

ООО «ЛинкТелеком»

**ПРОГРАММА ДЛЯ ЭВМ
«ПРОГРАММНЫЙ МОДУЛЬ ВЗАИМОДЕЙСТВИЯ С
ПРЕДОБУЧЕННЫМИ МОДЕЛЯМИ НЕЙРОННЫХ СЕТЕЙ
И ИНТЕГРАЦИИ С ВНЕШНИМИ СИСТЕМАМИ»**

Инструкция по установке

Листов: 11

Ярославль, 2026

СОДЕРЖАНИЕ

1. БАЗОВОЕ СИСТЕМНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ.....	2
2. ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ	2
3. УСТАНОВКА ПО.....	3
3.1. Подготовка окружения.....	3
3.2. Процедура установки ПО	4
3.3. Запуск приложений	4
3.4. Получение дистрибутива	4
4. ОПИСАНИЕ ПО	5
4.1. Описание ПО.....	5
4.2. Функциональные возможности.....	5
4.3. Базовые требования к автоматизированному рабочему месту Пользователя системы.....	6
5. ПРОВЕРКА РАБОТОСПОСОБНОСТИ ПО	6
5.1. Проверка состояния ПО	7
6. ВРЕМЯ РАБОТЫ И КОНТАКТЫ ТЕХНИЧЕСКОЙ ПОДДЕРЖКИ	11

1. БАЗОВОЕ СИСТЕМНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

Системные программные средства, для которых обеспечивается эффективная работа программного обеспечения «Программный модуль взаимодействия с предобученными моделями нейронных сетей и интеграции с внешними системами» (далее по тексту ПО):

- операционная система для устройств: Linux (Ubuntu 22.04) или Debian.

2. ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ

ПО устанавливается на устройство.

Технические характеристики устройства:

- Количество ядер процессора: не менее 16;
- Частота процессора: не менее 2500 МГц;
- Объем оперативной памяти: не менее 16 ГБ;
- Общий объем накопителей HDD: не менее 512 ГБ;
- Общий объем накопителей SSD: не менее 256 ГБ.
- Объем видеопамати (при наличии): не менее 8 ГБ

- Сеть: не менее 100 Мбит/с
- ОС: Linux (Ubuntu 22.04) или Debian

3. УСТАНОВКА ПО

3.1. Подготовка окружения

После установки ОС необходимо выполнить:

- `sudo apt-get update`
- `sudo apt-get upgrade`

Далее необходимо установить Docker Engine и плагин Docker Compose (по официальной инструкции Docker):

```
for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-  
docker containerd runc; do sudo apt-get remove $pkg; done  
  
# Add Docker's official GPG key:  
  
sudo apt-get update  
  
sudo apt-get install ca-certificates curl  
  
sudo install -m 0755 -d /etc/apt/keyrings  
  
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o  
/etc/apt/keyrings/docker.asc  
  
sudo chmod a+r /etc/apt/keyrings/docker.asc  
  
  
# Add the repository to Apt sources:
```

```

echo \

"deb [arch=$(dpkg --print-architecture) signed-
by=/etc/apt/keyrings/docker.asc]
https://download.docker.com/linux/ubuntu \

$(. /etc/os-release && echo "${UBUNTU_CODENAME:-$VERSION_CODENAME}")
stable" | \

sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update

sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-
plugin docker-compose-plugin

```

3.2. Процедура установки ПО

Необходимо получить архив дистрибутива ПО (.tar), создать каталог vap в директории /opt и распаковать в него архив:

- cd /opt
- sudo mkdir -p vap
- sudo tar -xf archive.tar -C /opt/vap

3.3. Запуск приложений

Произвести запуск ПО:

- cd /opt/vap/
- make up

Проверьте статус запуска: `docker ps -a`

Выключить ПО:

- cd /opt/vap/
- make down

3.4. Получение дистрибутива

Дистрибутив ПО предоставляется правообладателем по запросу на адрес электронной почты info@linktelekom.ru.

