ДОДАТОК А

Використані протоколи

Протокол для опису типу проекту

code of project type

1st digit - is code of task

1 - classification, 2 - clasterisation, 3 - regression

2nd digit - is a code for project input data type

0 - audio, 1 - text, 2 - image, 3 - video, 4 - table, 5 - timesteps

3rd digit - is a code for type of project

0 - custom, 1 - applied, 2 - optimising

4th digit - type of optimising algorithm

0 - without, 1 - genetic, 2 - immune

5-6th digit - type of aplied models project

00 - without, 01 - ResNet50V2 .....

7th digit - is additional

-if not optimising && image, 0 - simple convolutional, 1 - perceptron, ...

```
For example 1221000 is a code of project which use c2d_gui window:
task:                   (1)classification
input type:             (2)image
project type:           (2)optimising
optimising algorithm:   (1)genetic
applied:                (0)
                        (0)without
additional:             (0)simple convolutional
```

Лістинг – Протокол мережевої взаємодії

```
def send_remaster(target: str, data: Any, socket):
    data = {"target": target, "data": data}
    data = json.dumps(data)
    data += "&"
    socket.send(bytes(data, encoding="utf-8"))
```

Протокол визначення набору елементів для відображення як список проектів

"name" - name of project: QLabel

"type" - type of project: QLabel

"play_pause" - play/pause button: QToolButton

"archive" - archive button: QToolButton

"sett" - settings of project button: QToolButton

"stop" - stop button: QToolButton

"progbar" - progress bar: QProgressBar

"toQueue" - toQueue button: QToolButton

"plot" - show plots button: QToolButton

...

ДОДАТОК Б

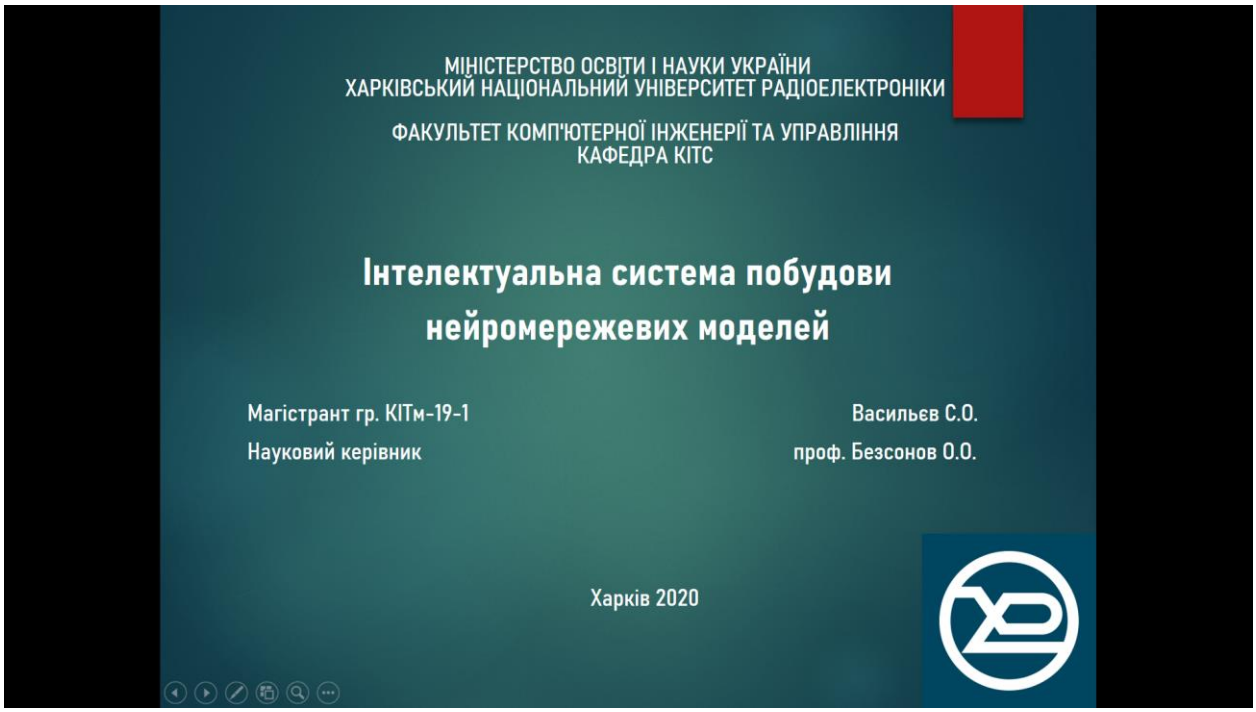
Реалізація основного циклу генетичного пошуку

