** | For the educational program “COMPUTER SCIENCE” of the specialty F7 Computer Science |
| --- | --- |
| **Level of Higher Education** | Second (Master’s) |
| **Language of Instruction** | English |
| **Discipline Volume, ECTS Credits** | 4 |
| **Discipline Status** | Elective |
| **Faculty** | Information Technologies |
| **Department** | Computer Science |

| **Form of Education** | **Discipline Volume** |  | **Number of Hours** |  |  |  |  |  | **Form of Semester Control** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **ECTS Credits** | **Hours** | **Classroom Sessions** |  |  |  |  | **Independent Study** |  |
|  |  |  | **Total** | **Lectures** | **Laboratory Work** | **Practical Sessions** | **Seminar Sessions** |  |  |
| Full-time | 4 | 120 | 64 | 32 | 32 |  |  | 56 | Credit |

The work program is based on the Master’s educational program “COMPUTER SCIENCE” and the higher education standard for the specialty F3 Computer Science.

|  |  |
| --- | --- |
| Work program compiled by \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **PhD Pavlo RADIUK** |
|  | Academic degree, academic title, Name, SURNAME |
| Signature |  |

Approved at the meeting of the Department of Computer Science

Protocol from **19.03** **2025** № **11**.

|  |  |
| --- | --- |
| Head of the Department of Computer Science | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **Oleksandr BARMAK** |
| Title | Signature            Name, SURNAME |

Khmelnytskyi 2025

### 1. Discipline Description

The academic course “Conducting Experiments and Processing the Results of Scientific Research” is an elective component of the “COMPUTER SCIENCE” educational program for students of the F7 Computer Science specialty. Taught in English at Khmelnytskyi National University, this course provides an in-depth, practical exploration of the complete machine learning experimental lifecycle using the PyTorch framework.

### 2. Aim, Subject, and Objectives of the Discipline

**Aim of the discipline.** To cultivate a comprehensive set of practical skills for conducting advanced scientific research in computer science. The course covers the entire experimental workflow, from conceptualization and PyTorch fundamentals to model construction, training, modular code design, replication of scientific literature, and deployment of machine learning solutions.

**Subject of the discipline.** The methodologies, tools, and best practices for executing machine learning experiments. This includes mastering the PyTorch framework, understanding standard experimental workflows, designing and implementing neural network architectures for classification and computer vision, managing custom datasets, applying principles of modular software engineering, replicating state-of-the-art research papers, and deploying trained models.

**Objectives of the discipline.**

* To master the fundamental concepts of PyTorch, including its tensor library and automatic differentiation engine.
* To learn and apply a standard, end-to-end PyTorch workflow, from data preprocessing to model evaluation.
* To acquire advanced skills in building, training, and validating deep learning models for complex classification and computer vision tasks.
* To gain proficiency in handling custom datasets, including data loading, preprocessing, and augmentation strategies.
* To master the principles of modular programming for creating scalable, reusable, and maintainable research code.
* To develop the ability to deconstruct and replicate model architectures and findings from scientific papers, with a focus on the Vision Transformer (ViT).
* To obtain hands-on experience in deploying machine learning models as interactive, user-facing web applications.

**Learning outcomes.** Upon successful completion of this course, a student will be able to: **know** the complete lifecycle of a machine learning experiment, the core components of the PyTorch ecosystem, the architectures of modern neural networks, advanced model evaluation and validation techniques, principles of modular software design, and fundamentals of model deployment; **be proficient** with the PyTorch framework, data visualization and deployment libraries (e.g., Matplotlib, Gradio), and version control systems for effective code management; **be able to** independently design, implement, train, and evaluate deep learning models for solving real-world problems, refactor experimental code from notebooks into modular scripts, critically analyze and replicate architectures from scientific literature, and deploy trained models as accessible web services.

### 3. Explanatory Note

The discipline “Conducting Experiments and Processing the Results of Scientific Research” is an elective component designed to deepen students’ practical skills in conducting scientific experiments with modern machine learning tools. This course logically complements the theoretical knowledge acquired from mandatory program components, such as “Models and Methods of Text Analytics” and “Large Computer Vision Models,” enabling students to apply these concepts to solve real-world problems. The course emphasizes the entire research cycle: from model development and training to validation, optimization, and deployment. Throughout this process, students cultivate essential soft skills, including the ability to independently analyze research papers, reproduce experimental results, manage code effectively, and present their work through interactive demonstrations.

**Aim of the discipline.** To master the complete cycle of conducting scientific experiments in computer science, from conceptualization to a fully deployed model, utilizing the PyTorch framework.