4. ОПИСАНИЕ ПО

4.1. Описание ПО

Программное обеспечение «Программный модуль взаимодействия с предобученными моделями нейронных сетей и интеграции с внешними системами» (далее по тексту ПО) предназначено для организации взаимодействия между источниками данных, предобученными моделями искусственных нейронных сетей и внешними информационными системами. ПО обеспечивает сквозную обработку данных: прием информации из множественных источников, ретрансляцию, кодирование и транскодирование, передачу структурированных данных предобученным нейросетевым моделям для анализа, получение результатов их работы, сохранение результатов в базе данных, передачу результатов во внешние системы через программные интерфейсы (API) и их визуализацию в интерфейсе пользователя.

4.2. Функциональные возможности

- Приём и передача данных из разнородных источников.
- Анализ и преобразование поступающей информации.
- Автоматизация ключевых операций в бизнес-процессах.
- Обеспечение совместимости и синхронизации между компонентами системы.
- Упрощение работы за счёт интуитивного интерфейса.
- Гибкое расширение функционала под растущие нагрузки.
- Техническое сопровождение и обслуживание.
- Идентификация и категоризация объектов в системе.
- Мониторинг перемещения динамических объектов.
- Выявление отклонений и нестандартных ситуаций.
- Подключение к сторонним сервисам и базам данных.
- Повышение производительности и оптимизация ресурсов.
- Контроль доступа и шифрование данных для защиты информации.

Тип ЭВМ: сервер.

Технические характеристики устройства:

Количество ядер процессора: не менее 16

Частота процессора: не менее 2500 МГц

Объем оперативной памяти: не менее 16 ГБ

Объем видеопамяти (при наличии): не менее 8 ГБ

Общий объем накопителей HDD: не менее 512 ГБ

Общий объем накопителей SSD: не менее 256 ГБ

Сеть: не менее 100 Мбит/с

ОС: Linux (Ubuntu 22.04) или Debian

4.3. Базовые требования к автоматизированному рабочему месту Пользователя системы

Аппаратные требования:

ПК на базе процессора Intel Core i3 (или более производительный);

оперативная память: не менее 2 GB (рекомендуется 4 GB и более);

интернет-соединение: не менее 100 мбит/с (рекомендуется 1 гбит/с);

память видеоадаптера: не менее 256 MB (рекомендуется 512 MB и более).

Программные требования:

ОС Windows 7/8/10, Linux, MAC OS.