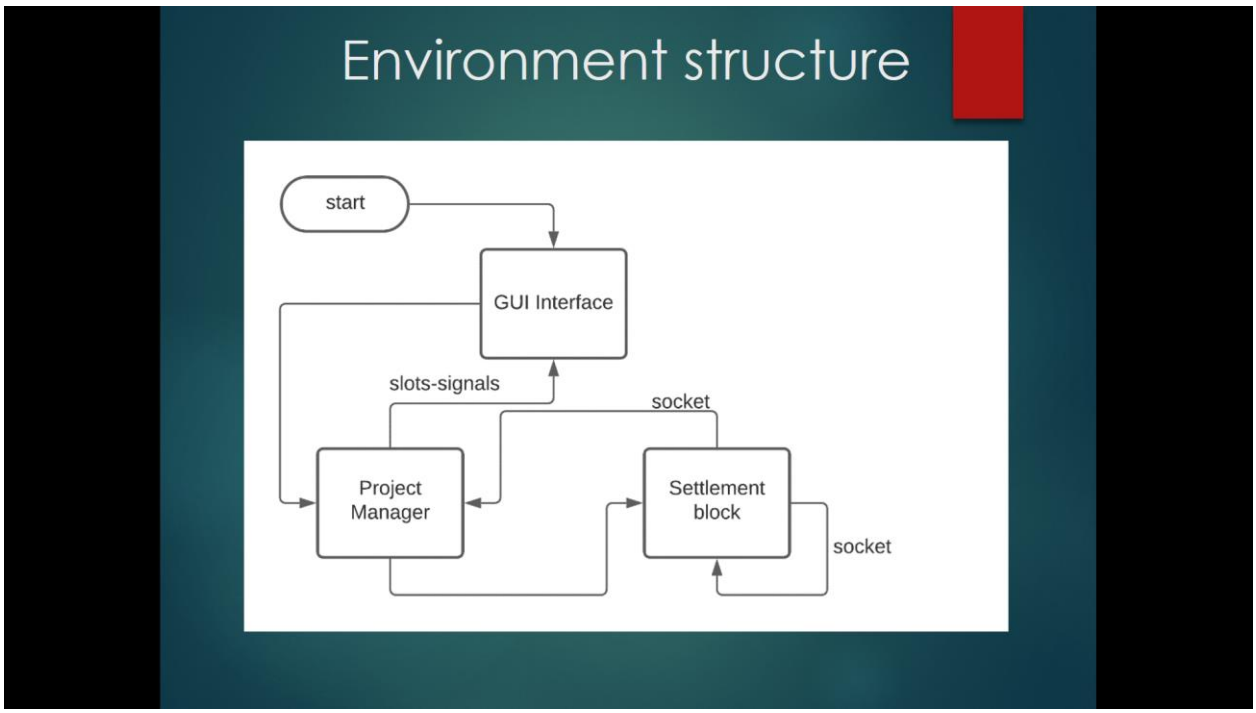
```
def c2dEvolve(self):
    self.population.clear()
    ########## Initialise population ##########
    send_remaster("gen_epoch", 0, self.pc_sock)
    for i in range(self.project_params.popSize):
        self.population.append(C2dChromosome(self.project_params))
        self.population[i].name = str(i + 1)
        print(self.population[-1].getNetConfig(0))
        send_remaster("chr_config", self.population[-1].getNetConfig(0),
                      self.pc_sock)
        self.calculate_c2d(self.population[-1], self.project_params,
                           self.net_port, self.opt_sock)
        data = json.loads(self.listener.buffer)
        self.listener.buffer = ""
        paramsCount = data.get("data")[0]
        report = data.get("data")[1]
        self.tempMetrics.append([paramsCount, report])
        self.population[i].paramsCount = paramsCount
        self.population[i].report = report
        send_remaster("interim_est",
                      [paramsCount, report.get("accuracy")], self.pc_sock)

    self.setAssessment(1, self.tempMetrics)
    self.population.sort(key=lambda x: x.assessment, reverse=True)
    send_remaster("accuracy", self.getAccuracy(0), self.pc_sock)
    send_remaster("assesment", self.getAssessment(1, 0), self.pc_sock)
    send_remaster("params", self.getParams(0), self.pc_sock)
    send_remaster("assesment_str", self.getAssessment(0, 0), self.pc_sock)

    ##########################################

    ########## Main cycle with genetic operators ##########
    select = selection(len(self.population), self.project_params.selection)
    for epoch in range(self.project_params.genEpoch):
        self.tempMetrics.clear()
        send_remaster("gen_epoch", epoch + 1, self.pc_sock)
        for i in range(len(self.population)):
```

```
            is_cross = False
            is_mutate = False
            if i < select[0]:
                self.tempMetrics.append([self.population[i].paramsCount,
                                         self.population[i].report])
                continue
            elif i < select[0] + select[1]:
                is_cross = self.population[i].mutate(self.project_params.mutateRate)
            else:
                is_mutate = self.population[i].mutate(self.project_params.mutateRate)
            if is_cross or is_mutate:
                send_remaster("chr_config", self.population[i].getNetConfig(0),
                              self.pc_sock)
                self.calculate_c2d(self.population[i], self.project_params,
                                   self.net_port, self.opt_sock)
                data = json.loads(self.listener.buffer)
                self.listener.buffer = ""
                paramsCount = data.get("data")[0]
                report = data.get("data")[1]
                send_remaster("interim_est",
                              [paramsCount, report.get("accuracy")], self.pc_sock)
                self.population[i].paramsCount = paramsCount
                self.population[i].report = report
                self.tempMetrics.append([paramsCount, report])

            else:
                self.tempMetrics.append([self.population[i].paramsCount,
                                         self.population[i].report])

        self.setAssessment(1, self.tempMetrics)
        self.population.sort(key=lambda x: x.assessment, reverse=True)
        send_remaster("accuracy", self.getAccuracy(epoch + 1), self.pc_sock)
        send_remaster("assesment", self.getAssessment(1, epoch + 1), self.pc_sock)
        send_remaster("params", self.getParams(epoch + 1), self.pc_sock)
        send_remaster("assesment_str", self.getAssessment(0, epoch+1), self.pc_sock)

    self.project_controller.is_run = False
    self.pc_sock.close()
```

