МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра математического обеспечения и применения ЭВМ

ОТЧЕТ

по лабораторной работе №3

"Регрессионная модель изменения цен на дома в Бостоне"

по дисциплине «Искусственные нейронные сети»

Студентка гр. 8383 _____ Ишанина Л.Н.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург

2021

Цель работы.

Реализовать предсказание медианной цены на дома в пригороде Бостона в середине 1970-х по таким данным, как уровень преступности, ставка местного имущественного налога и т. д.

Данный набор содержит относительно немного образцов данных: всего 506, разбитых на 404 обучающих и 102 контрольных образца. И каждый признак во входных данных (например, уровень преступности) имеет свой масштаб. Например, некоторые признаки являются пропорциями и имеют значения между 0 и 1, другие — между 1 и 12 и т. д.

Задание.

- Ознакомиться с задачей регрессии
- Изучить отличие задачи регрессии от задачи классификации
- Создать модель
- Настроить параметры обучения
- Обучить и оценить модели
- Ознакомиться с перекрестной проверкой

Объяснить различия задач классификации и регрессии

Изучить влияние кол-ва эпох на результат обучения модели

Выявить точку переобучения

Применить перекрестную проверку по K блокам при различных K

Построить графики ошибки и точности во время обучения для моделей, а также усредненные графики по всем моделям.

Выполнение работы.

Классификация - это процесс поиска или обнаружения модели (функции), которая помогает разделить данные на несколько категориальных

классов. При классификации определяется членство группы в проблеме, что означает, что данные классифицируются под разными метками в соответствии с некоторыми параметрами, а затем метки прогнозируются для данных. Прогнозирующее регрессионное моделирование - это задача приближения функции отображения (f) от входных переменных (X) к непрерывной выходной переменной (y).

Непрерывная выходная переменная - это действительное значение, такое как целое число или значение с плавающей запятой. Это часто количества, такие как суммы и размеры.

Например, можно предположить, что дом будет продаваться по определенной долларовой стоимости, возможно, в диапазоне от 100 000 до 200 000 долларов.

Набор данных присутствует в составе Keras.

Были импортированы необходимые для работы классы и функции, а также загружены данные. Листинг представлен ниже:

```
import numpy as np
from tensorflow.keras.layers import Dense
from tensorflow.keras.models import Sequential
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.datasets import boston_housing
(train_data,    train_targets),    (test_data,    test_targets)    =
boston_housing.load_data()
print(train_data.shape)
print(test_data.shape)
print(test_targets)
```

Далее к данным была применена нормализация: для каждого признака во входных данных (столбца в матрице входных данных) из каждого значения вычитается среднее по этому признаку, и разность делится на стандартное отклонение, в результате признак центрируется по нулевому значению и имеет стандартное отклонение, равное единице. Такую нормализацию легко выполнить с помощью Numpy. Листинг представлен ниже:

```
mean = train_data.mean(axis=0)
```

```
train_data -= mean
std = train_data.std(axis=0)
train_data /= std
test_data -= mean
test_data /= std
```

Затем была создана функция `build_model()`. Листинг приведен ниже:

```
def build_model():
    model = Sequential()
    model.add(Dense(64, activation='relu',
input_shape=(train_data.shape[1],)))
    model.add(Dense(64, activation='relu'))
    model.add(Dense(1))
    model.compile(optimizer='rmsprop', loss='mse', metrics=['mae'])
    return model
```

Сеть заканчивается одномерным слоем, не имеющим функции активации (это линейный слой). Это типичная конфигурация для скалярной регрессии (целью которой является предсказание одного значения на непрерывной числовой прямой). Применение функции активации могло бы ограничить диапазон выходных значений: например, если в последнем слое применить функцию активации sigmoid, сеть обучилась бы предсказывать только значения из диапазона между 0 и 1.
В данном случае, с линейным последним слоем, сеть способна предсказывать значения из любого диапазона.

Далее, чтобы надежно оценить качество модели, была применена перекрестная проверка по K блокам (K-fold cross-validation). Суть ее заключается в разделении доступных данных на K блоков (обычно K = 4 или 5), создании K идентичных моделей и обучении каждой на K—1 блоках с оценкой по оставшимся блокам. По полученным K оценкам вычисляется среднее значение, которое принимается как оценка модели. Листинг представлен ниже:

```
k = 4
num_val_samples = len(train_data) // k
num_epochs = 100
all_scores = []
for i in range(k):
    print('processing fold #', i)
```

```
    val_data = train_data[i * num_val_samples: (i + 1) *
num_val_samples]
    val_targets = train_targets[i * num_val_samples: (i + 1) *
num_val_samples]
    partial_train_data = np.concatenate([train_data[:i *
num_val_samples], train_data[(i + 1) * num_val_samples:]],
                                        axis=0)
    partial_train_targets = np.concatenate(
        [train_targets[:i * num_val_samples], train_targets[(i + 1)
* num_val_samples:]], axis=0)
    model = build_model()
    model.fit(partial_train_data, partial_train_targets,
epochs=num_epochs, batch_size=1, verbose=0)
    val_mse, val_mae = model.evaluate(val_data, val_targets,
verbose=0)
    all_scores.append(val_mae)
print(np.mean(all_scores))
```

Для построения графика ошибок и точности в ходе обучения, а также усредненных графиков по всем моделям, была подключена библиотека matplotlib. Листинг подключения представлен ниже:

```
import matplotlib.pyplot as plt
```

Затем были построены сами графики. Листинг приведен ниже:

```
# Получение ошибки и точности в процессе обучения


    loss = H.history['loss']
    val_loss = H.history['val_loss']
    mae = H.history['mean_absolute_error']
    val_mae = H.history['val_mean_absolute_error']
    all_mae.append(val_mae)


    epochs = range(1, len(loss) + 1)


    # Построение графика ошибки


    plt.plot(epochs, loss, 'm*', label='Training loss')
    plt.plot(epochs, val_loss, 'b', label='Validation loss')
    plt.title('Training and validation loss')
    plt.xlabel('Epochs')
    plt.ylabel('Loss')
    plt.legend()
    plt.show()
```

```
    # Построение графика mae

    plt.clf()
    plt.plot(epochs, mae, 'm*', label='Training mae')
    plt.plot(epochs, val_mae, 'b', label='Validation mae')
    plt.title('Training and validation accuracy')
    plt.xlabel('Epochs')
    plt.ylabel('Accuracy')
    plt.legend()
    plt.show()

# Построение итогового графика
for i in range(num_epochs):
    arr.append(np.mean([epochs[i] for epochs in all_mae]))
plt.plot(range(1, num_epochs + 1), arr, 'g')
plt.xlabel('Epochs')
plt.ylabel('MAE')
plt.show()
```

Анализ различных искусственных нейронных сетей.

Изначально, была модель с параметрами обучения и перекрестной проверки:

```
k = 4
num_epochs = 100
```

Результат тестирования представлен на рис.1-8.

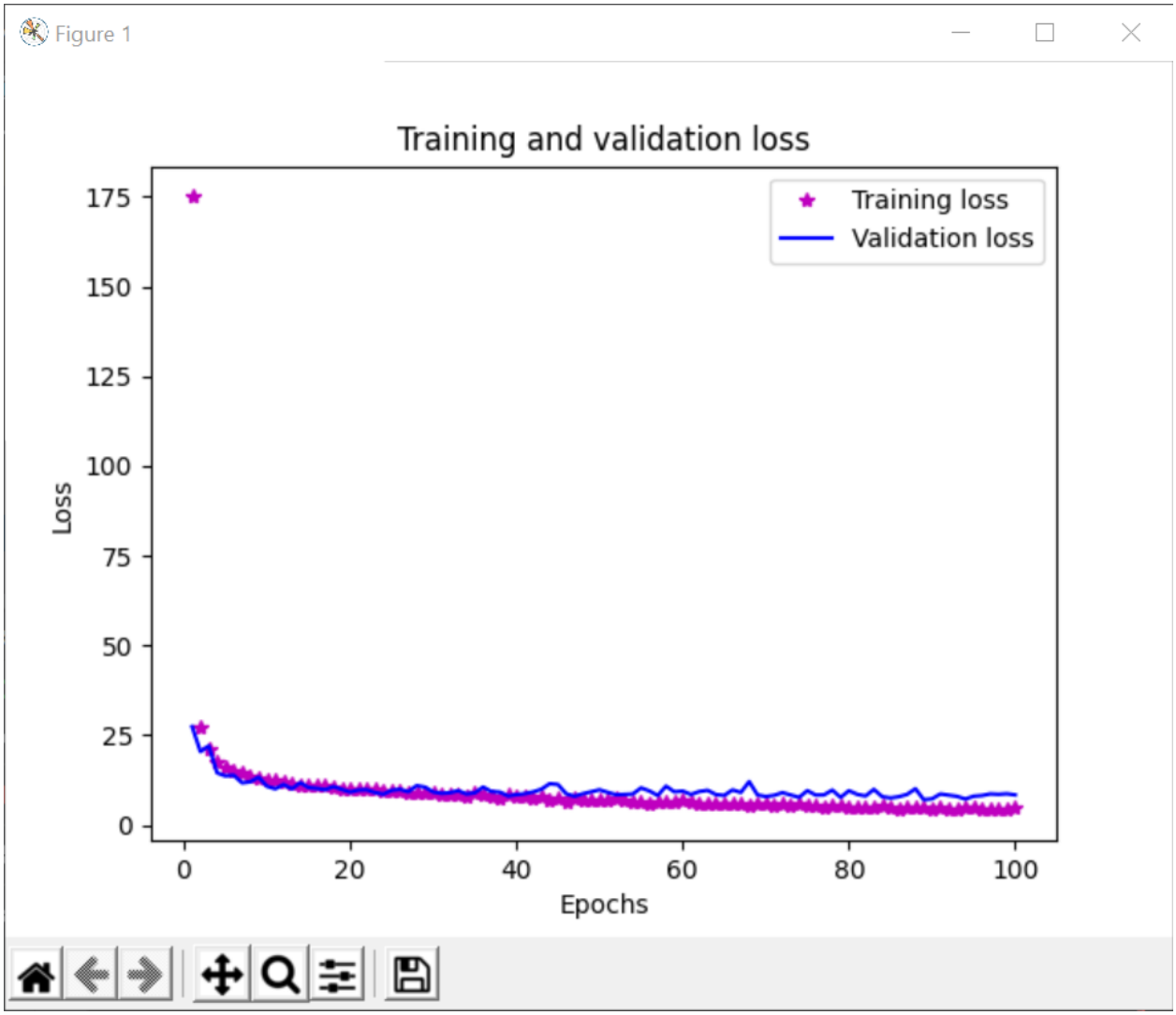

Рисунок 1 – графики потери сети на обучающих данных и данных, не участвовавших в обучении первого блока.

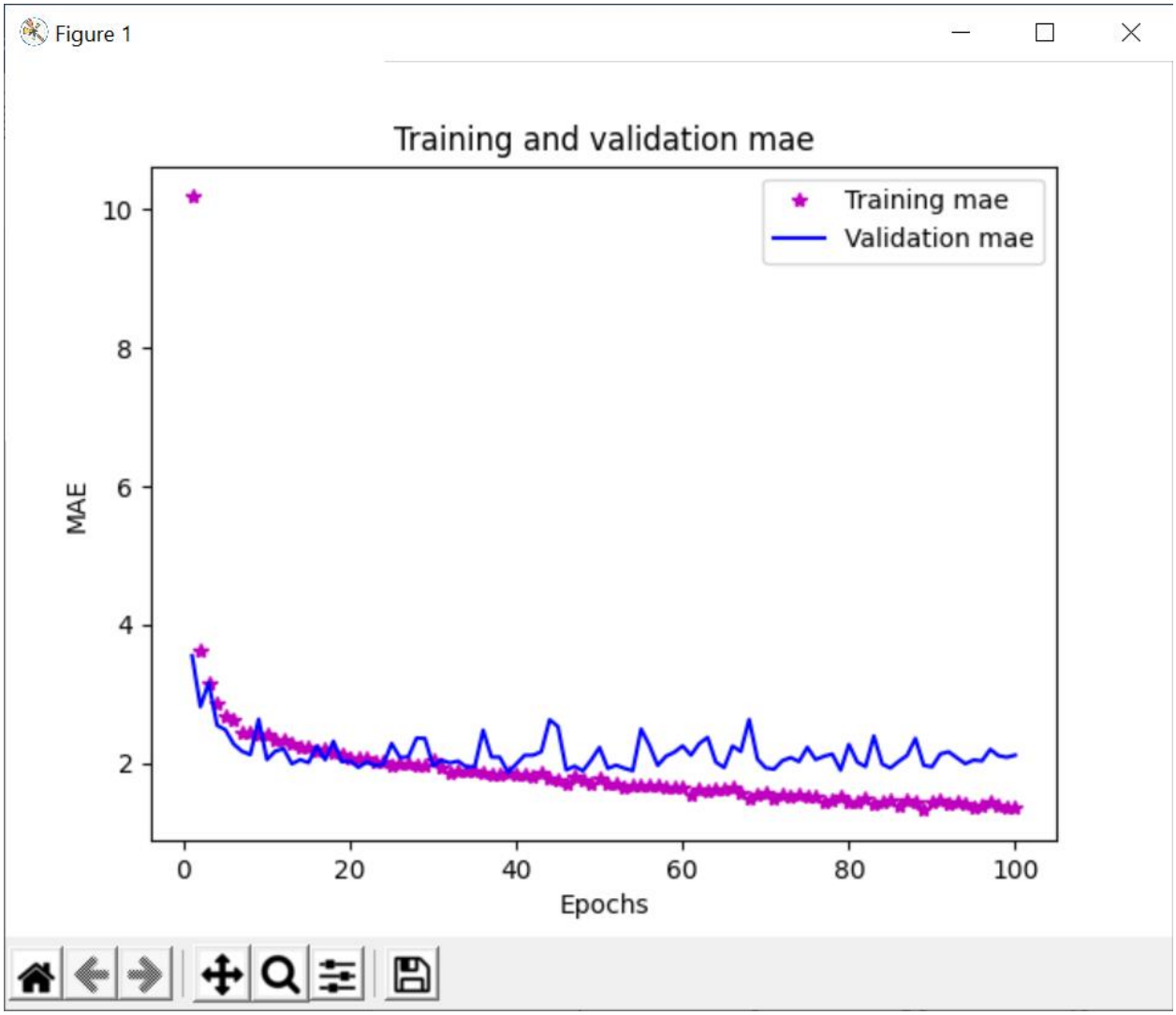

Рисунок 2 – графики оценки средней абсолютной ошибки первого блока.

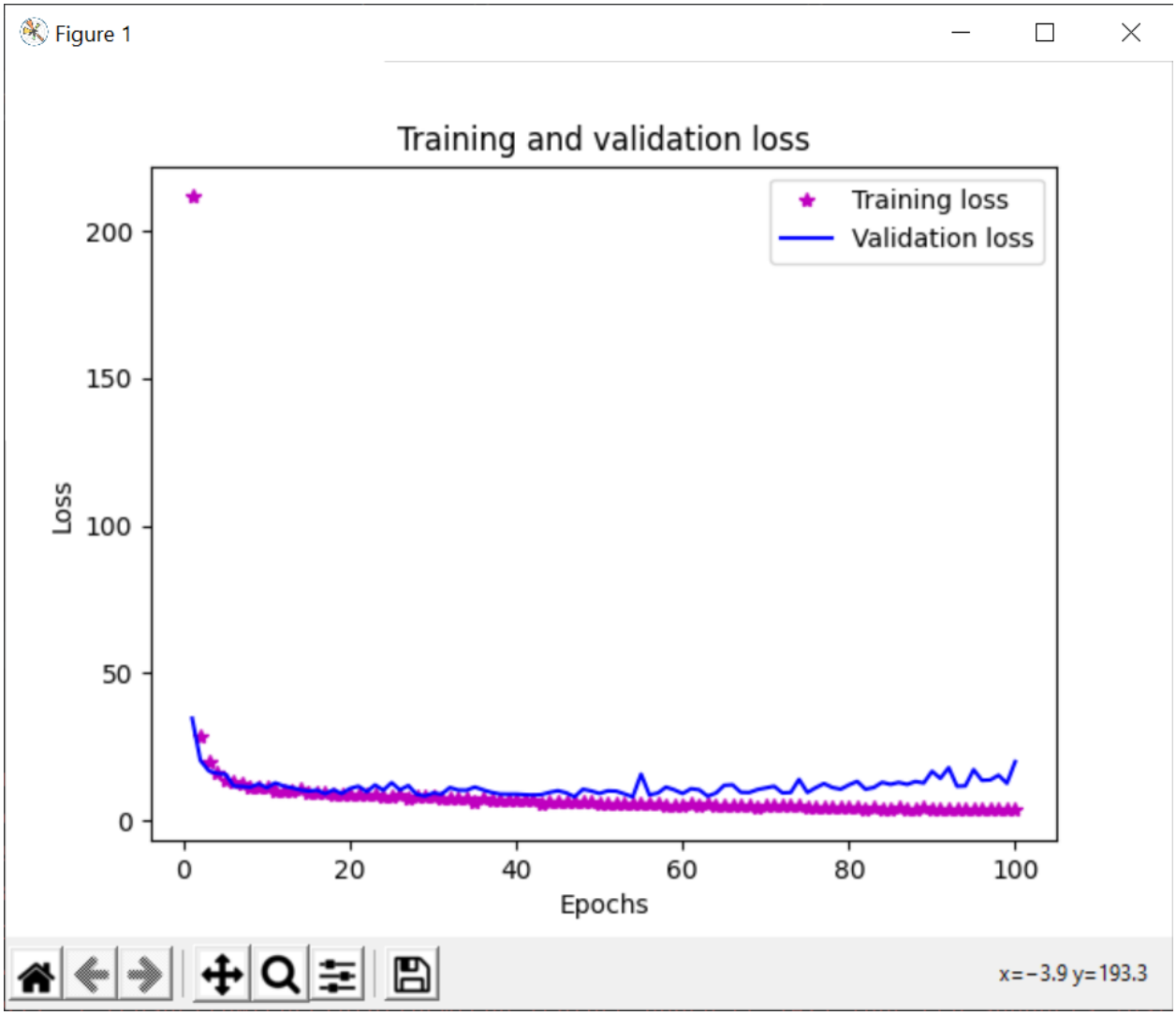

Рисунок 3 – графики потери сети на обучающих данных и данных, не участвовавших в обучении второго блока.