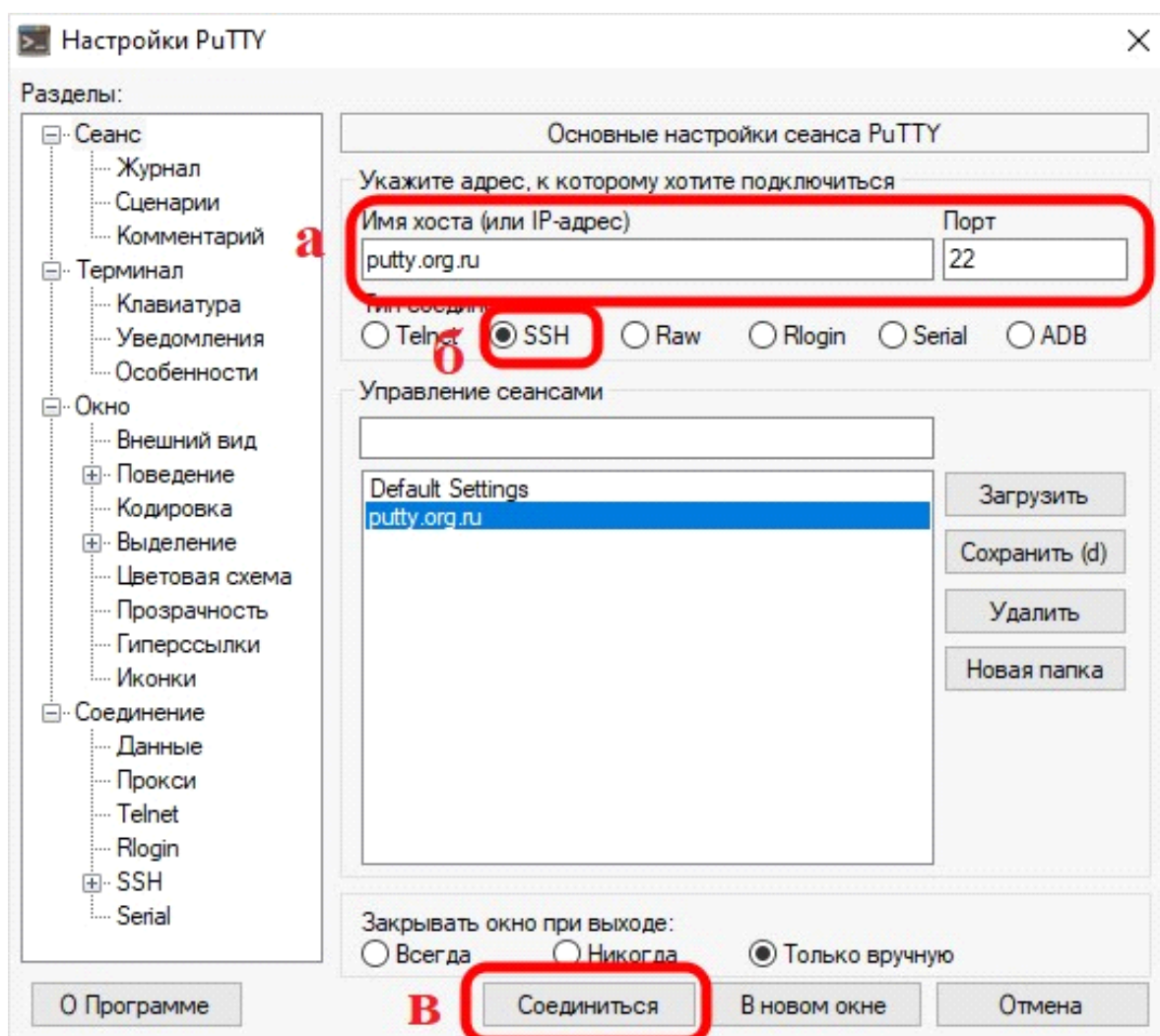
Браузер Google Chrome версии не ниже 50 (или другой с аналогичными характеристиками).

5. ПРОВЕРКА РАБОТОСПОСОБНОСТИ ПО

5.1. Проверка состояния ПО

Для того, чтобы открыть и проанализировать файлы логирования всех служб, участвующих в процессе детектирования объектов, необходимо провести ряд следующих действий:

- Установить программу PuTTY по ссылке: <https://putty.org.ru/download>
- Запустить программу;
- Ввести в поле «Имя хоста» IP-адрес сервера, порт 22;
- Установить тип соединения SSH и нажать кнопку **Соединиться**.
-



В появившемся окне ввести логин и пароль пользователя сервера.

Для выхода из любого лога достаточно нажать сочетание клавиш **ctrl + c**.

Для дальнейшей проверки работы служб необходимо открыть **консольное окно**.

Команды, которые необходимо ввести для проверки той или иной службы, представлены ниже.

- Команда `docker logs --tail 100 deploy-cutlet-1` покажет работу сервиса ИИ-аналитики (запуск моделей нейронных сетей и gRPC-сервера):

```
=====
== CUDA ==
=====

CUDA Version 12.9.1

Container image Copyright (c) 2016-2023, NVIDIA CORPORATION & AFFILIATES. All rights reserved.

This container image and its contents are governed by the NVIDIA Deep Learning Container License.
By pulling and using the container, you accept the terms and conditions of this license:
https://developer.nvidia.com/ngc/nvidia-deep-learning-container-license

A copy of this license is made available in this container at /NGC-DL-CONTAINER-LICENSE for your convenience.

2026/10/16 07:46:19 started grpc server on 0.0.0.0:9093
2026/10/16 07:46:27 Worker (port - 35008, model - License-plate, pid - 45) started
2026/10/16 07:46:30 Worker (port - 35006, model - graffiti-old, pid - 42) started
2026/10/16 07:46:30 Worker (port - 35005, model - Stickers, pid - 52) started
2026/10/16 07:46:30 Worker (port - 35001, model - Smoking, pid - 54) started
2026/10/16 07:46:30 Worker (port - 35000, model - Playground-Equipment-Damage, pid - 49) started
2026/10/16 07:46:30 Worker (port - 35007, model - Car-Truck_1243, pid - 40) started
2026/10/16 07:46:30 Worker (port - 35004, model - Playground-pavement-damage, pid - 43) started
2026/10/16 07:46:30 Worker (port - 35002, model - Flooding, pid - 46) started
2026/10/16 07:46:32 Worker (port - 35003, model - Person-Dog_1938, pid - 47) started
```