**Subject of the discipline.** The methodologies and tools for conducting experiments in machine learning, including the PyTorch framework, neural network architectures for classification and computer vision, data management, principles of modular programming, replication of scientific papers, and model deployment techniques.

**Tasks of the discipline.** To teach students to apply PyTorch for creating and training deep learning models; implement a complete experimental workflow; work with diverse data types, including custom datasets; structure code into modular Python scripts; analyze and reproduce architectures from scientific publications; and deploy trained models as interactive web applications to demonstrate their findings.

**Learning outcomes.** After completing the course, the student must: **know** the theoretical foundations and practical aspects of the full research cycle in machine learning, modern neural network architectures, and methods for their evaluation and validation; **be proficient** with PyTorch, Gradio, Hugging Face, and relevant data visualization libraries; **be able to** independently develop, train, and deploy deep learning models, replicate results from research papers, effectively structure projects, and present their research findings.

### 4. Structure of ECTS Credits of the Discipline

| **Topic Name** | **Lectures** | **Laboratory Work** | **Independent Study** |
| --- | --- | --- | --- |
| Topic 1. PyTorch Fundamentals and Workflow | 4 | 4 | 7 |
| Topic 2. Architectures for Neural Network Classification | 4 | 4 | 7 |
| Topic 3. End-to-End Computer Vision with torchvision | 4 | 4 | 7 |
| Topic 4. Advanced Data Handling with Custom Datasets | 4 | 4 | 7 |
| Topic 5. Modularizing Code for Reproducible Experiments | 4 | 4 | 7 |
| Topic 6. Paper Replicating: Implementing the Vision Transformer | 4 | 4 | 7 |
| Topic 7. Creating an Interactive Machine Learning Demo with Gradio | 4 | 4 | 7 |
| Topic 8. Machine Learning Model Deployment and Real-World Applications | 4 | 4 | 7 |
| **Total for the semester:** | **32** | **32** | **56** |

### 5. Program of the Academic Discipline

#### 5.1 Content of the Lecture Course

| **Lecture Number** | **List of Lecture Topics and Their Annotations** | **Number of Hours** |
| --- | --- | --- |
|  | **Topic 1. PyTorch Fundamentals and Workflow** | **4** |
| 1 | Introduction to PyTorch and Tensor Fundamentals. Core concepts of tensors, fundamental operations. | 2 |
| 2 | Advanced Tensor Operations and GPU Acceleration. Manipulating tensor shapes, running computations on different devices. | 2 |
|  | **Topic 2. Architectures for Neural Network Classification** | **4** |
| 3 | The Standard PyTorch Workflow. Data preparation, model building, defining loss functions and optimizers, the training loop. | 2 |
| 4 | Building a Complete Linear Regression Model. Implementing an end-to-end workflow for a regression task, including model saving and loading. | 2 |
|  | **Topic 3. End-to-End Computer Vision with torchvision** | **4** |
| 5 | Architecting Neural Networks for Binary Classification. Understanding non-linearity and activation functions for binary problems. | 2 |
| 6 | Scaling Up: Multi-Class Classification. Adapting architectures for multi-class problems and evaluating performance. | 2 |
|  | **Topic 4. Advanced Data Handling with Custom Datasets** | **4** |
| 7 | Introduction to Computer Vision with torchvision. Exploring torchvision.datasets and torchvision.transforms. | 2 |
| 8 | Building and Evaluating a Baseline Convolutional Neural Network (CNN). Creating a simple CNN and establishing a performance baseline. | 2 |
|  | **Topic 5. Modularizing Code for Reproducible Experiments** | **4** |
| 9 | Loading Data with ImageFolder and Custom Dataset Classes. Techniques for handling standard and non-standard data structures. | 2 |
| 10 | Data Augmentation Techniques for Improved Generalization. Using TrivialAugmentWide and other transforms to prevent overfitting. | 2 |
|  | **Topic 6. Paper Replicating: Implementing the Vision Transformer** | **4** |
| 11 | Principles of Modular Code. The rationale and techniques for converting notebooks into reusable Python scripts. | 2 |
| 12 | Creating Reusable Components. Building data\_setup.py, engine.py, and model\_builder.py for scalable projects. | 2 |
|  | **Topic 7. Creating an Interactive Machine Learning Demo with Gradio** | **4** |
| 13 | Deconstructing a Research Paper: The Vision Transformer (ViT). Analyzing the paper’s architecture, equations, and key contributions. | 2 |
| 14 | Implementing the Core Components of ViT. Coding the patch embedding, attention mechanism, and encoder blocks. | 2 |
|  | **Topic 8. Machine Learning Model Deployment and Real-World Applications** | **4** |
| 15 | Introduction to Model Deployment with Gradio. Understanding the input-function-output paradigm and creating simple interfaces. | 2 |
| 16 | Building and Deploying a Full-Stack ML App. Structuring a Gradio project and deploying it to Hugging Face Spaces. | 2 |
|  | **Total for the semester:** | **32** |