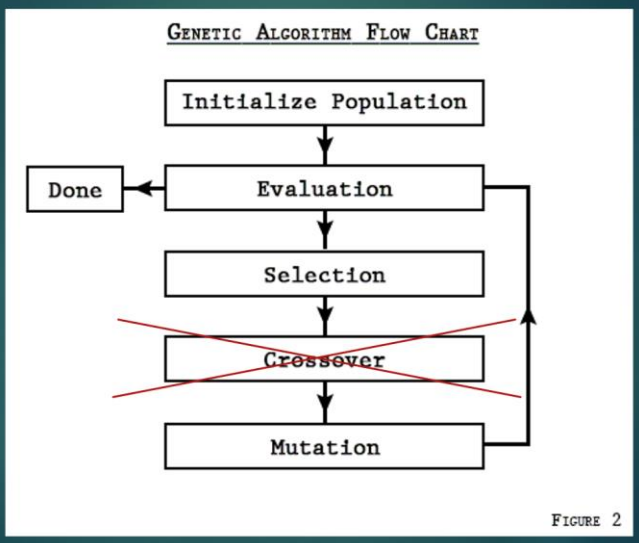
ДОДАТОК В

Перелік графічних матеріалів





Genetic algorithm



Hyperparameters

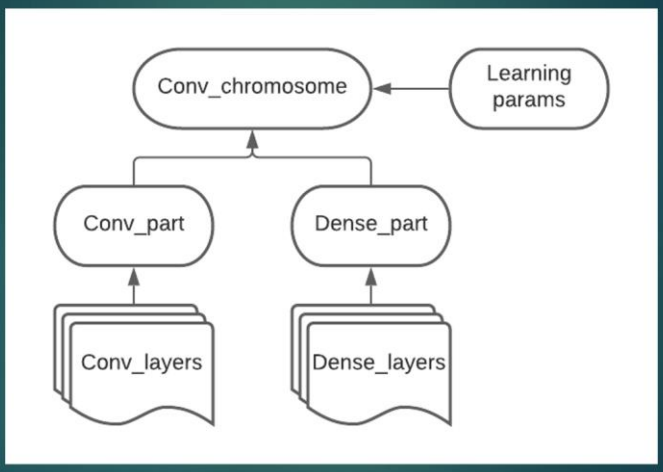
Structure

- Numb and type of layers
- Activation function
- Layer params

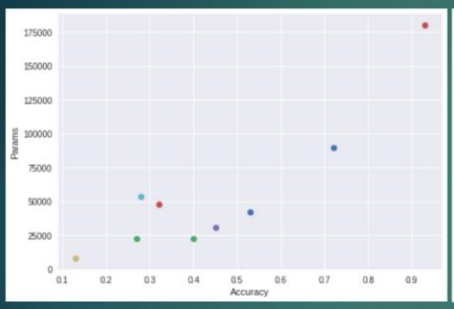
Fitting

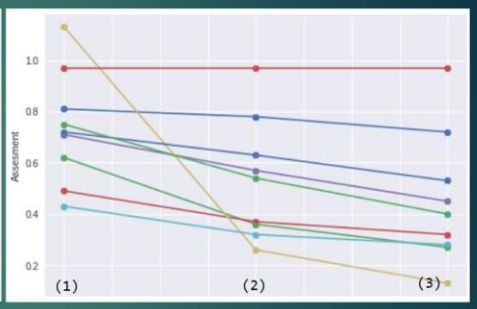
- Optimizer
- Loss function
- Init LR
- Epoch

Structure of chromosome