Сообщения об успешной инициализации процессов.

- Команда `docker logs --tail 100 -f deploy-notification_service-1` покажет процесс работы сервиса Веб-сокеты уведомлений системы

```
2026-10-16 07:46:14,052 - app.views - INFO - Подключение к RabbitMQ очередь: incident_status
2026-10-16 07:46:14,053 - app.views - INFO - Подключение к RabbitMQ очередь: realtime_events
2026-10-16 07:46:14,053 - app.views - INFO - Подключение к RabbitMQ очередь: new_confirmed_detects
2026-10-16 07:46:14,059 - app.views - INFO - Соединение установлено: realtime_events
2026-10-16 07:46:14,060 - app.views - INFO - Соединение установлено: incident_status
2026-10-16 07:46:14,060 - app.views - INFO - Соединение установлено: new_confirmed_detects
```

Подключение сервиса к очередям брокера сообщений.

- Команда `docker logs --tail 100 -f deploy-web-1` покажет процесс работы службы API работы с пользователями, управление платформы и получение данных аналитик

```
Полигоны успешно записаны в Redis
[2026-10-16 07:46:12 +0000] [56] [INFO] Starting gunicorn 23.0.0
[2026-10-16 07:46:12 +0000] [56] [INFO] Listening at: http://0.0.0.0:8000 (56)
[2026-10-16 07:46:12 +0000] [56] [INFO] Using worker: sync
[2026-10-16 07:46:12 +0000] [57] [INFO] Booting worker with pid: 57
[2026-10-16 07:46:12 +0000] [58] [INFO] Booting worker with pid: 58
[2026-10-16 07:46:12 +0000] [59] [INFO] Booting worker with pid: 59
[2026-10-16 07:46:13 +0000] [60] [INFO] Booting worker with pid: 60
[2026-10-16 07:46:13 +0000] [61] [INFO] Booting worker with pid: 61
```

Запуск веб-сервера и рабочих процессов API.

- Команда `docker logs --tail 100 apps-frontend-1` покажет работу веб-сервера (Nginx), обслуживающего интерфейс пользователя:

```
2026/10/16 07:46:04 [notice] 1#1: using the "epoll" event method
2026/10/16 07:46:04 [notice] 1#1: nginx/1.31.4
2026/10/16 07:46:04 [notice] 1#1: built by gcc 14.2.0 (Debian 14.2.0-19)
2026/10/16 07:46:04 [notice] 1#1: OS: Linux 6.8.0-138-generic
2026/10/16 07:46:04 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1024:524288
2026/10/16 07:46:04 [notice] 1#1: start worker processes
2026/10/16 07:46:04 [notice] 1#1: start worker process 28
2026/10/16 07:46:04 [notice] 1#1: start worker process 29
2026/10/16 07:46:04 [notice] 1#1: start worker process 30
2026/10/16 07:46:04 [notice] 1#1: start worker process 31
2026/10/16 07:46:04 [notice] 1#1: start worker process 32
2026/10/16 07:46:04 [notice] 1#1: start worker process 33
2026/10/16 07:46:04 [notice] 1#1: start worker process 34
2026/10/16 07:46:04 [notice] 1#1: start worker process 35
2026/10/16 07:46:04 [notice] 1#1: start worker process 36
2026/10/16 07:46:04 [notice] 1#1: start worker process 37
2026/10/16 07:46:04 [notice] 1#1: start worker process 38
2026/10/16 07:46:04 [notice] 1#1: start worker process 39
2026/10/16 07:46:04 [notice] 1#1: start worker process 40
2026/10/16 07:46:04 [notice] 1#1: start worker process 41
2026/10/16 07:46:04 [notice] 1#1: start worker process 42
2026/10/16 07:46:04 [notice] 1#1: start worker process 43
```

Запуск рабочих процессов веб-сервера.