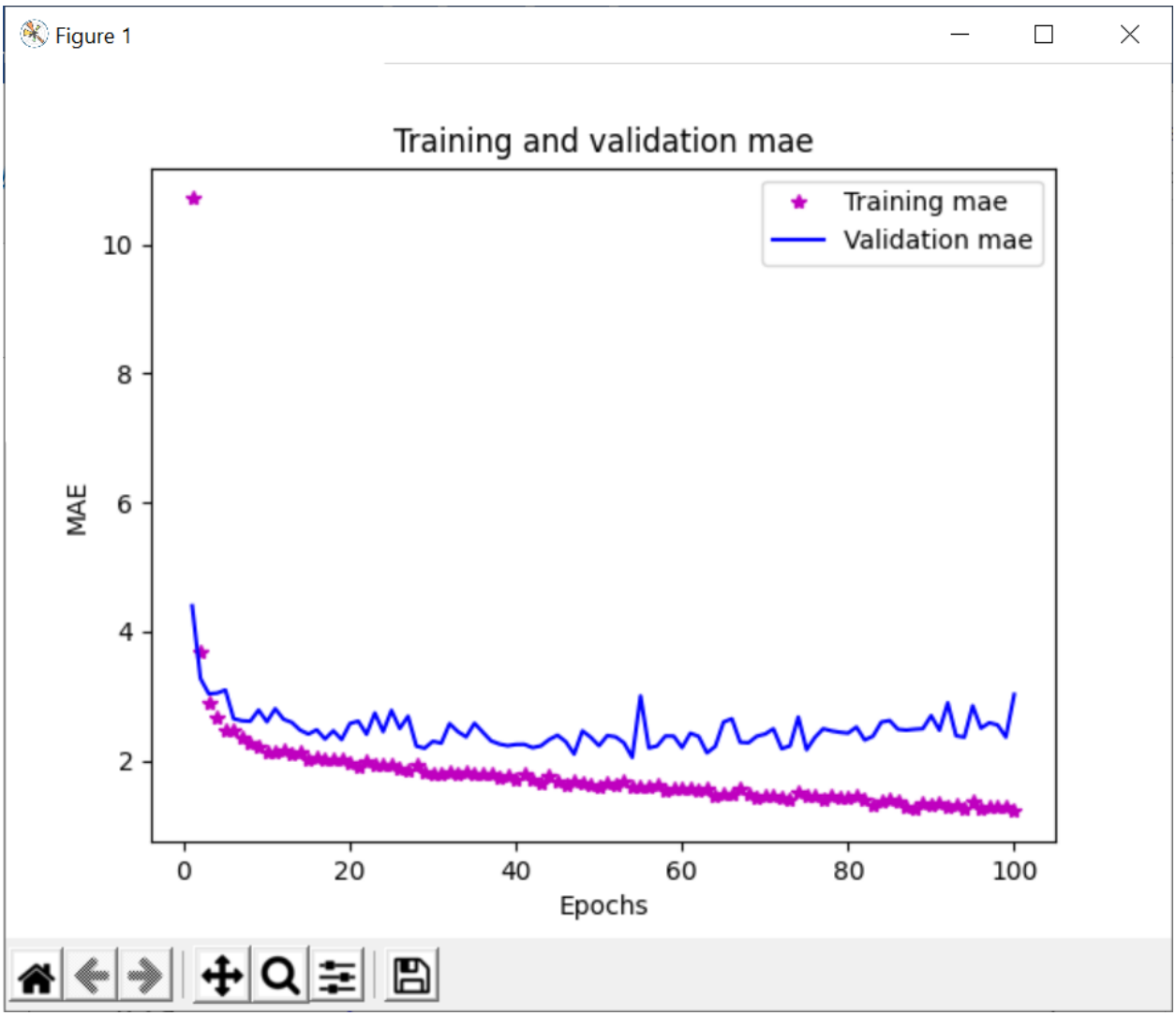

Рисунок 4 – графики оценки средней абсолютной ошибки второго блока.

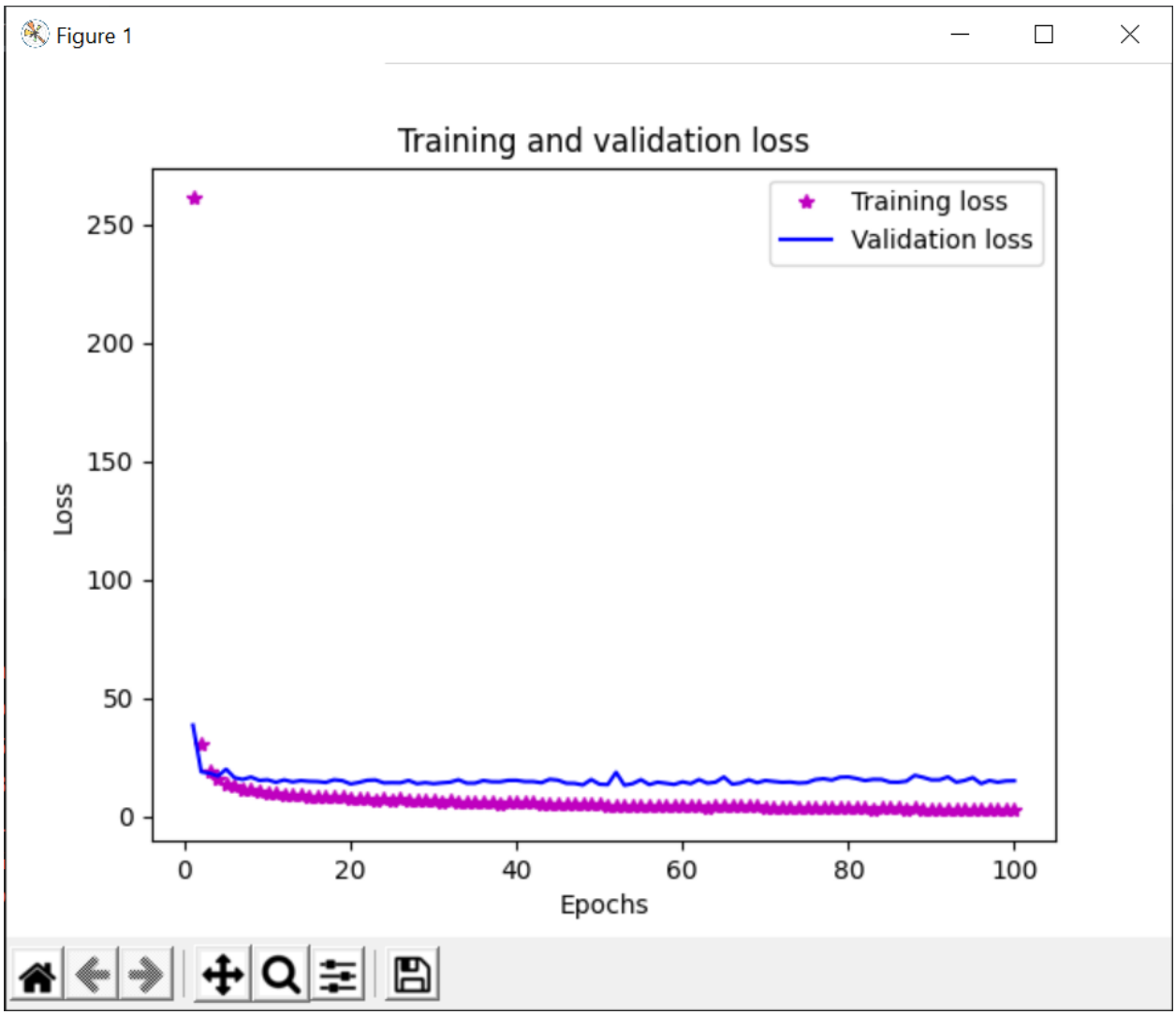

Рисунок 5 – графики потери сети на обучающих данных и данных, не участвовавших в обучении третьего блока.

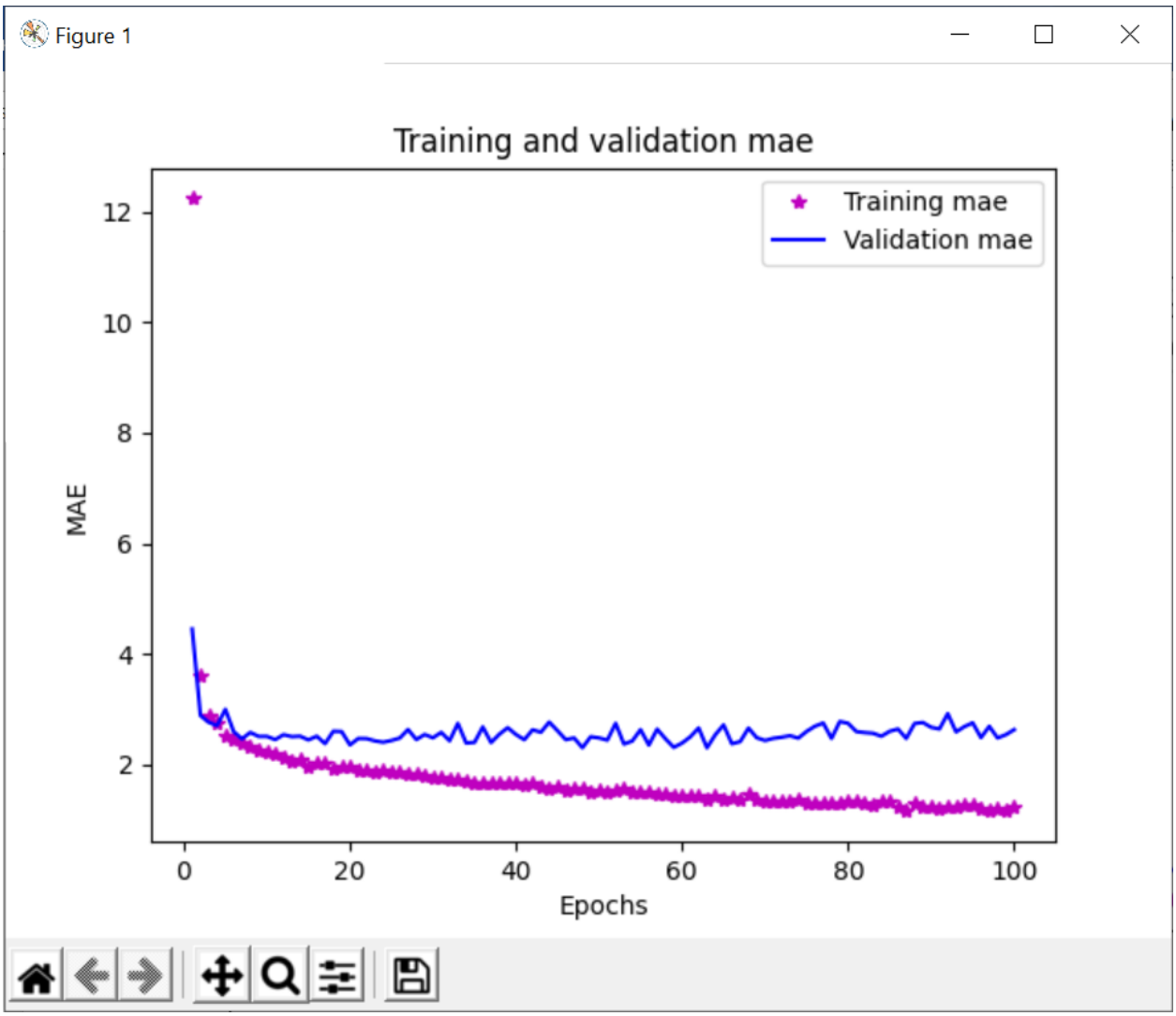

Рисунок 6 – графики оценки средней абсолютной ошибки третьего блока.

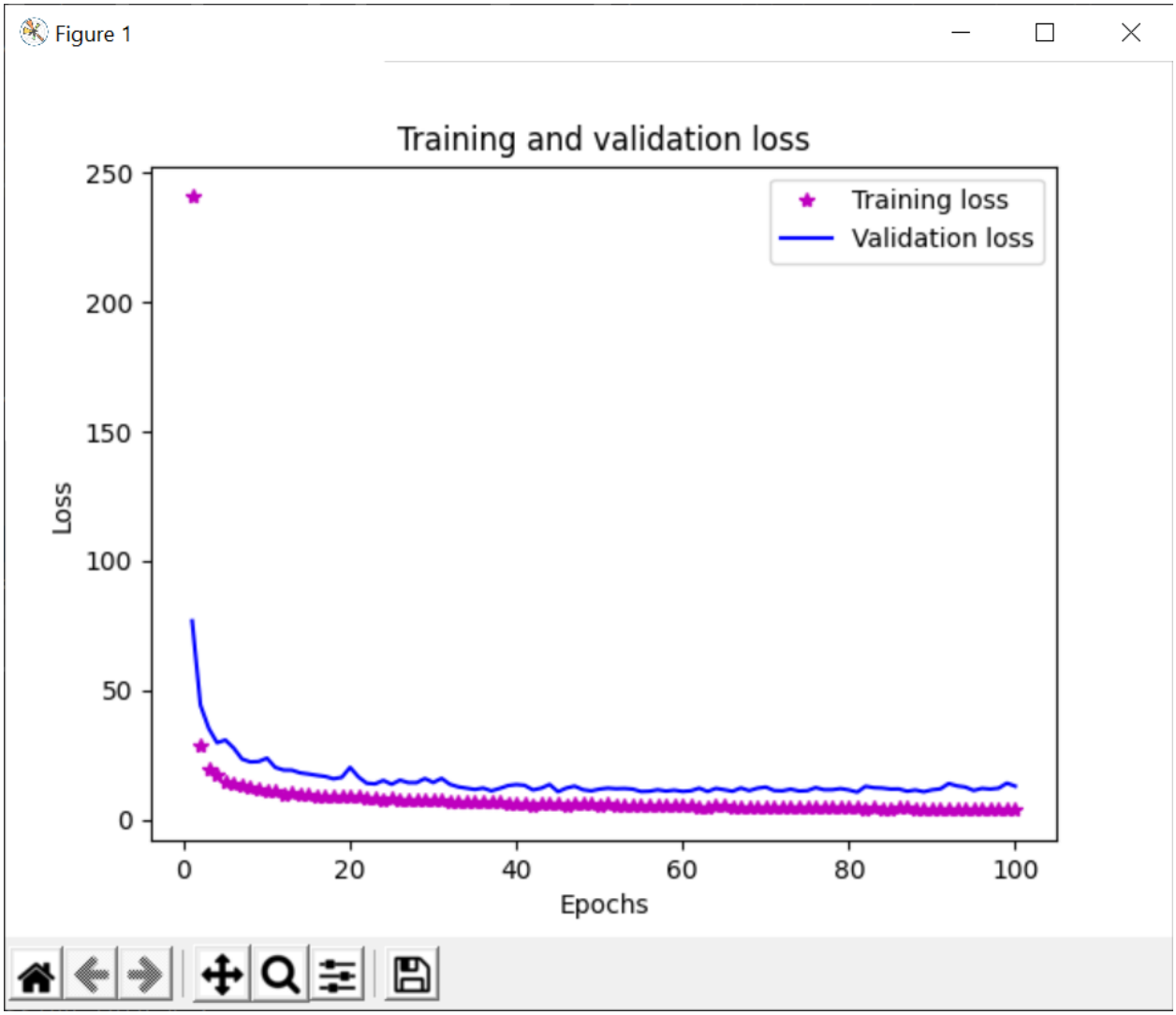

Рисунок 7 – графики потери сети на обучающих данных и данных, не участвовавших в обучении четвертого блока.

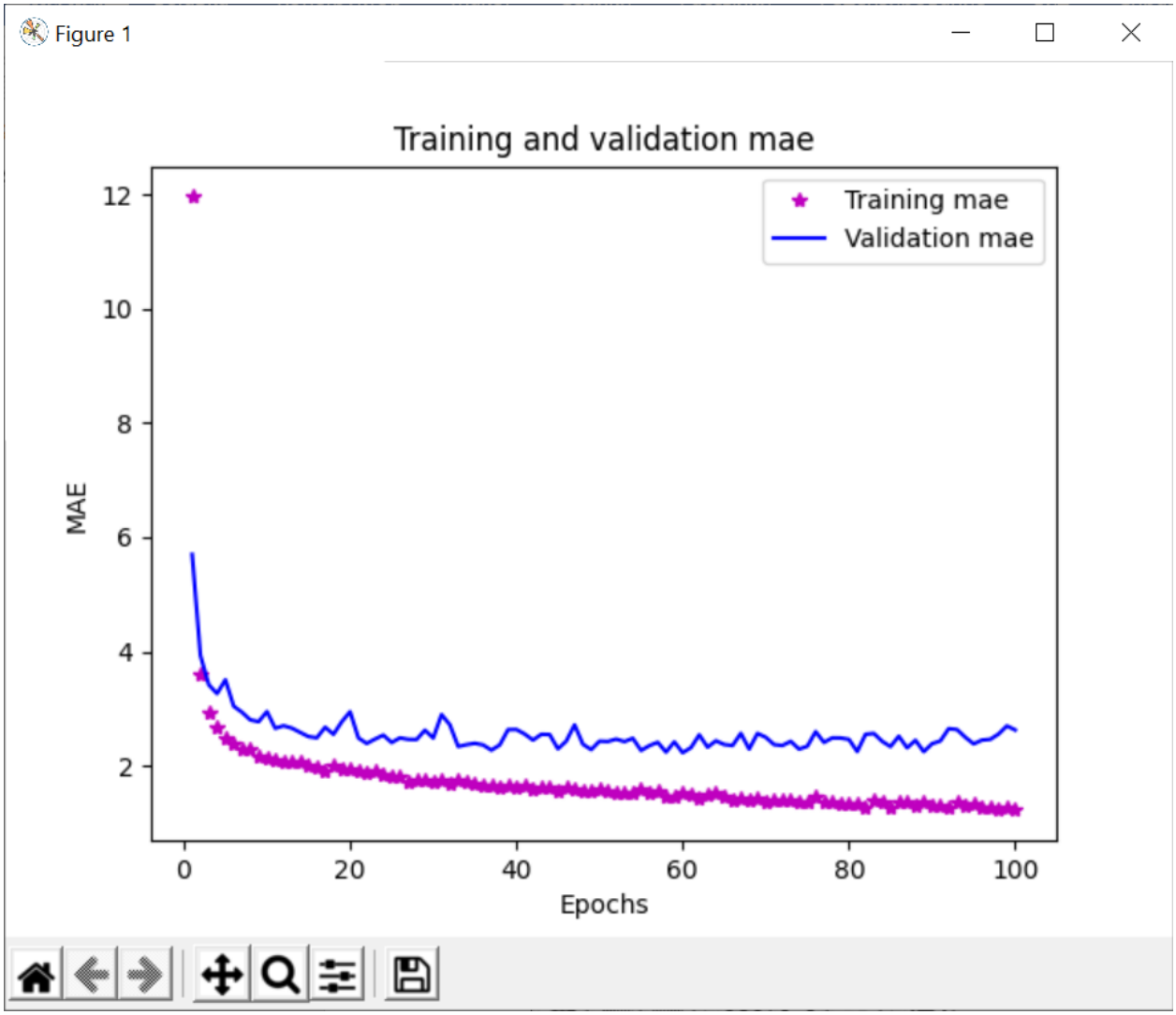

Рисунок 7 – графики оценки средней абсолютной ошибки четвертого блока.