Fitness Function



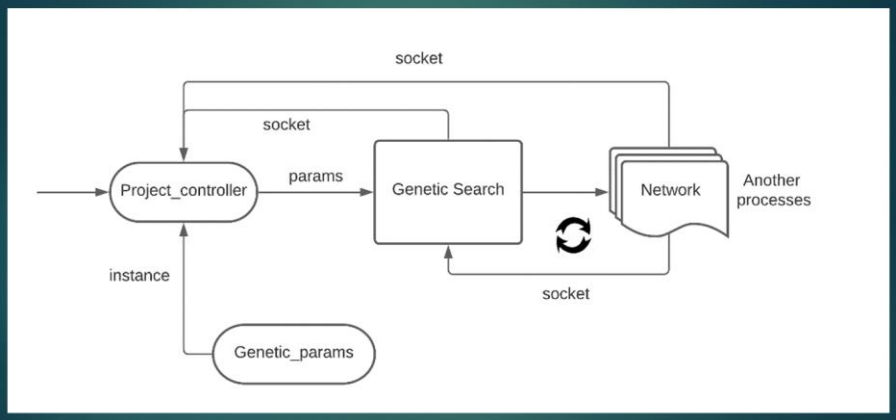


$$A_i = \alpha_i + \frac{p_{\min}}{p_i} \quad (1)$$

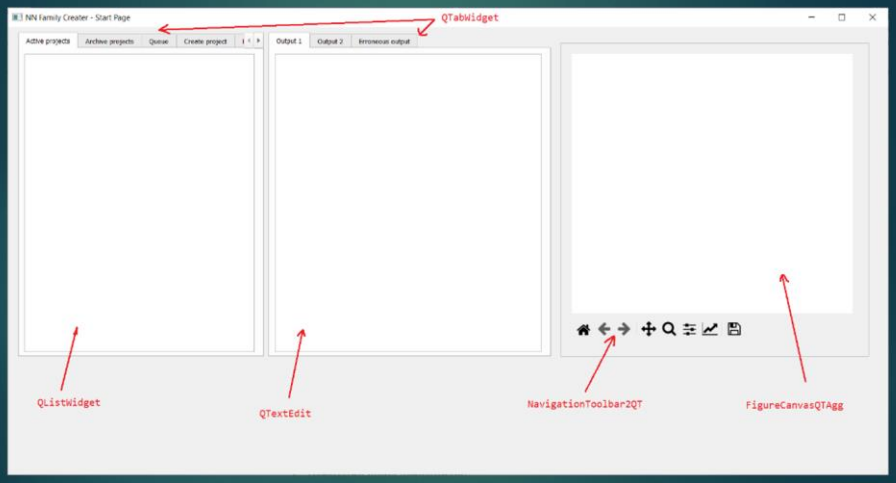
$$A_i = \alpha_i + \alpha_i \cdot \frac{p_{\min}}{p_i} \quad (2)$$

$$\begin{cases} A_i = \alpha_i + \alpha_i \cdot \frac{p_{\min}}{p_i}, \text{if } \alpha_i > t \\ A_i = \alpha_i, \text{if } \alpha_i < t \end{cases} \quad (3)$$