#### 5.2 Content of Laboratory Sessions

| **№** | **Topic of the Laboratory Session** | **Number of Hours** |
| --- | --- | --- |
| 1 | Tensor Creation, Indexing, and Manipulation in PyTorch.Source: Notebook 01. | 2 |
| 2 | Implementing GPU-Accelerated Tensor Computations.Source: Notebook 01. | 2 |
| 3 | Building a Complete End-to-End PyTorch Workflow for a Regression Task.Source: Notebook 02. | 2 |
| 4 | Experimenting with Different Optimizers and Learning Rate Scheduling.Source: Notebook 02. | 2 |
| 5 | Developing a Binary Classification Model for Non-Linear Data.Source: Notebook 03. | 2 |
| 6 | Implementing a Multi-Class Classification Model and Analyzing its Performance.Source: Notebook 03. | 2 |
| 7 | Training a Baseline Computer Vision Model on a Standard Dataset (e.g., FashionMNIST).Source: Notebook 04. | 2 |
| 8 | Comparing the Performance and Inference Speed of Different CNN Architectures.Source: Notebook 04. | 2 |
| 9 | Creating a Custom Dataset for a Unique Image Classification Task.Source: Notebook 05. | 2 |
| 10 | Applying TrivialAugmentWide and Other Advanced Data Augmentation Strategies.Source: Notebook 05. | 2 |
| 11 | Refactoring a Jupyter Notebook into a Set of Modular Python Scripts.Source: Notebook 06. | 2 |
| 12 | Constructing a Command-Line Trainable Script (train.py) Using Pre-built Modules.Source: Notebook 06. | 2 |
| 13 | Implementing the Patch Embedding and Transformer Encoder Blocks of a ViT.Source: Notebook 07. | 2 |
| 14 | Assembling and Verifying a Complete Vision Transformer Model.Source: Notebook 07. | 2 |
| 15 | Building an Interactive Web Demo for a Trained Model Using Gradio.Source: Notebook 08. | 2 |
| 16 | Deploying a Complete Machine Learning Application to Hugging Face Spaces.Source: Notebook 08. | 2 |
|  | **Total for the semester:** | **32** |

#### 5.3 Content of the Student’s Independent Study

The student’s independent work consists of systematically studying the course material from the provided Jupyter Notebooks, preparing for laboratory sessions, completing individual homework assignments (IHA), and preparing for the final assessment. Students have access to the department’s page in the Modular Learning Environment, which hosts the Work Program and all necessary educational materials.

| **Week Number** | **Type of Independent Work** | **Number of Hours** |
| --- | --- | --- |
| 1-4 | Review of lecture materials (Topics 1-2). Preparation for Lab Works №1-4. | 14 |
| 5-8 | Review of lecture materials (Topics 3-4). Preparation for Lab Works №5-8. Completion of IHA №1 “Comparative Analysis of Classification Models on a Custom Dataset”. | 14 |
| 9-12 | Review of lecture materials (Topics 5-6). Preparation for Lab Works №9-12. Completion of IHA №2 “Modularizing a Research Project: From Notebook to a Structured Python Application”. Completion of IHA №3 “Replicating and Extending a Research Paper’s Architecture (ViT)”. | 14 |
| 13-16 | Review of lecture materials (Topics 7-8). Preparation for Lab Works №13-16. Completion of IHA №4 “Developing and Deploying a Full-Stack FoodVision Application with Gradio and Hugging Face”. | 14 |
|  | **Total for the semester:** | **56** |

**Notes:** IHA – individual homework assignment.

### 6. Teaching Technologies and Methods

The learning process for this discipline is based on modern teaching technologies and methods, including: **lectures** (utilizing problem-based and interactive learning, live code demonstrations, and information-communication technologies); **laboratory sessions** (focusing on the practical implementation, training, and analysis of machine learning models); **independent study** (reviewing theoretical materials, preparing for laboratory work, and completing individual homework assignments), using Jupyter Notebooks, PyTorch, Gradio, and other state-of-the-art tools.