- Команда `docker logs --tail 100 deploy-detect_save_worker-1` покажет работу сервиса сохранения событий

```
[+] Connected to queue
```

- Команда `docker logs --tail 100 deploy-slicer-1` покажет работу сервиса нарезки кадров из RTSP- и HTTP-источников с последующей передачей данных на распознавание объектов нейронными сетями:

```
2026/10/16 07:46:18 started http server on 0.0.0.0:8093
```

- Команда `docker logs --tail 100 deploy-mediatrix-1` покажет работу сервиса ретрансляции RTSP-потокков:

```
2026-10-16T07:46:02.372228933Z 2026/10/16 07:46:02 INF MediaMTX v1.5.1
2026-10-16T07:46:02.372257979Z 2026/10/16 07:46:02 INF configuration loaded from /mediatrix.yml
2026-10-16T07:46:02.372350773Z 2026/10/16 07:46:02 INF [metrics] listener opened on 0.0.0.0:9998
2026-10-16T07:46:02.372357129Z 2026/10/16 07:46:02 INF [pprof] listener opened on 0.0.0.0:9999
2026-10-16T07:46:02.372411955Z 2026/10/16 07:46:02 INF [RTSP] listener opened on :8554 (TCP)
2026-10-16T07:46:02.372425107Z 2026/10/16 07:46:02 INF [RTMP] listener opened on :1935
2026-10-16T07:46:02.372443414Z 2026/10/16 07:46:02 INF [HLS] listener opened on 0.0.0.0:5275
2026-10-16T07:46:02.372504989Z 2026/10/16 07:46:02 INF [WebRTC] listener opened on :8889 (HTTP), :8189 (ICE/UDP)
2026-10-16T07:46:02.372631786Z 2026/10/16 07:46:02 INF [SRT] listener opened on :8890 (UDP)
2026-10-16T07:46:02.372681766Z 2026/10/16 07:46:02 INF [API] listener opened on 0.0.0.0:9997
```

- Команда `docker logs --tail 100 deploy-rabbit-1` покажет работу брокера сообщений:

```

completed with 5 plugins.
2026-10-16 07:46:13.899842+00:00 [info] <0.704.0> Server startup complete; 5 plugins started.
2026-10-16 07:46:13.899842+00:00 [info] <0.704.0> * rabbitmq_prometheus
2026-10-16 07:46:13.899842+00:00 [info] <0.704.0> * rabbitmq_federation
2026-10-16 07:46:13.899842+00:00 [info] <0.704.0> * rabbitmq_management
2026-10-16 07:46:13.899842+00:00 [info] <0.704.0> * rabbitmq_management_agent
2026-10-16 07:46:13.899842+00:00 [info] <0.704.0> * rabbitmq_web_dispatch
2026-10-16 07:46:13.934812+00:00 [info] <0.866.0> accepting AMQP connection <0.866.0> (172.19.0.17:47040 -> 172.19.0.3:5672)
2026-10-16 07:46:13.934812+00:00 [info] <0.866.0> connection <0.866.0> (172.19.0.17:47040 -> 172.19.0.3:5672): user '*****' authenticated and granted access to vhost '/'
2026-10-16 07:46:13.996719+00:00 [info] <0.885.0> accepting AMQP connection <0.885.0> (172.19.0.11:37156 -> 172.19.0.3:5672)
2026-10-16 07:46:14.000292+00:00 [info] <0.885.0> connection <0.885.0> (172.19.0.11:37156 -> 172.19.0.3:5672): user '*****' authenticated and granted access to vhost '/'
2026-10-16 07:46:14.001498+00:00 [info] <0.9.0> Time to start RabbitMQ: 0117 ms
2026-10-16 07:46:14.056215+00:00 [info] <0.903.0> accepting AMQP connection <0.903.0> (172.19.0.14:51014 -> 172.19.0.3:5672)
2026-10-16 07:46:14.056540+00:00 [info] <0.909.0> accepting AMQP connection <0.909.0> (172.19.0.14:51028 -> 172.19.0.3:5672)
2026-10-16 07:46:14.056534+00:00 [info] <0.906.0> accepting AMQP connection <0.906.0> (172.19.0.14:51026 -> 172.19.0.3:5672)