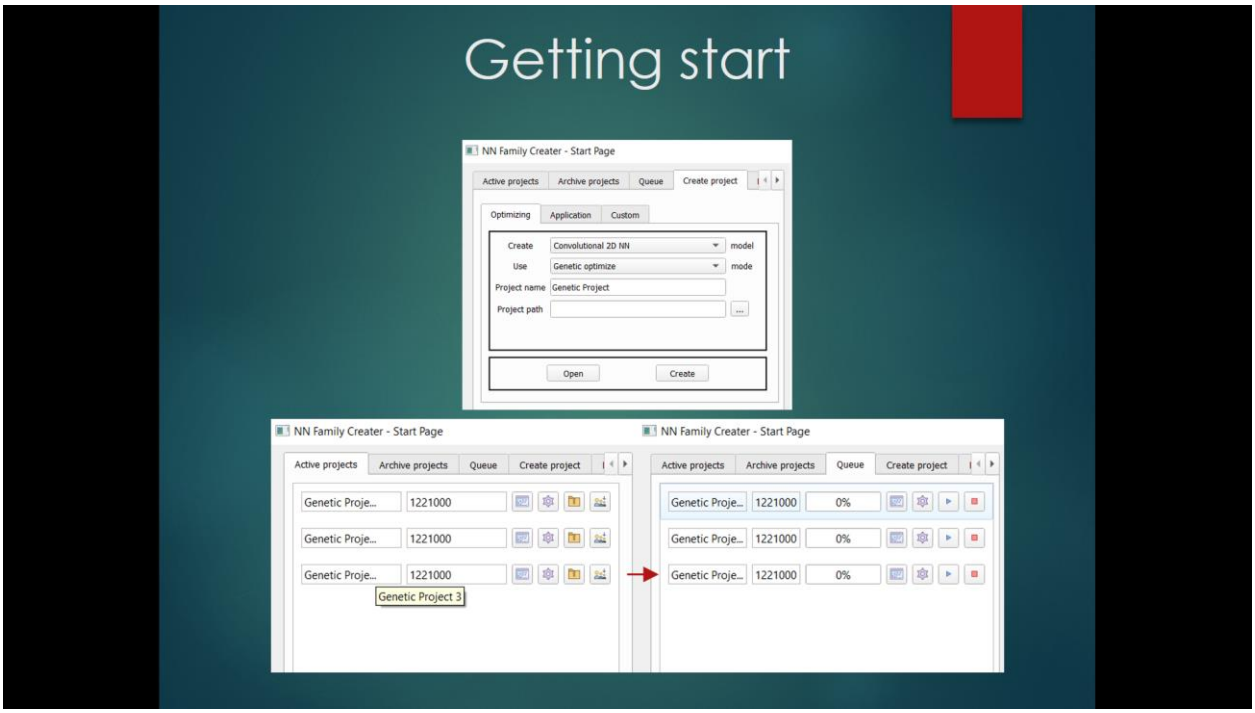
Genetic fitting



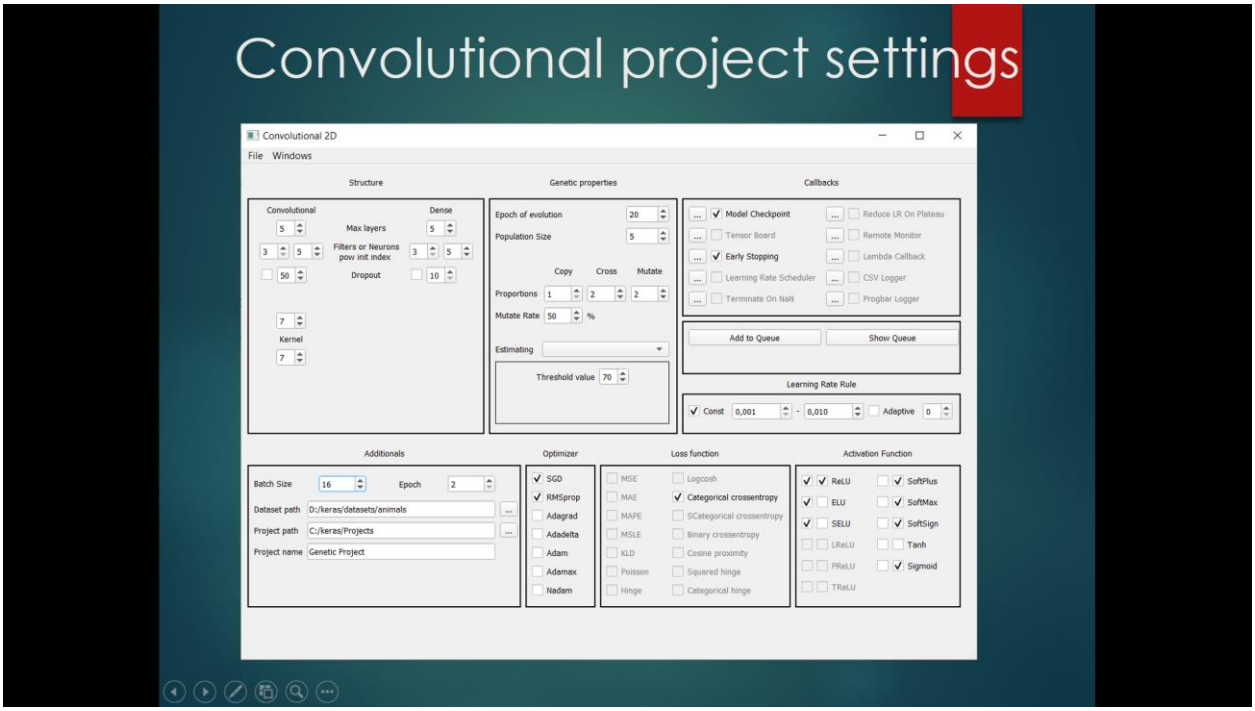
Start page



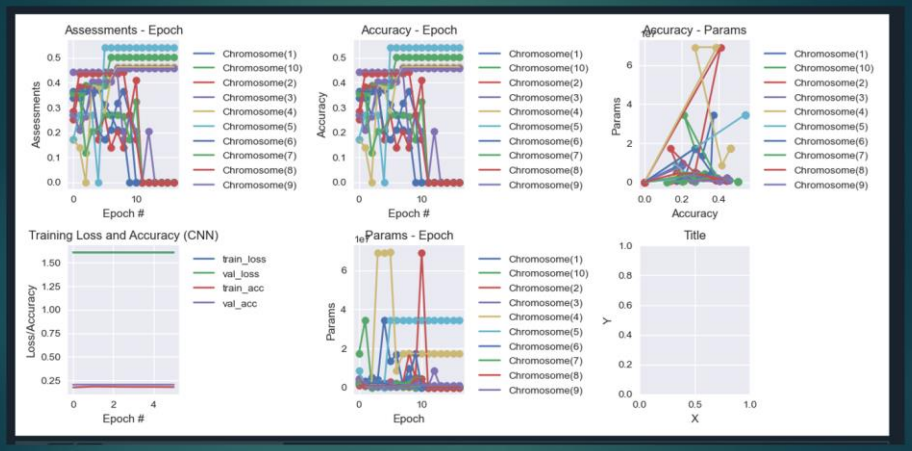
Getting start



Convolutional project settings



Different Plots



№	Позначення	Найменування	Дод. відомості
		Текстові документи	
1	ГЮІК.ХХХХХХ.ХХХ ПЗ	Пояснювальна записка	80 с.
		Графічні матеріали	
2		Слайд-презентація	14 слайдів
3	ГЮІК.ХХХХХХ.ХХХ Е1	Схема електрична структурна	1 арк. ф.А1
		пристрою шифрування	
4	ГЮІК.ХХХХХХ.ХХХ Е3	Схема електрична принципова	1 арк. ф.А1
		пристрою шифрування	
5	ГЮІК.ХХХХХХ.ХХХ Д1	Схема алгоритму шифрування	1 арк. ф.А1
		Носії інформації	
6		Компакт-диск (CD)	1 шт.

Змін.	Арк.	Номер докум.	Підп.	Дата	ГЮІК.ХХХХХХ.ХХХ ВД		
Розроб.					Проектування комп'ютерної мережі малого підприємства. Засоби захисту інформації	Літ	Аркуш 1 / Аркушів 1
Перевір.						У	
						ХНУРЕ Кафедра КІТС	
Н.контр.							
Затв.		Руденко О.Г.					