### 7. Methods of Control

Ongoing assessment is conducted during laboratory sessions and through the submission of individual assignments as scheduled in the work program. The primary methods of control include:

* Evaluation of the defense of laboratory work.
* Evaluation of the completion and defense of individual homework assignments.

The final semester grade is based on the cumulative results of this ongoing assessment.

### 8. Discipline Policy

The discipline’s policy is defined by the requirements for students as stipulated by the University’s current regulations. Attendance is mandatory. Successful completion requires active participation, timely submission and defense of all laboratory work and individual assignments. All forms of academic dishonesty, including plagiarism and cheating, are **strictly prohibited**. In case of violations, the student will receive an unsatisfactory grade and must redo the assignment.

### 9. Assessment of Student Learning Outcomes in the Semester

Student academic achievement is assessed on a 100-point scale. Each laboratory work and individual assignment is graded according to a predetermined number of points. To pass the course, a student must accumulate a minimum of 60 points.

**Structuring the Discipline by Types of Academic Work and Assessing Student Learning Outcomes**

| **Laboratory Work №:** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | **Individual Homework Assignment (IHA)** |  |  |  | **Credit** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **1** | **2** | **3** | **4** |  |
| **Points for each type of academic work (min-max)** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 1-2.5 | 1-2.5 | 1-2.5 | 1-2.5 | 1-2.5 | 1-2.5 | 1-2.5 | 1-2.5 | 1-2.5 | 1-2.5 | 1-2.5 | 1-2.5 | 1-2.5 | 1-2.5 | 1-2.5 | 1-2.5 | 9-15 | 9-15 | 9-15 | 9-15 | Based on rating |
| **16-40** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | **36-60** |  |  |  | **60-100** |

### 10. Questions for Self-Assessment of Learning Outcomes

1. The concept of a tensor in PyTorch and its fundamental operations.
2. The main stages of the standard machine learning workflow in PyTorch.
3. The roles of a loss function and an optimizer in model training.
4. The architecture of a basic neural network for classification tasks.
5. The definition of a Convolutional Neural Network (CNN) and its primary applications.
6. The function of nn.Conv2d and nn.MaxPool2d layers in a CNN.
7. The process of creating and utilizing a custom Dataset in PyTorch.
8. The concept of data augmentation and its importance in model training.
9. The rationale for converting code from Jupyter Notebooks to modular Python scripts.
10. The core components of the Vision Transformer (ViT) architecture.
11. The concept of “patch embedding” as used in the Vision Transformer.
12. The definition and purpose of the self-attention mechanism.
13. Different approaches to deploying machine learning models, such as on-device and cloud-based.
14. The functionality and use case of the Gradio library for creating ML demos.
15. The typical file structure for a Gradio application deployed on Hugging Face Spaces.
16. The process of splitting a dataset into training, validation, and testing sets.
17. The concept of a learning rate and its impact on model training.
18. The difference between binary, multi-class, and multi-label classification.
19. The purpose of non-linear activation functions like ReLU and Sigmoid.
20. The functionality of torchvision.transforms for image preprocessing.
21. The relationship between torch.utils.data.Dataset and torch.utils.data.DataLoader.
22. The definition of a “baseline model” and its role in machine learning experiments.
23. The performance-speed tradeoff in selecting a model for deployment.
24. The concept of transfer learning and how to implement it using a feature extractor.
25. The process of freezing layers in a pretrained model.
26. The steps involved in replicating a model architecture from a research paper.
27. The role of the forward() method in a PyTorch nn.Module.
28. The concept of a residual or skip connection in neural networks.
29. How to use torchinfo.summary() to inspect a model’s architecture.
30. The distinction between model logits, prediction probabilities, and prediction labels.
31. The function of the softmax activation in multi-class classification.
32. The process of creating a requirements.txt file for a Python application.
33. The concept of “production code” in the context of machine learning.
34. The pros and cons of using notebooks versus Python scripts for ML projects.
35. The purpose of setting a random seed for reproducibility in experiments.
36. The definition of hyperparameters and examples from the ViT paper (e.g., MLP size, number of heads).
37. The difference between a “layer” and a “block” in a neural network architecture.
38. The role of the learnable [class] token in the Vision Transformer.
39. The purpose of position embeddings in the Vision Transformer.
40. How a convolutional layer can be used to create image patches.
41. The process of flattening a tensor and its application in neural networks.
42. The definition of “underfitting” and “overfitting” and how to identify them using loss curves.
43. Techniques for mitigating overfitting, such as data augmentation and dropout.
44. Techniques for mitigating underfitting, such as increasing model complexity.
45. The function of model.train() and model.eval() modes in PyTorch.
46. The purpose of the torch.inference\_mode() context manager.
47. How to save and load a trained model’s state\_dict in PyTorch.
48. The definition of an API and its role in model deployment.
49. The benefits of deploying a model as an interactive web application.
50. The workflow for deploying a Gradio application from a local machine to Hugging Face Spaces.