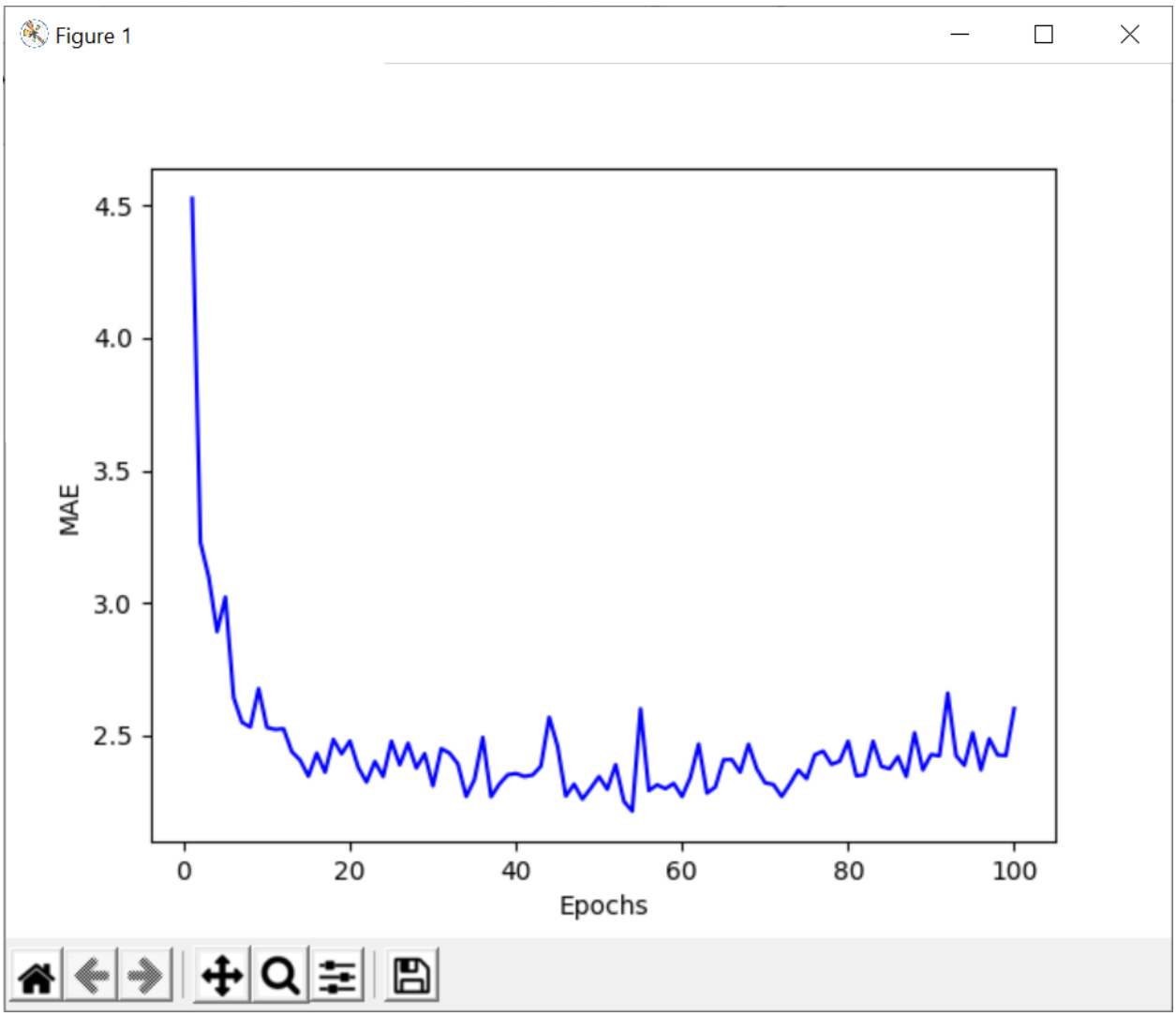

Рисунок 8 – график среднего значения оценки средней абсолютной ошибки

Средняя mae после полного обучения по k блокам равна 2.6006837.

Исходя из графика, представленного выше, можно сделать вывод, что примерно на 50 эпохе происходит переобучение, а значит количество эпох можно снизить до 50.

Таким образом, была выбрана модель с параметрами обучения и перекрестной проверки:

```
k = 4
num_epochs = 50
```

Результат тестирования представлен на рис.9-17.

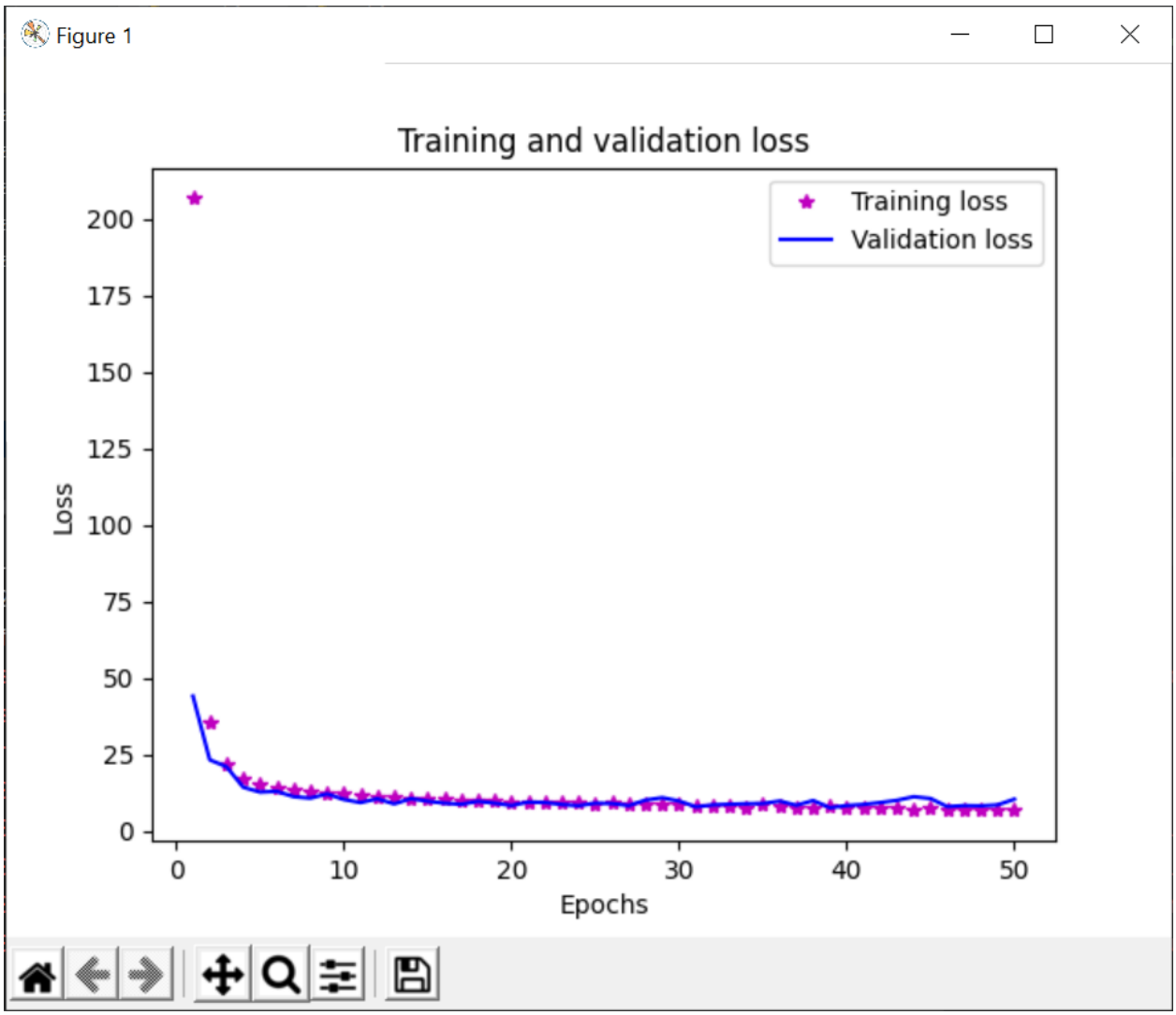

Рисунок 9 – графики потери сети на обучающих данных и данных, не участвовавших в обучении первого блока.

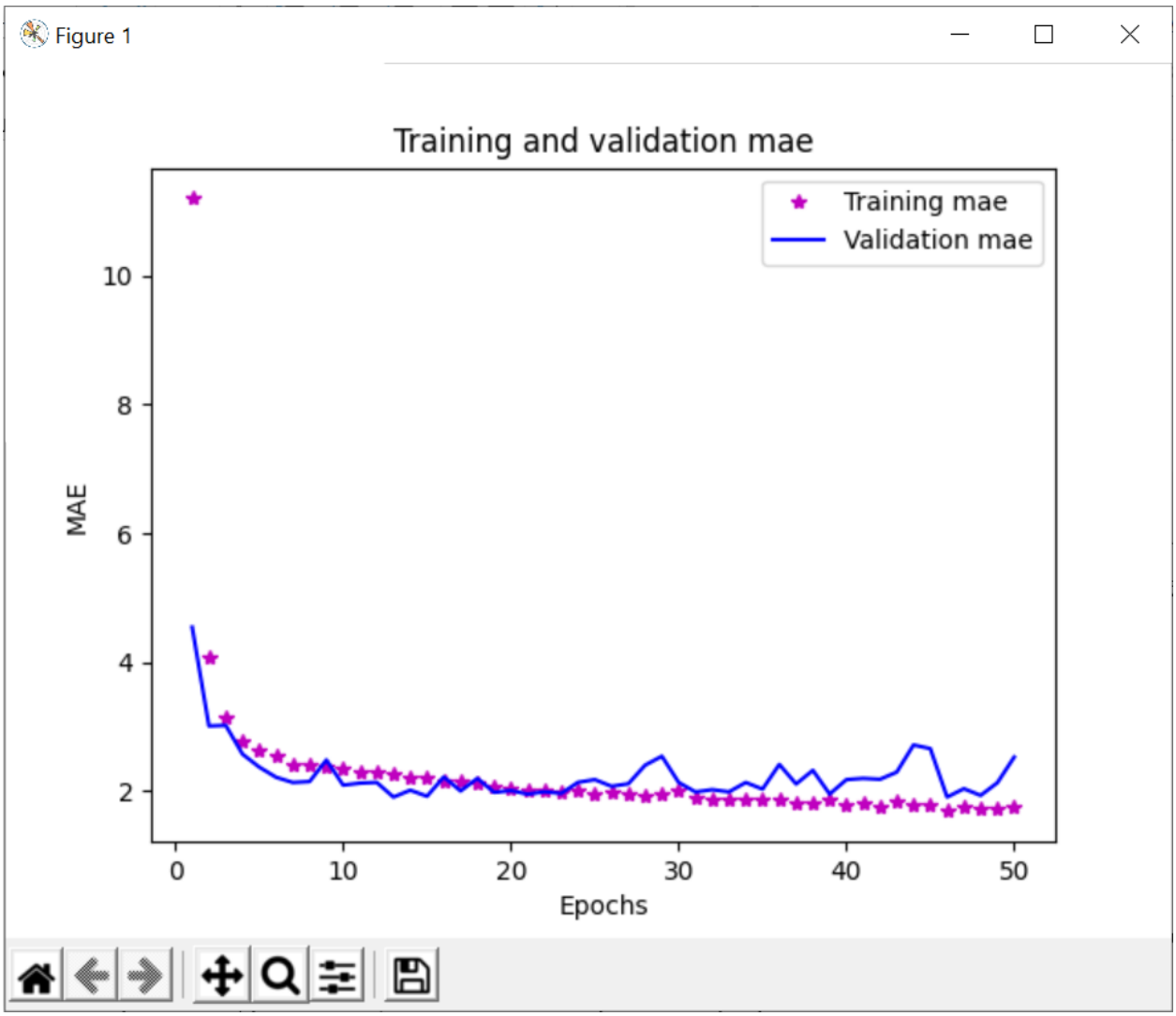

Рисунок 10 – графики оценки средней абсолютной ошибки первого блока.

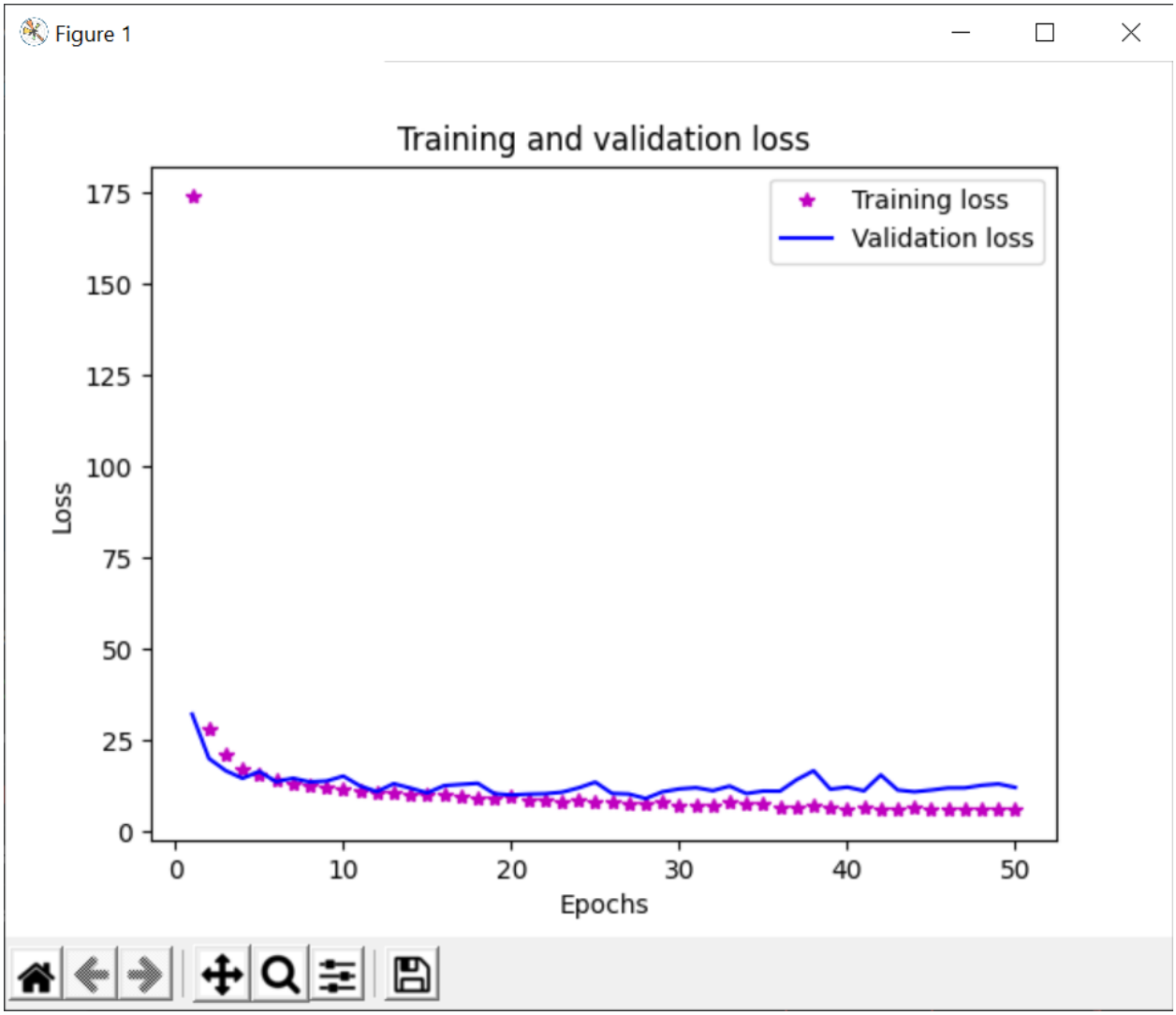

Рисунок 11 – графики потери сети на обучающих данных и данных, не участвовавших в обучении второго блока.