```

- Команда `docker logs --tail 100 deploy-redis-1` покажет работу сервиса кэширования и хранения данных в оперативной памяти:

```

1:M 16 Oct 2026 07:46:02.631 * <timeseries> Redis version found by RedisTimeSeries : 8.0.2 - oss
1:M 16 Oct 2026 07:46:02.631 * <timeseries> Registering configuration options: [
1:M 16 Oct 2026 07:46:02.631 * <timeseries> { ts-compaction-policy :      }
1:M 16 Oct 2026 07:46:02.631 * <timeseries> { ts-num-threads :          3 }
1:M 16 Oct 2026 07:46:02.631 * <timeseries> { ts-retention-policy :      0 }
1:M 16 Oct 2026 07:46:02.631 * <timeseries> { ts-duplicate-policy :     block }
1:M 16 Oct 2026 07:46:02.631 * <timeseries> { ts-chunk-size-bytes :    4096 }
1:M 16 Oct 2026 07:46:02.631 * <timeseries> { ts-encoding :          compressed }
1:M 16 Oct 2026 07:46:02.631 * <timeseries> { ts-ignore-max-time-diff:      0 }
1:M 16 Oct 2026 07:46:02.631 * <timeseries> { ts-ignore-max-val-diff :    0.000000 }
1:M 16 Oct 2026 07:46:02.631 * <timeseries> ]
1:M 16 Oct 2026 07:46:02.631 * <timeseries> Detected redis oss
1:M 16 Oct 2026 07:46:02.631 * Module 'timeseries' loaded from /usr/local/lib/redis/modules//redistimeseries.so
1:M 16 Oct 2026 07:46:02.639 * <ReJSON> Created new data type 'ReJSON-RL'
1:M 16 Oct 2026 07:46:02.639 * <ReJSON> version: 80001 git sha: unknown branch: unknown
1:M 16 Oct 2026 07:46:02.639 * <ReJSON> Exported RedisJSON_V1 API
1:M 16 Oct 2026 07:46:02.639 * <ReJSON> Exported RedisJSON_V2 API
1:M 16 Oct 2026 07:46:02.639 * <ReJSON> Exported RedisJSON_V3 API
1:M 16 Oct 2026 07:46:02.639 * <ReJSON> Exported RedisJSON_V4 API
1:M 16 Oct 2026 07:46:02.639 * <ReJSON> Exported RedisJSON_V5 API
1:M 16 Oct 2026 07:46:02.639 * <ReJSON> Enabled diskless replication
1:M 16 Oct 2026 07:46:02.639 * <ReJSON> Initialized shared string cache, thread safe: false.
1:M 16 Oct 2026 07:46:02.639 * Module 'ReJSON' loaded from /usr/local/lib/redis/modules//rejson.so
1:M 16 Oct 2026 07:46:02.639 * <search> Acquired RedisJSON_V5 API
1:M 16 Oct 2026 07:46:02.639 * Server initialized
1:M 16 Oct 2026 07:46:02.639 * Ready to accept connections tcp

```

6. ВРЕМЯ РАБОТЫ И КОНТАКТЫ ТЕХНИЧЕСКОЙ ПОДДЕРЖКИ

Время работы технической поддержки:

понедельник-пятница с 9.00 – 18.00 по московскому времени.

Технический специалист: Елисеев Михаил
Дмитриевич.

Телефон тех. специалиста: +7 (906) 768-50-68

Почта: info@linktelekom.ru