### 11. Educational and Methodological Support

The educational process for this discipline is fully supported by a comprehensive suite of educational materials in the form of Jupyter Notebooks. These materials provide theoretical explanations, practical code examples, and hands-on assignments.

### 12. Material, Technical, and Software Support

Laboratory sessions are conducted in the department’s computer labs, which are equipped with modern computer hardware. The required software includes operating systems (Windows, Linux), a Jupyter Notebook environment (or Google Colab), Python 3.x, and essential libraries such as PyTorch, torchvision, torchinfo, pandas, matplotlib, and Gradio.

### 13. Recommended Literature

* 1. PyTorch Fundamentals (Jupyter Notebook)
  2. PyTorch Workflow Fundamentals (Jupyter Notebook)
  3. PyTorch Neural Network Classification (Jupyter Notebook)
  4. PyTorch Computer Vision (Jupyter Notebook)
  5. PyTorch Custom Datasets (Jupyter Notebook)
  6. PyTorch Going Modular (Markdown document)
  7. PyTorch Paper Replicating (Jupyter Notebook)
  8. PyTorch Model Deployment (Jupyter Notebook)

1. PyTorch Documentation. URL: https://pytorch.org/docs/stable/index.html
2. Dosovitskiy, A., et al. (2020). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *arXiv preprint arXiv:2010.11929*. URL: https://arxiv.org/abs/2010.11929

### 14. Information Resources

1. Modular Learning Environment of Khmelnytskyi National University.
2. Electronic Library of Khmelnytskyi National University. URL: http://library.khmnu.edu.ua/
3. Institutional Repository of Khmelnytskyi National University. URL: https://elar.khmnu.edu.ua/home

### ANNOTATION OF THE DISCIPLINE (FOR THE CATALOG)

**CONDUCTING EXPERIMENTS AND PROCESSING THE RESULTS OF SCIENTIFIC RESEARCH**

|  |  |
| --- | --- |
| **Discipline Type** | Elective |
| **Level of Higher Education** | Second (Master’s) |
| **Language of Instruction** | English |
| **ECTS Credits** | 4.0 |
| **Forms of Education** | Full-time (day) |

**Learning outcomes.** After completing the course, the student must: **know** the theoretical foundations and practical aspects of the full research cycle in machine learning, modern neural network architectures, and methods for their evaluation and validation; **be proficient** with PyTorch, Gradio, Hugging Face, and relevant data visualization libraries; **be able to** independently develop, train, and deploy deep learning models, replicate results from research papers, effectively structure projects, and present their research findings.

**Content of the academic discipline.** Fundamentals of PyTorch, including tensors, operations, and automatic differentiation. The standard machine learning workflow: data preparation, model building, training, and evaluation. Architectures of neural networks for classification and computer vision, including Convolutional Neural Networks (CNNs). Working with custom datasets and data augmentation. Modularization of code and converting Jupyter Notebooks into Python scripts. Analysis and replication of architectures from scientific papers, focusing on the Vision Transformer (ViT). Deployment of models as interactive web applications using Gradio and Hugging Face Spaces.

**Planned learning activities.** The minimum volume of study for one ECTS credit of the academic discipline for the second (Master’s) level of higher education in the full-time form of education is 10 hours per 1 ECTS credit.

**Forms (methods) of teaching:** lectures (using problem-based and interactive learning, and live code demonstrations); laboratory sessions (focusing on the practical implementation, training, and analysis of models); independent study (reviewing theoretical materials and completing individual homework assignments) using Jupyter Notebooks and modern deep learning tools.

**Forms of assessment of learning outcomes:** assessment of the completion and defense of laboratory work; assessment of the completion and defense of individual homework assignments.

**Type of semester control:** credit.

**Educational resources:**

1. A series of educational materials in the Jupyter Notebook format.
2. Official PyTorch documentation.
3. Scientific articles from leading conferences and archives (e.g., arXiv).
4. Modular Learning Environment of Khmelnytskyi National University.
5. Electronic Library of Khmelnytskyi National University.

**Instructors:** PhD, Senior Lecturer Pavlo M. Radiuk