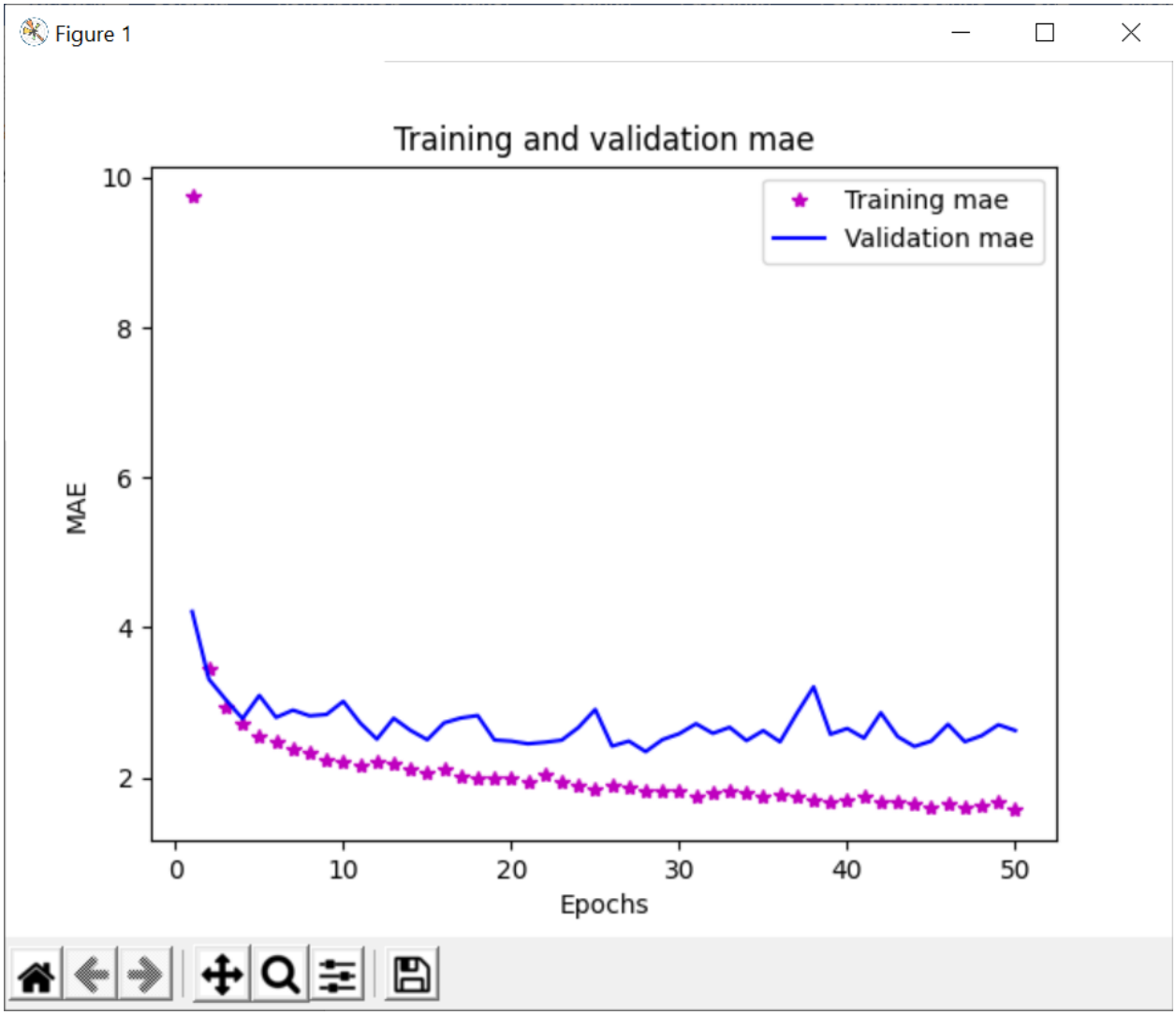

Рисунок 12 – графики оценки средней абсолютной ошибки второго блока.

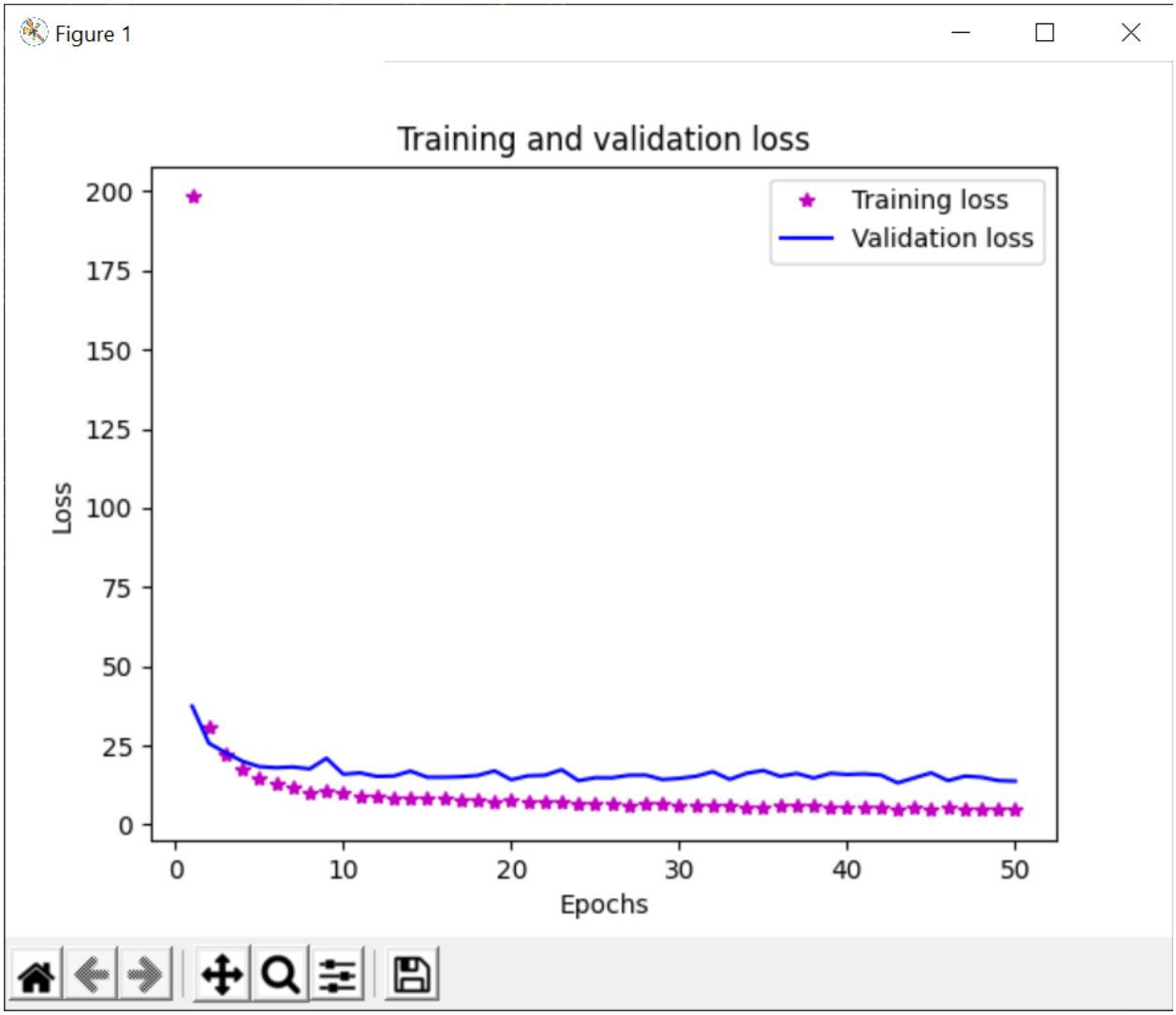

Рисунок 13 – графики потери сети на обучающих данных и данных, не участвовавших в обучении третьего блока.

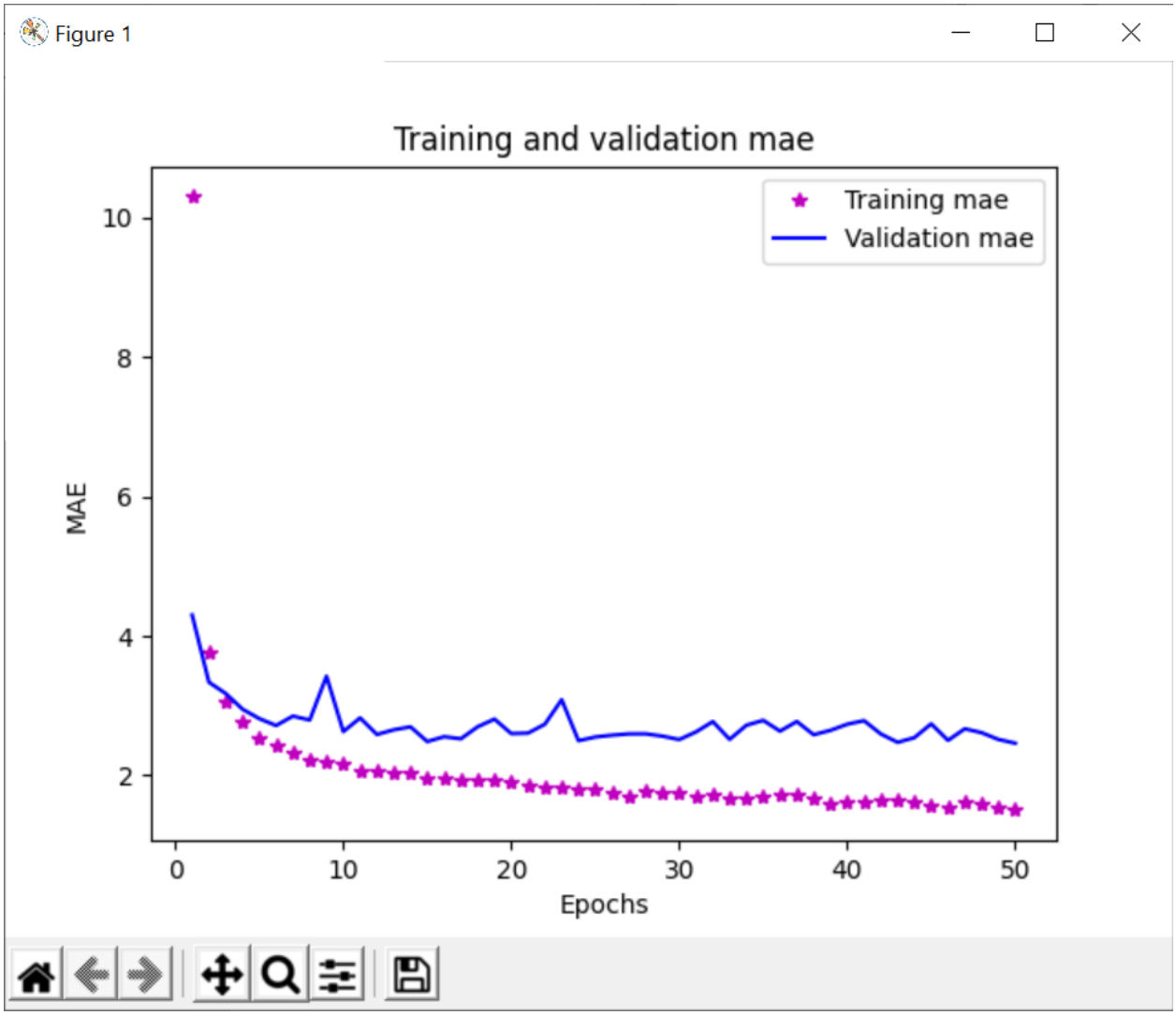

Рисунок 14 – графики оценки средней абсолютной ошибки третьего блока.

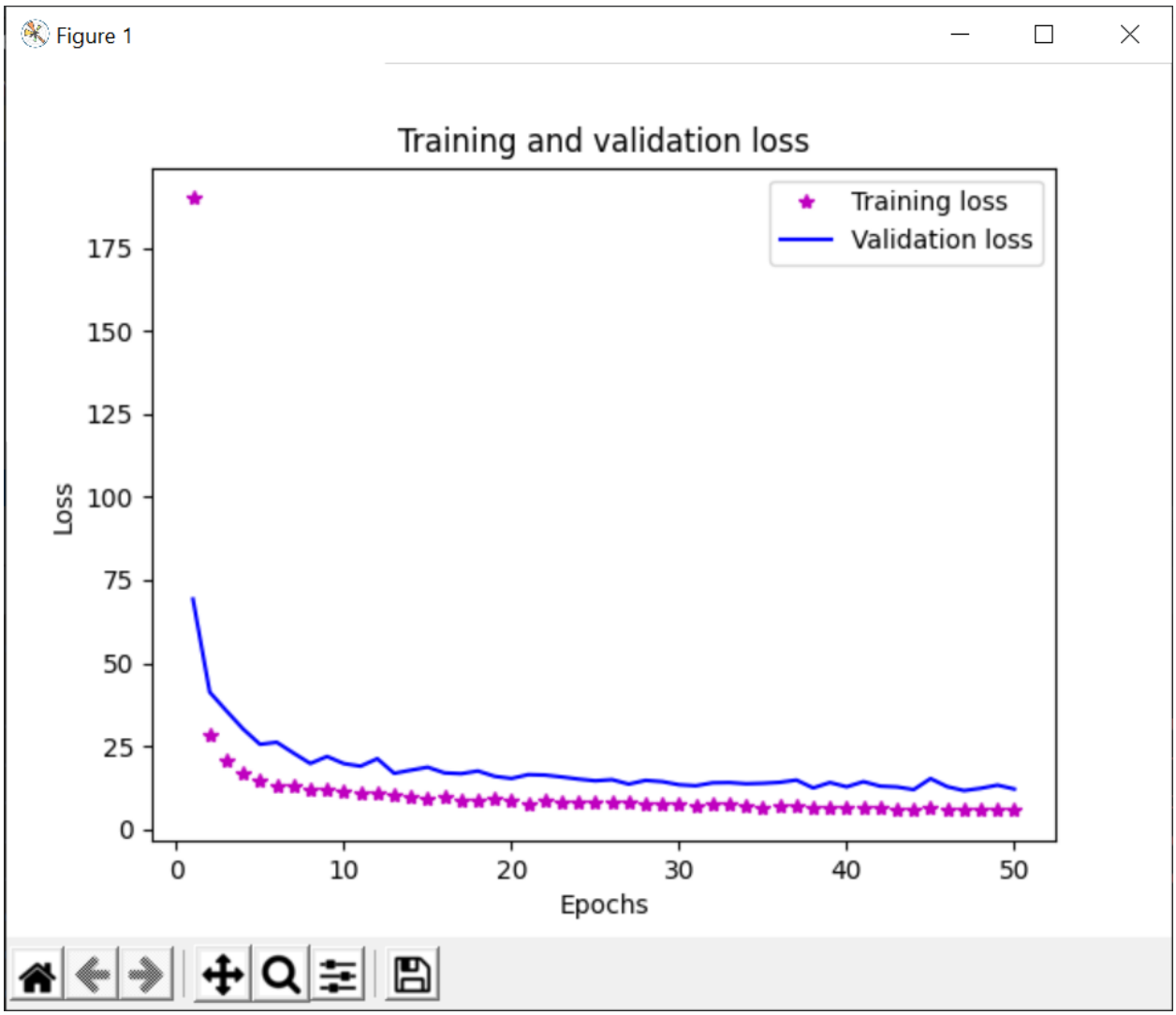

Рисунок 15 – графики потери сети на обучающих данных и данных, не участвовавших в обучении четвертого блока.

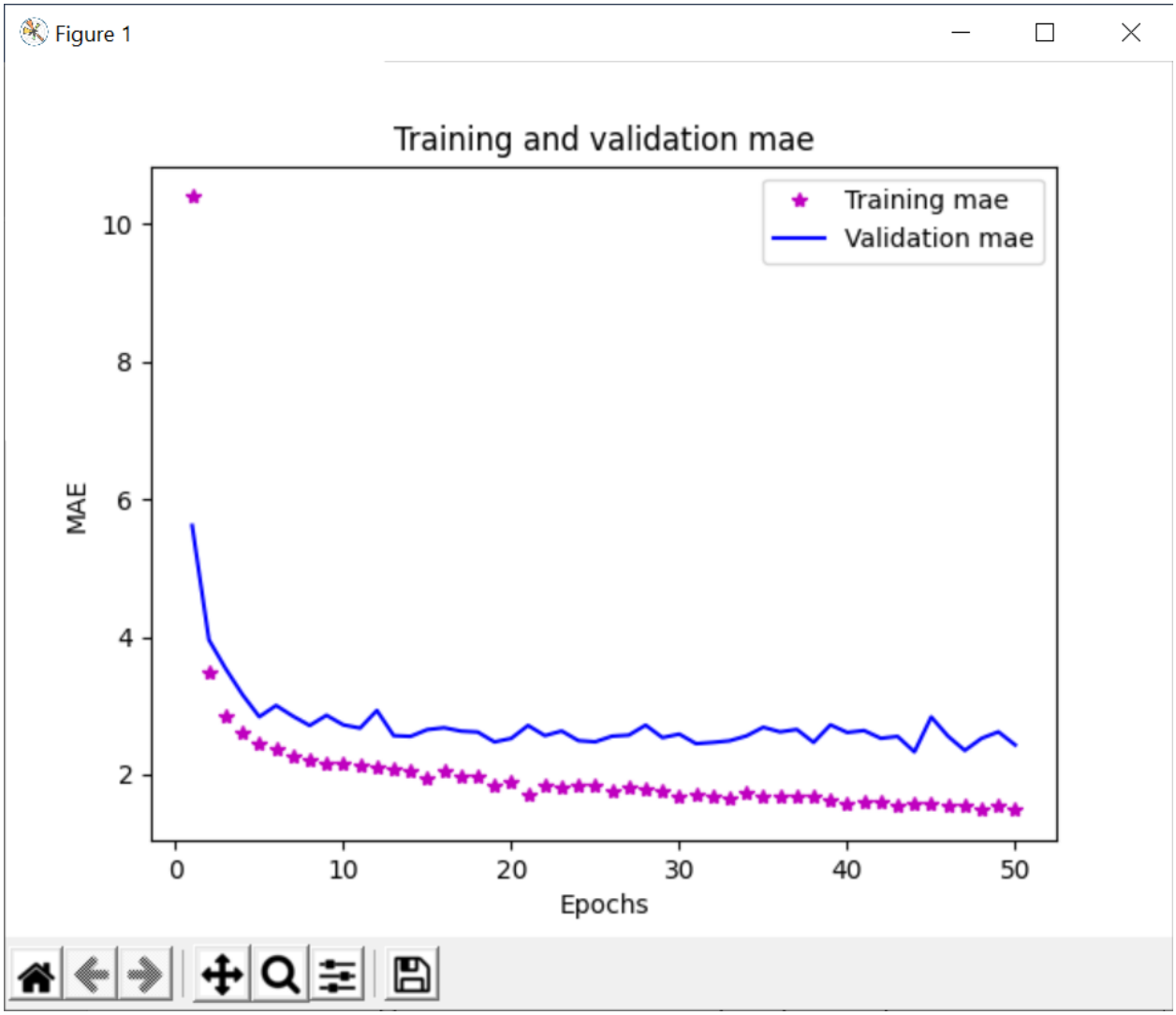

Рисунок 16 – графики оценки средней абсолютной ошибки четвертого блока.

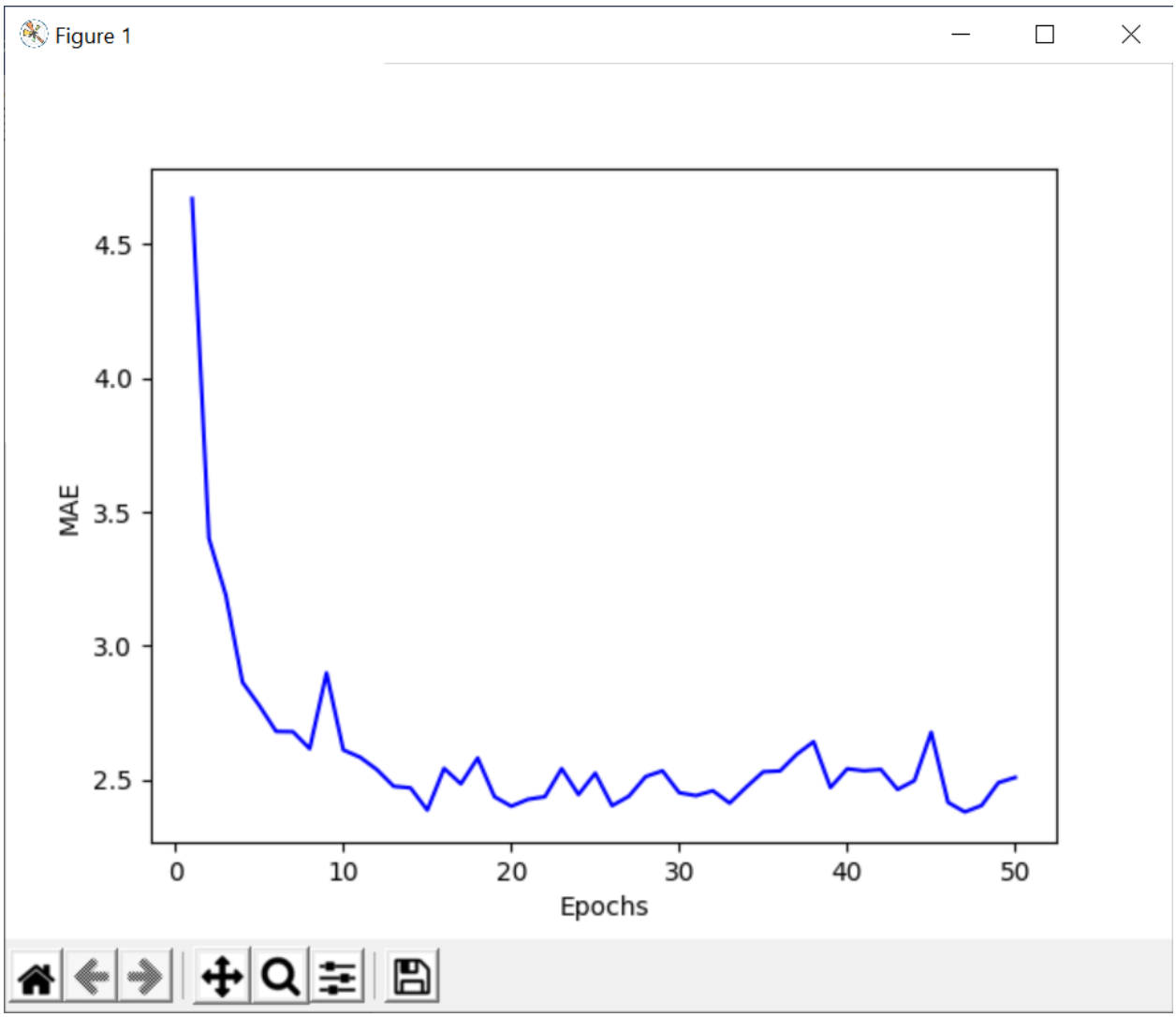

Рисунок 17 – график среднего значения оценки средней абсолютной ошибки

Средняя mae после полного обучения по k блокам равна 2.5364273

Исходя из графика среднего значения оценки средней абсолютной ошибки видно, что примерно после 35 всё равно начинает происходить переобучение, а значит, что количество эпох можно ещё сократить до 35 эпох.

Таким образом, была выбрана модель с параметрами обучения и перекрестной проверки:

```
k = 4
num_epochs = 35
```

Результат тестирования представлен на рис.18-26.

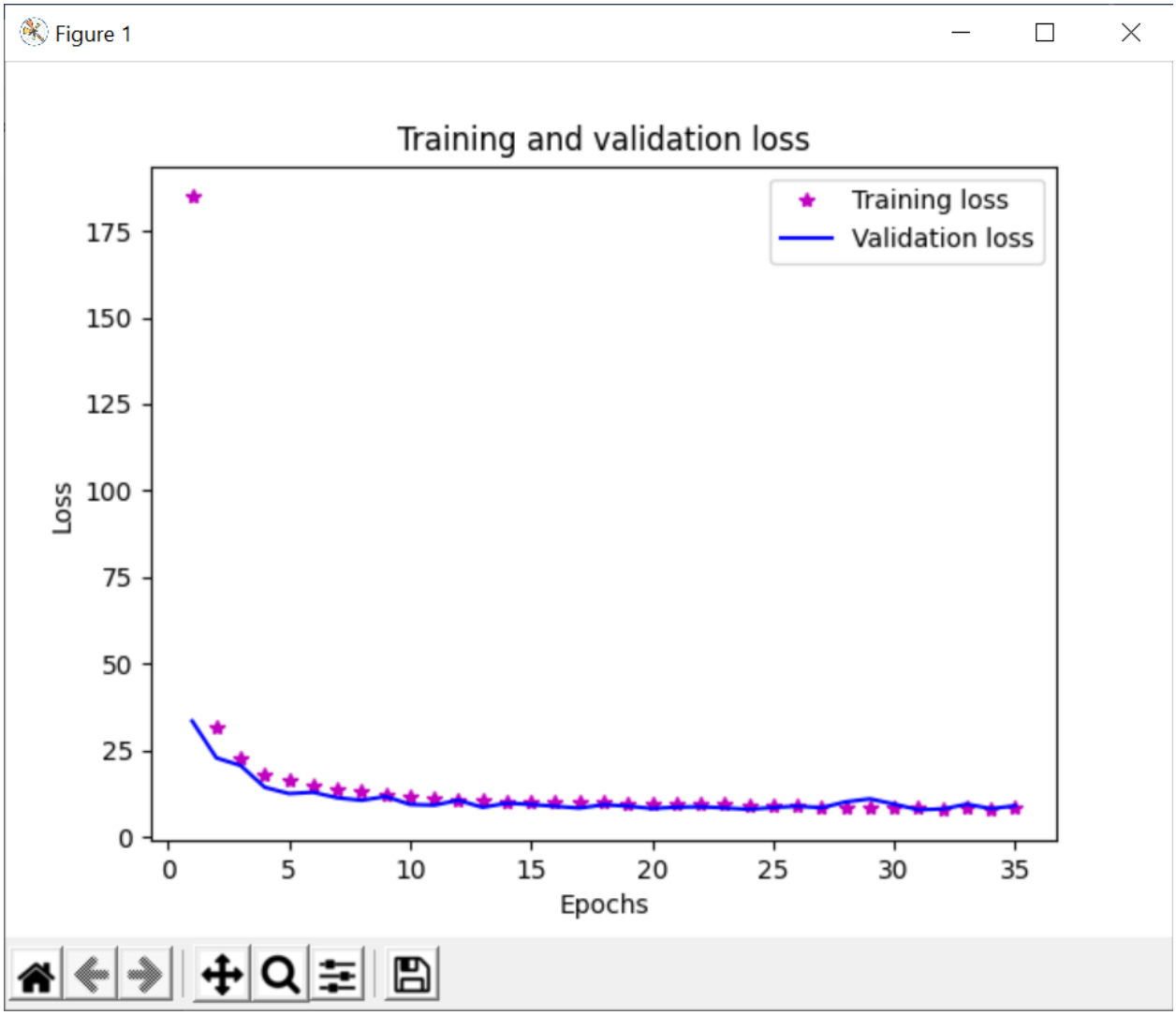

Рисунок 18 – графики потери сети на обучающих данных и данных, не участвовавших в обучении первого блока.

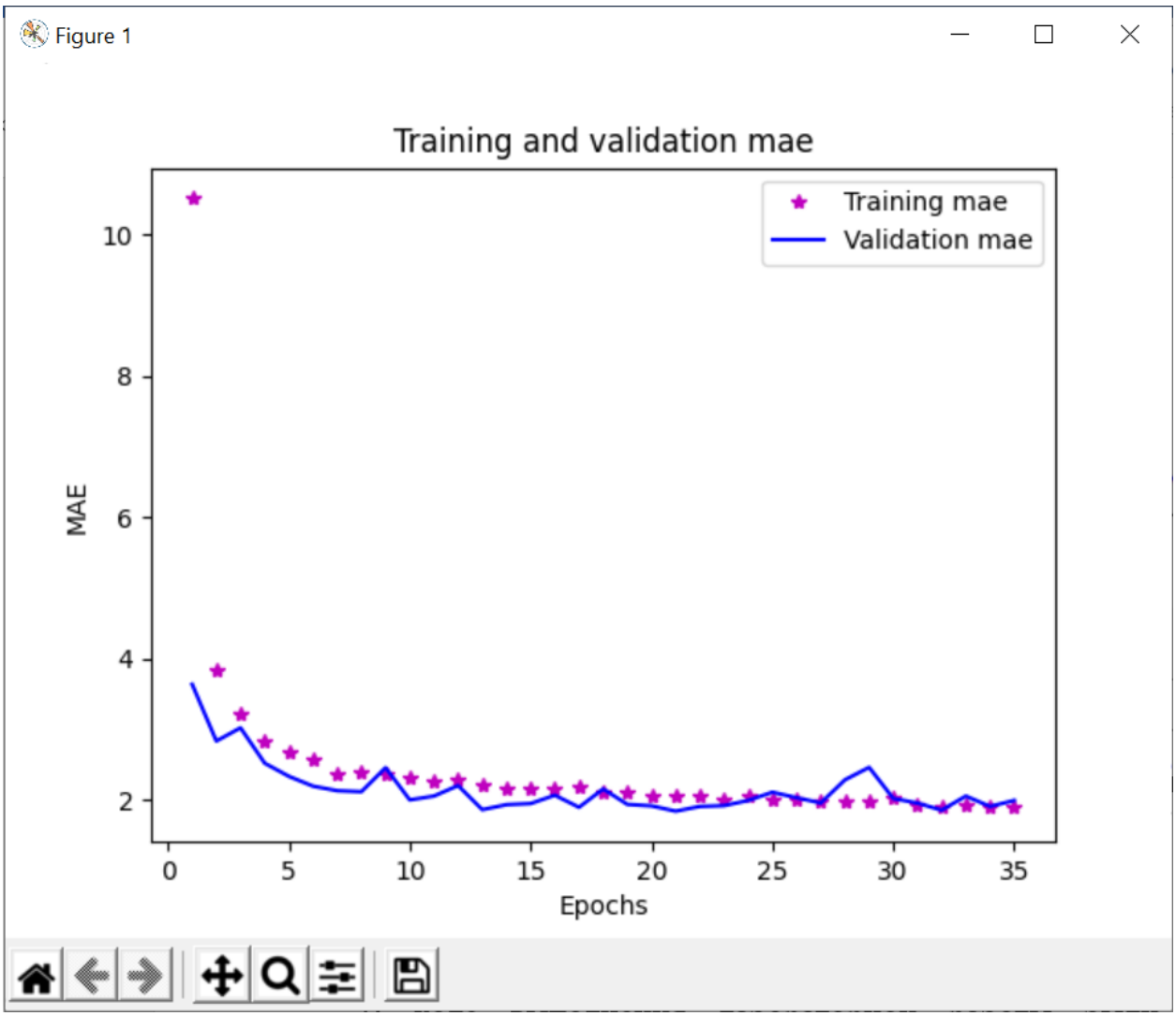

Рисунок 19 – графики оценки средней абсолютной ошибки первого блока.

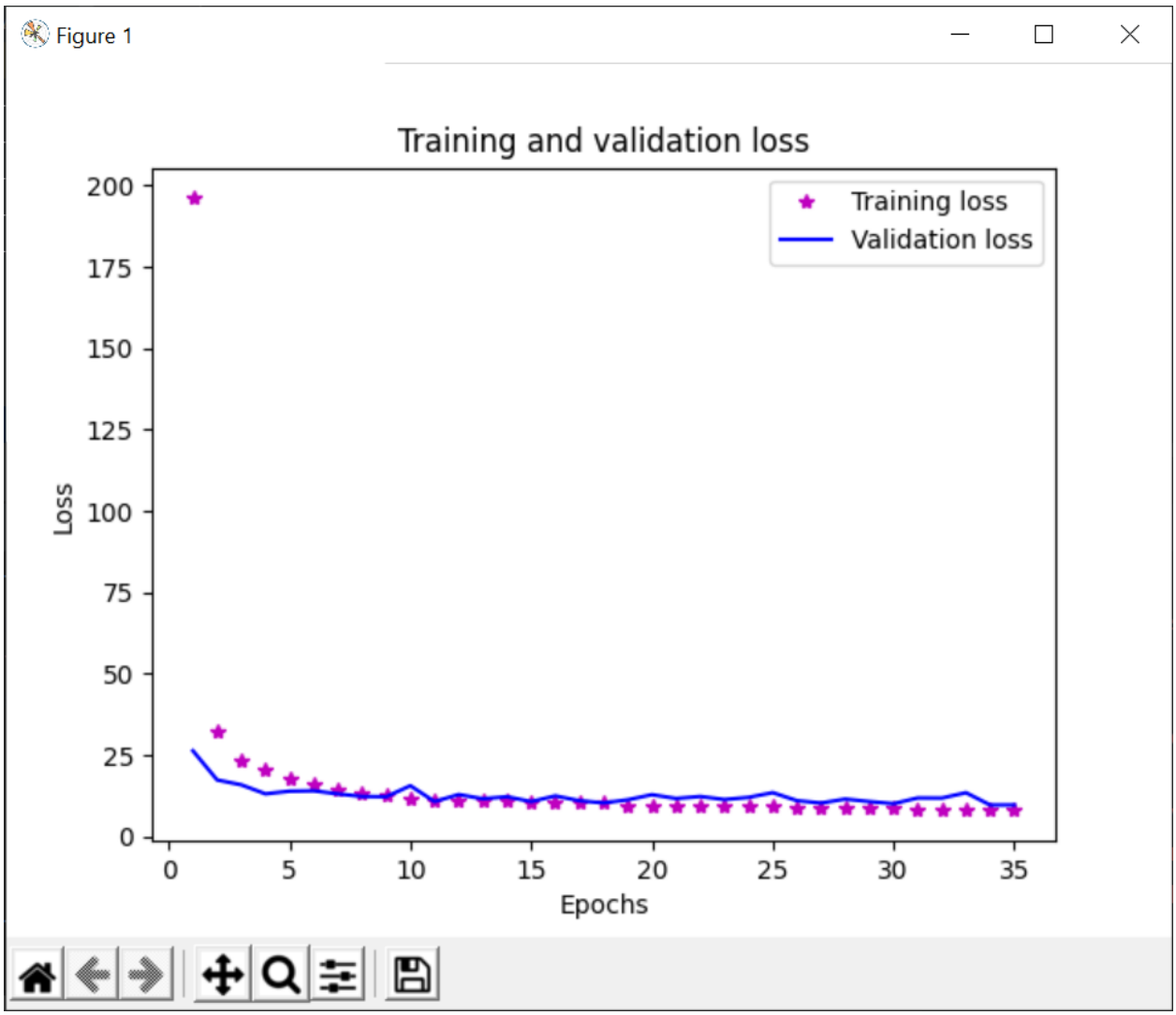

Рисунок 20 – графики потери сети на обучающих данных и данных, не участвовавших в обучении второго блока.

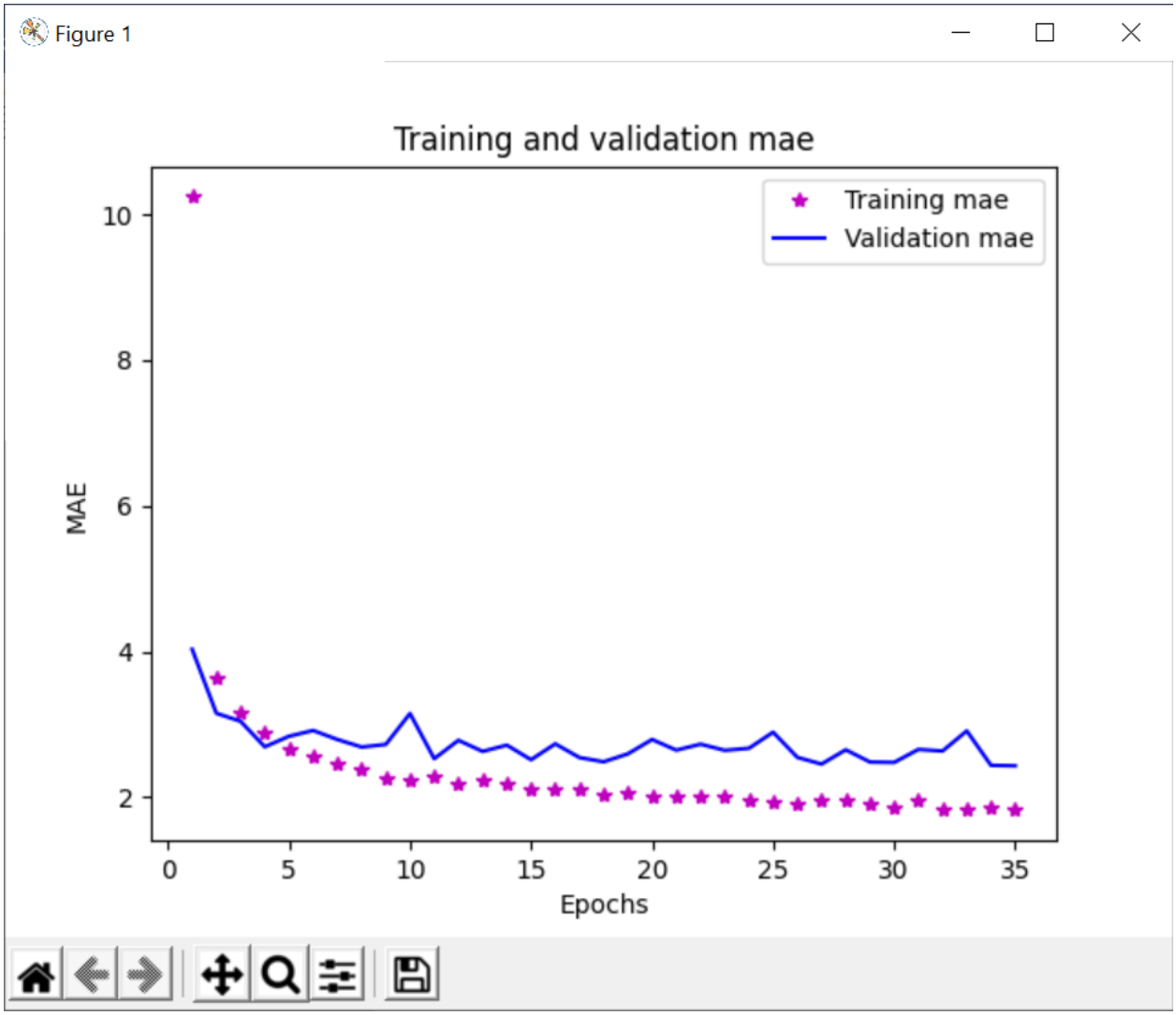

Рисунок 21 – графики оценки средней абсолютной ошибки второго блока.

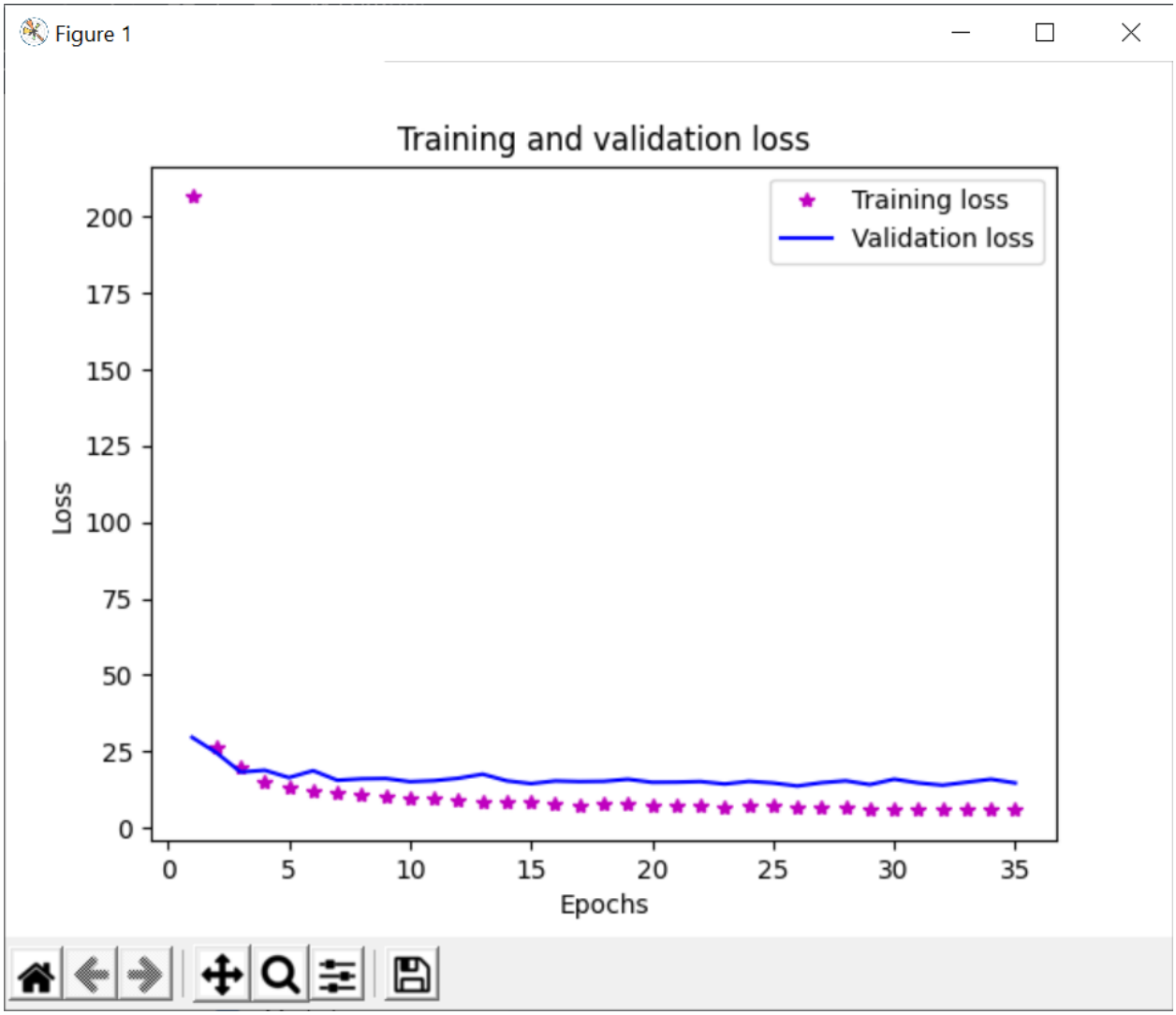

Рисунок 22 – графики потери сети на обучающих данных и данных, не участвовавших в обучении третьего блока.

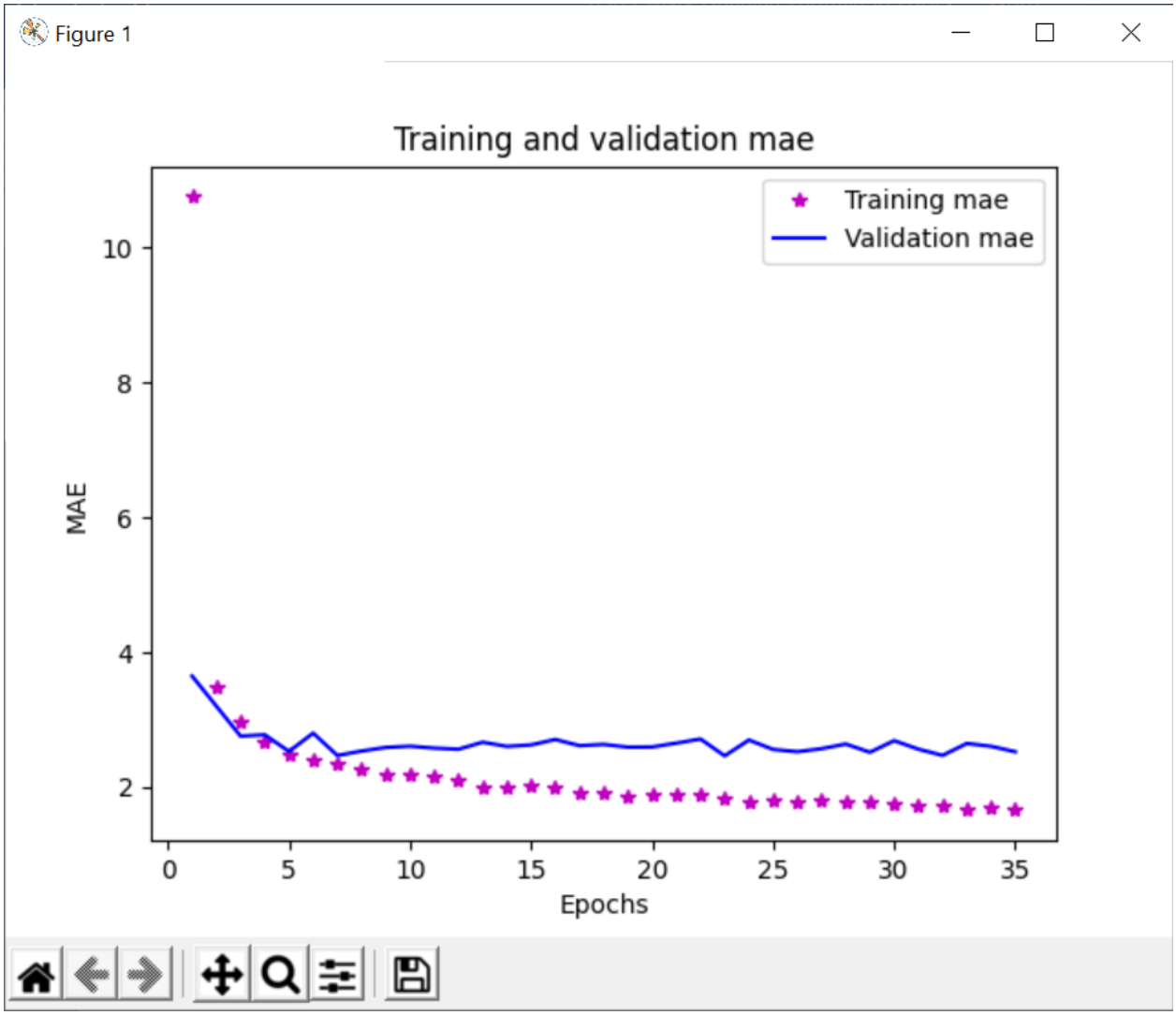

Рисунок 23 – графики оценки средней абсолютной ошибки третьего блока.

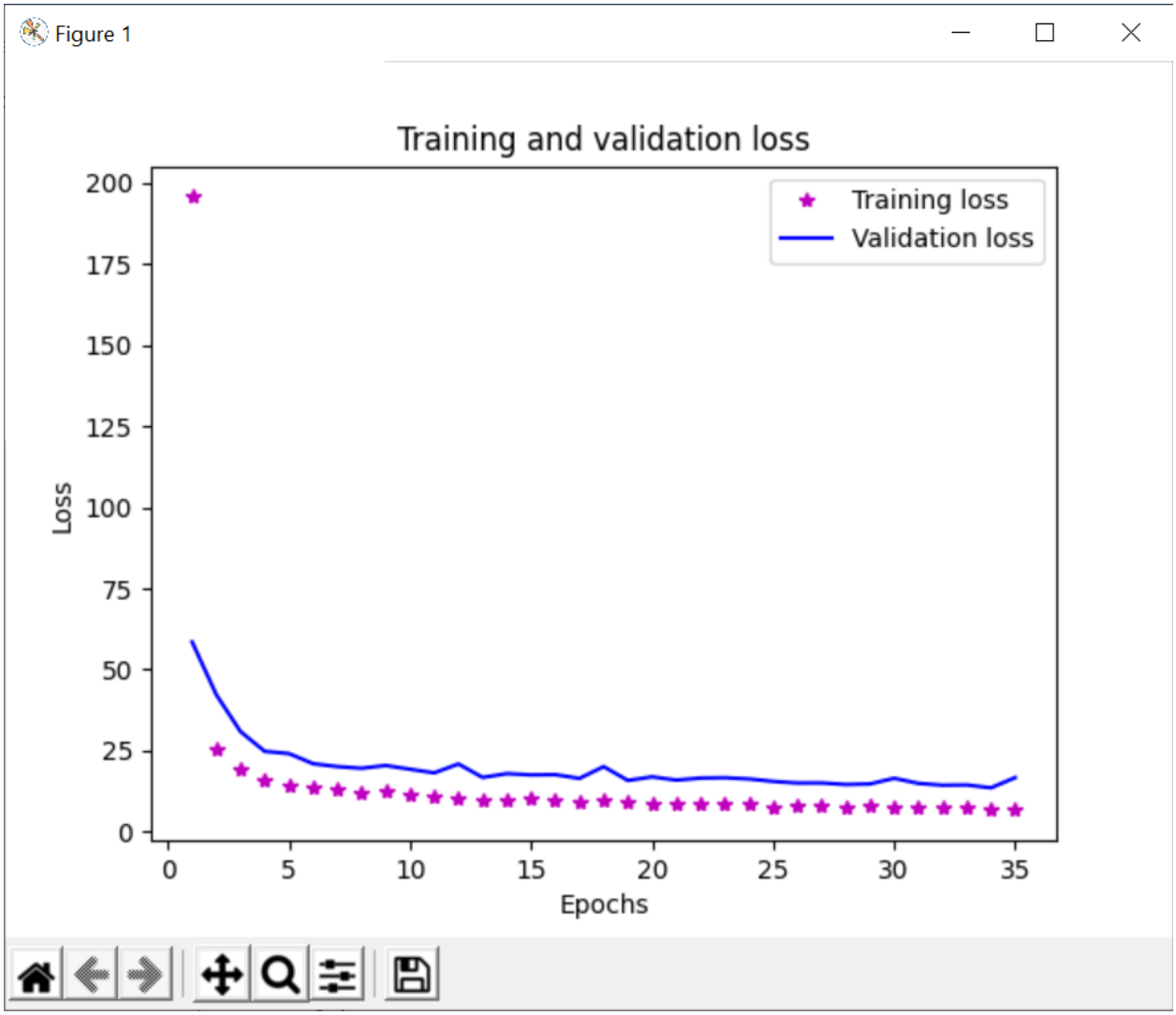

Рисунок 24 – графики потери сети на обучающих данных и данных, не участвовавших в обучении четвертого блока.

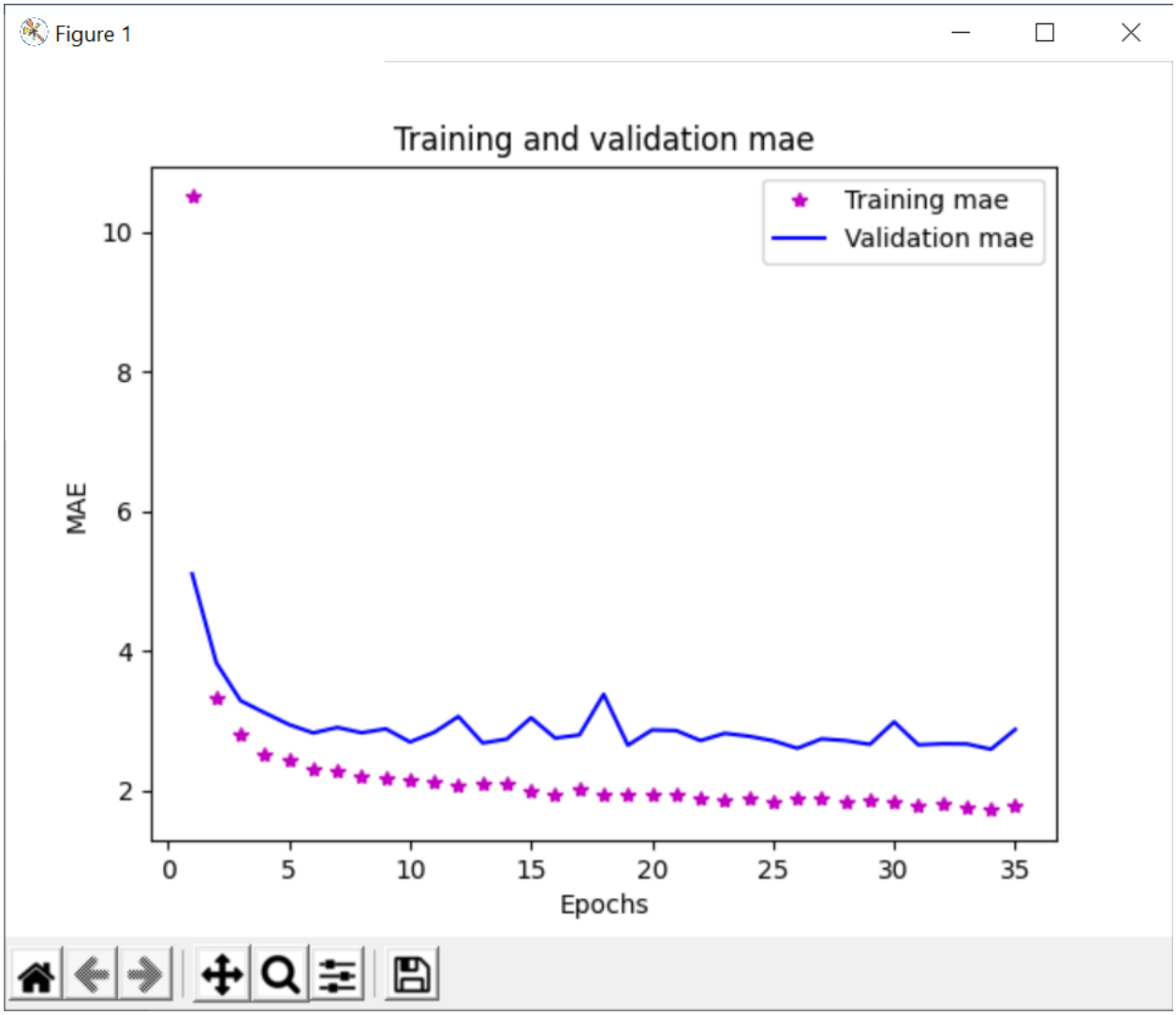

Рисунок 25 – графики оценки средней абсолютной ошибки третьего блока.

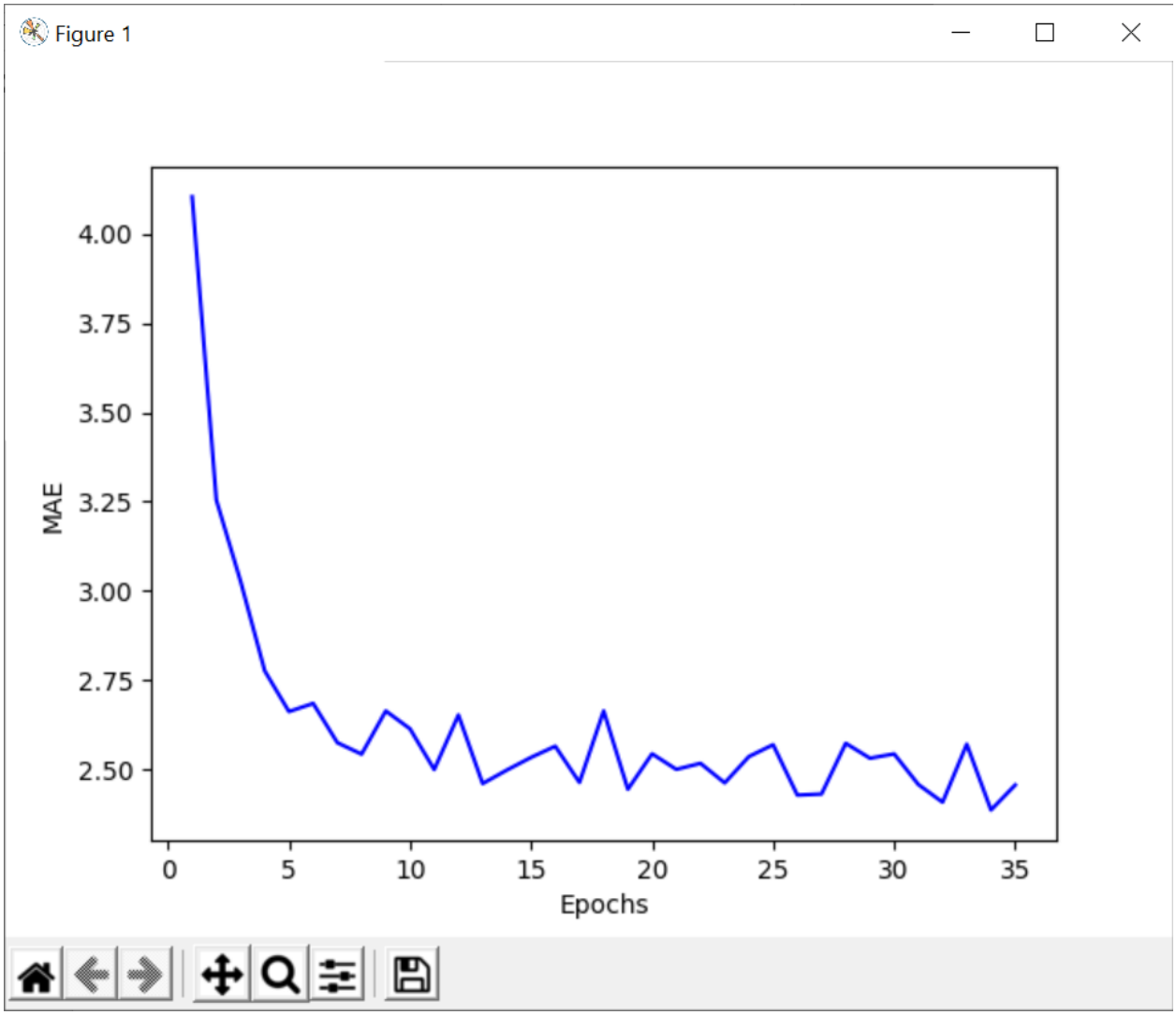

Рисунок 26 – график среднего значения оценки средней абсолютной ошибки

Таким образом, обучаемость модели при 35 эпохах улучшилась. Средняя mae после полного обучения по k блокам равна 2.4557216, что является лучшим результатом.

Далее была выбрана модель с параметрами обучения и перекрестной проверки:

```
k = 3
num_epochs = 35
```

Результат тестирования представлен на рис.27-33.

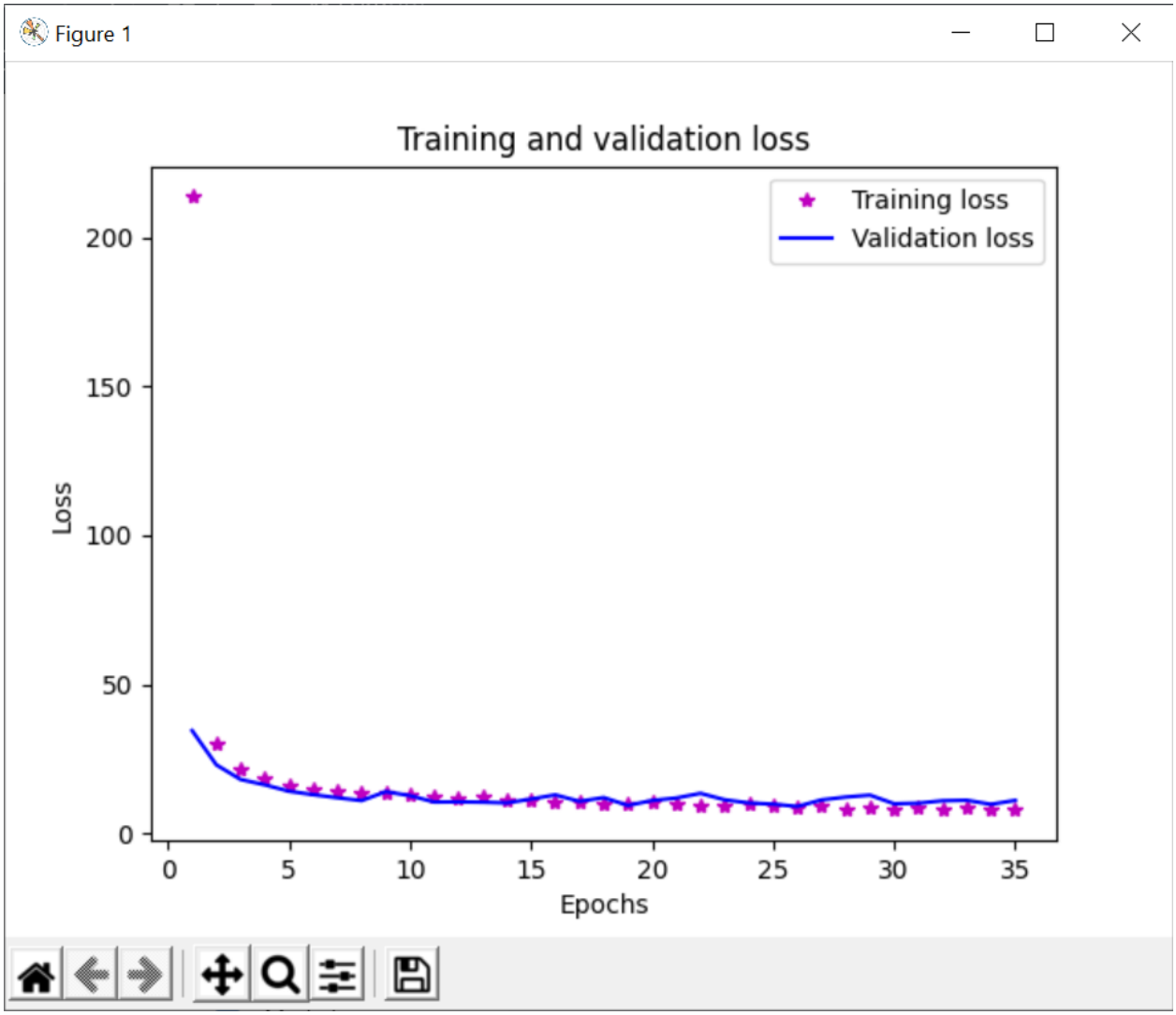

Рисунок 27 – графики потери сети на обучающих данных и данных, не участвовавших в обучении первого блока.

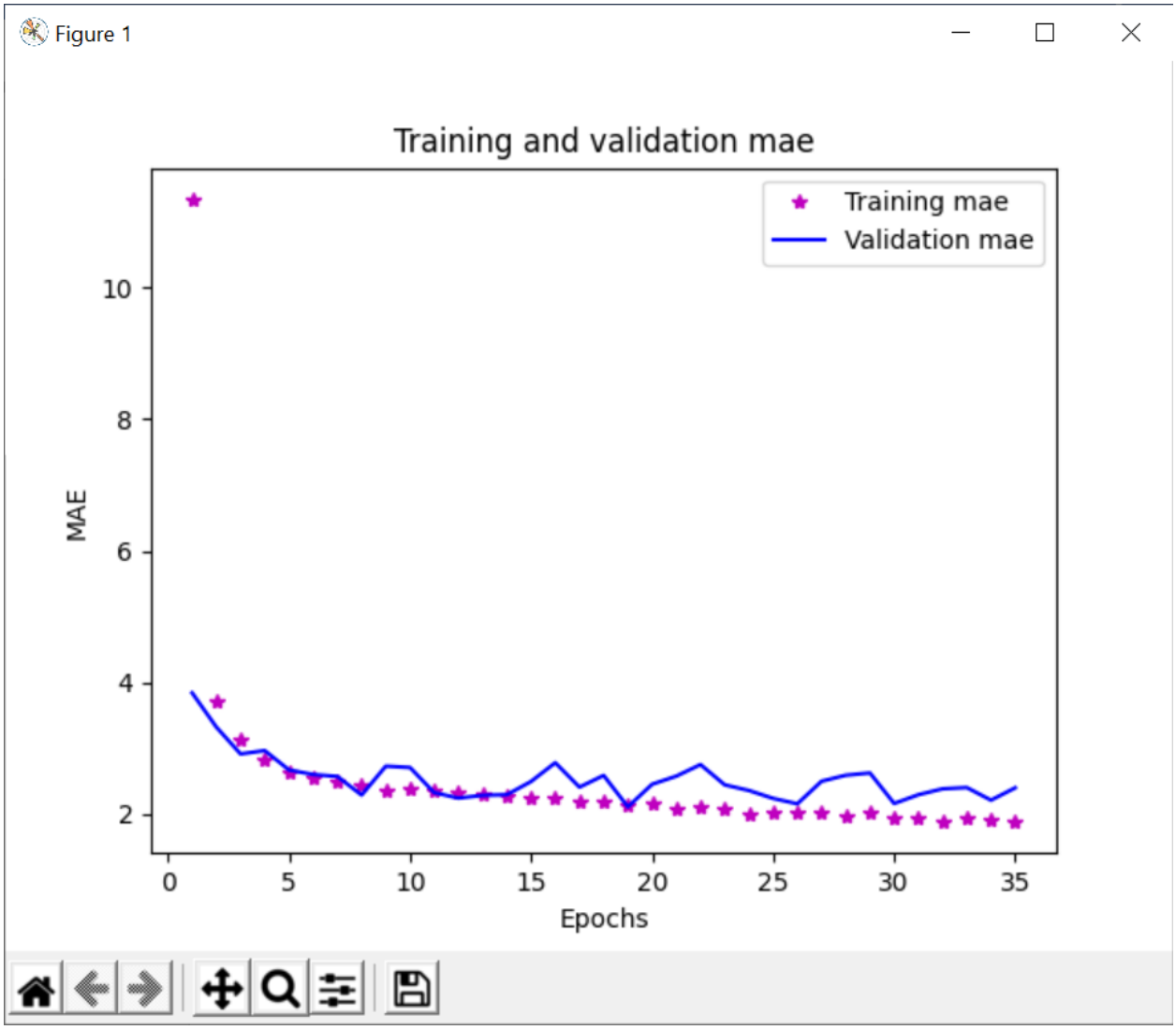

Рисунок 28 – графики оценки средней абсолютной ошибки первого блока.

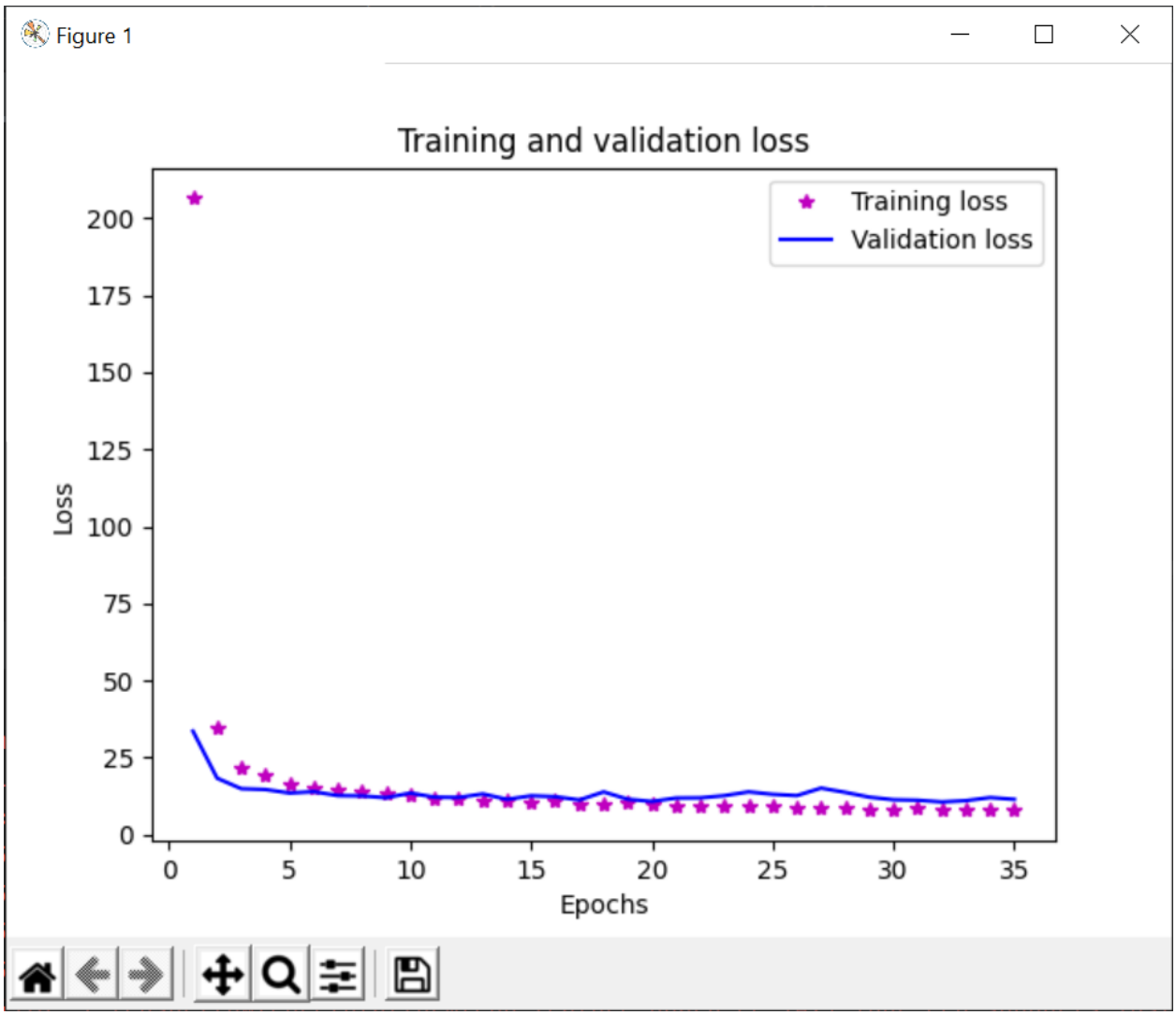

Рисунок 29 – графики потери сети на обучающих данных и данных, не участвовавших в обучении второго блока.

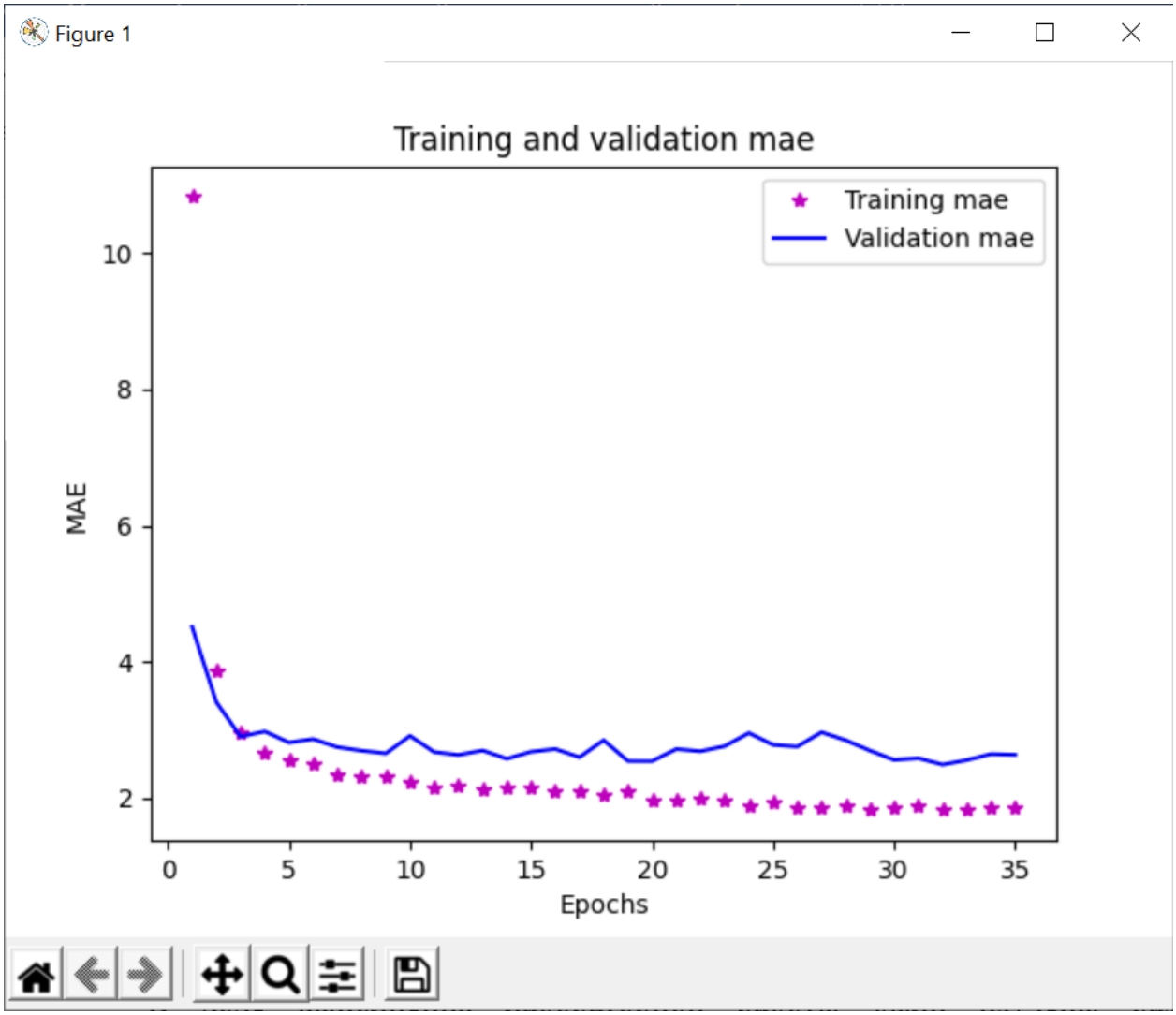

Рисунок 30 – графики оценки средней абсолютной ошибки второго блока.

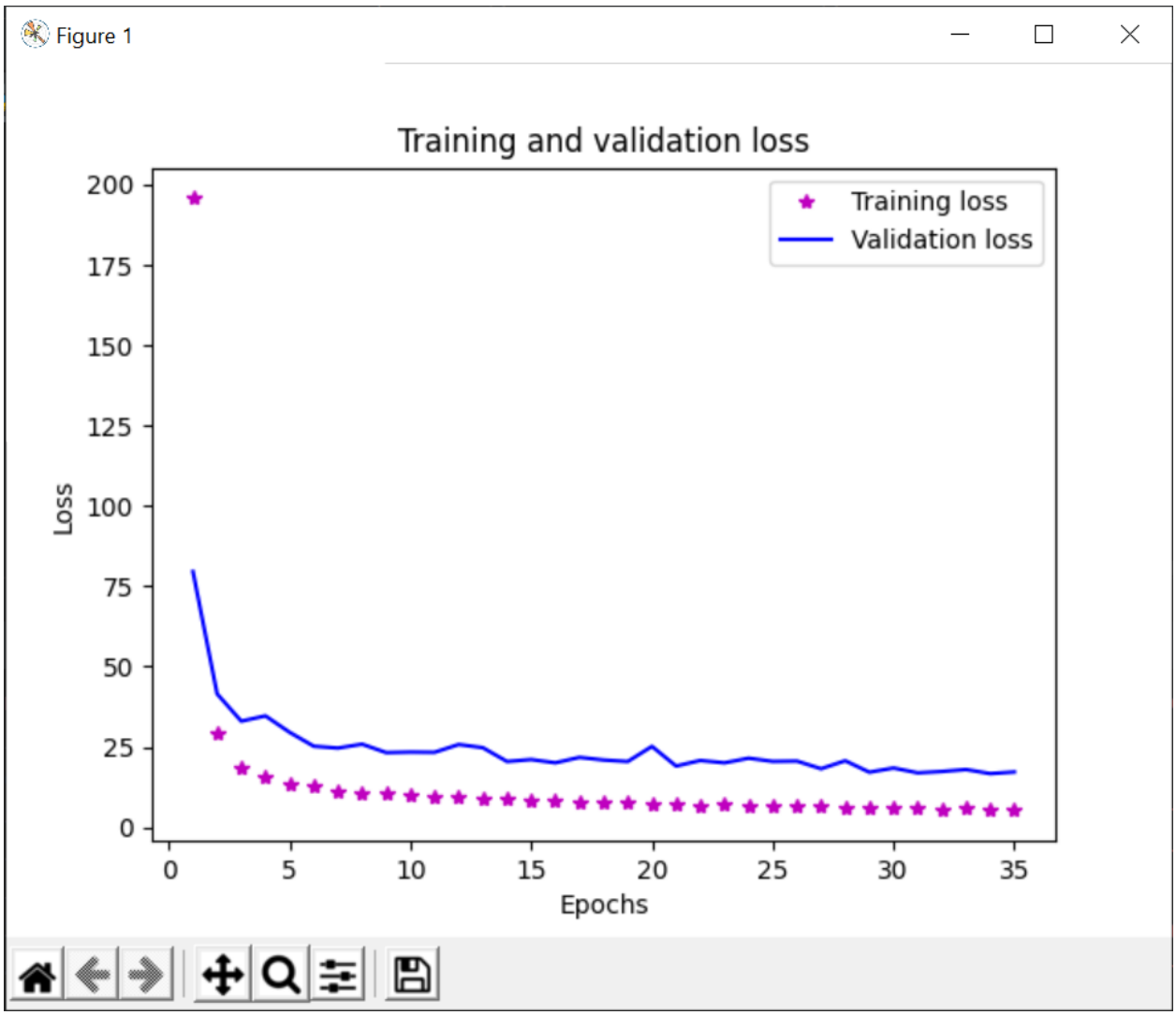

Рисунок 31 – графики потери сети на обучающих данных и данных, не участвовавших в обучении третьего блока.

Рисунок 32 – графики оценки средней абсолютной ошибки третьего блока.

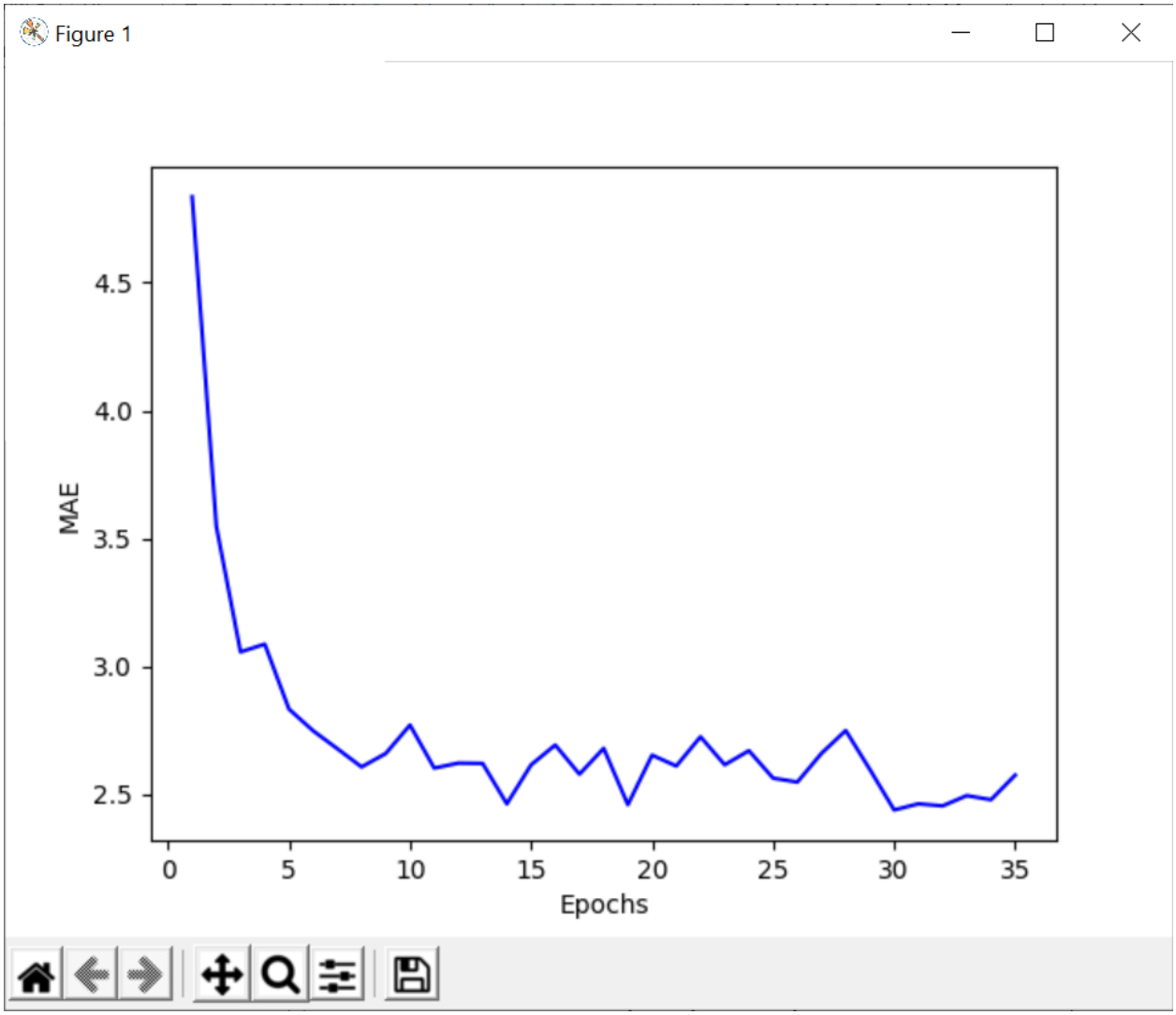

Рисунок 33 – график среднего значения оценки средней абсолютной ошибки

Средняя mae после полного обучения по k блокам равна 2.5744281.

Сравнив среднее mae после полного обучения с предыдущим результатом при k = 4 (2.4557216), видно, что значение ухудшилось. Значит, уменьшение числа блоков приводит к ухудшению эффективности обучаемости ИНС.

Далее была выбрана модель с параметрами обучения и перекрестной проверки:

```
k = 5
num_epochs = 35
```

Результат тестирования представлен на рис.34-44.

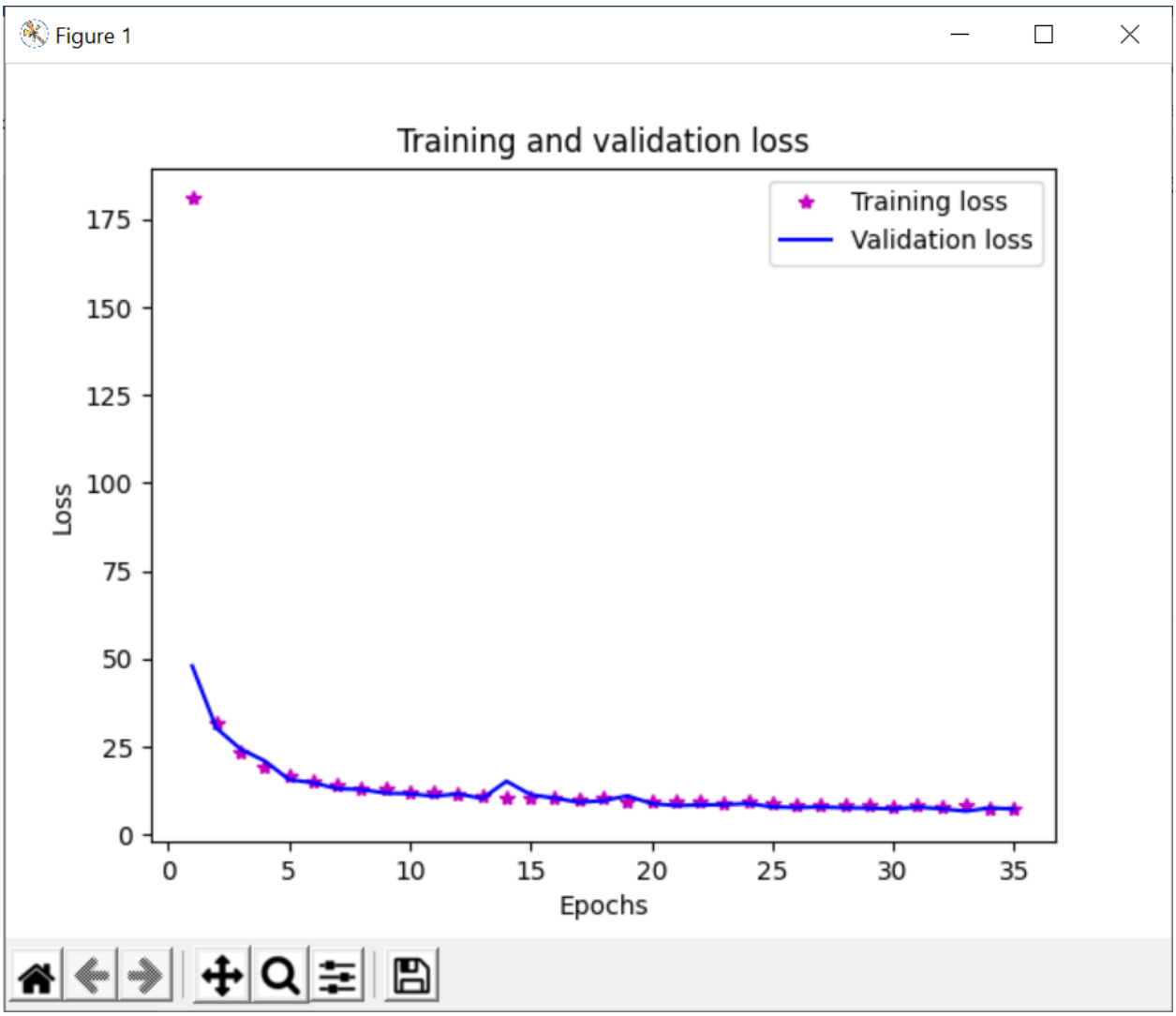

Рисунок 34 – графики потери сети на обучающих данных и данных, не участвовавших в обучении первого блока.

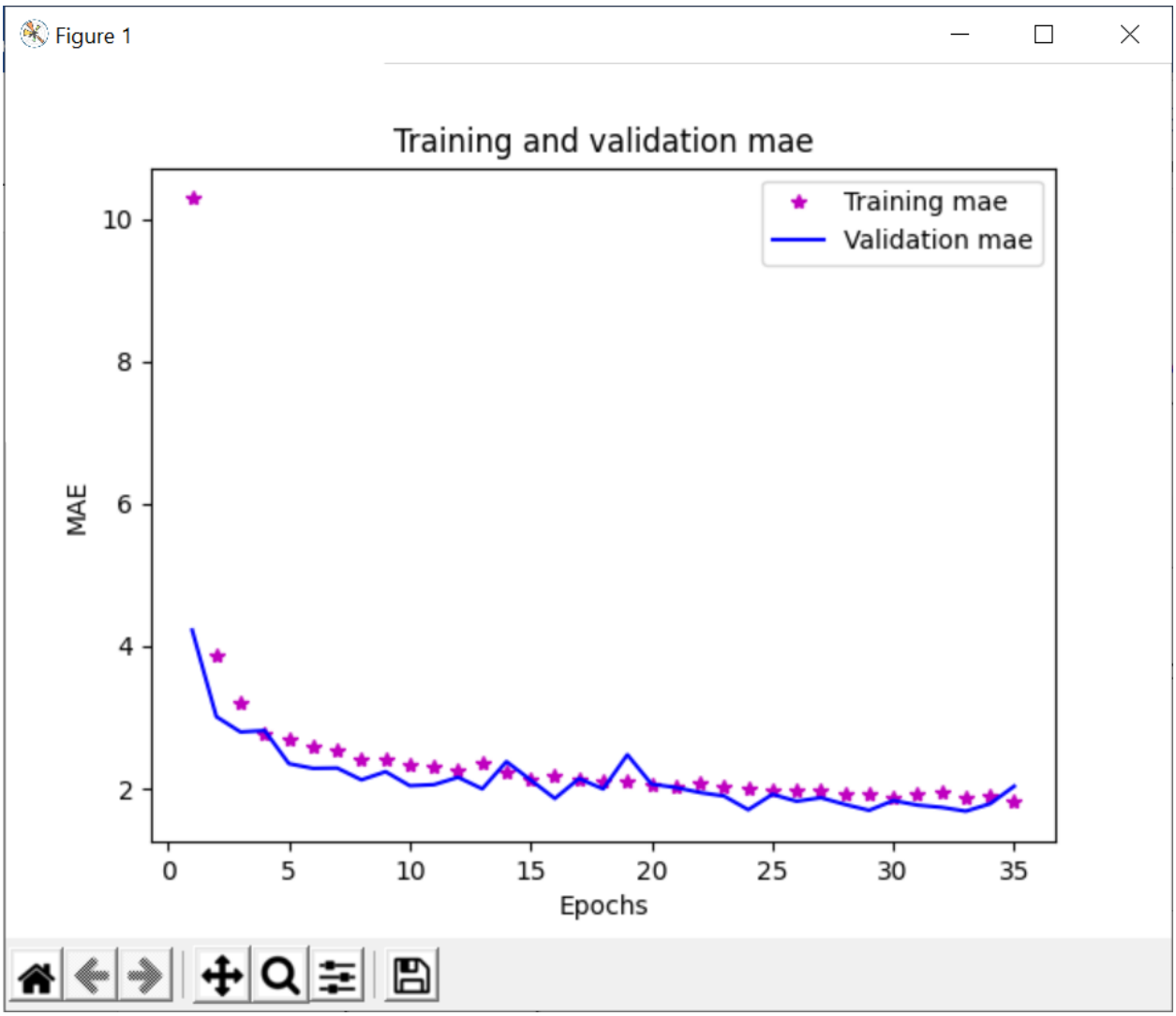

Рисунок 35 – графики оценки средней абсолютной ошибки первого блока.

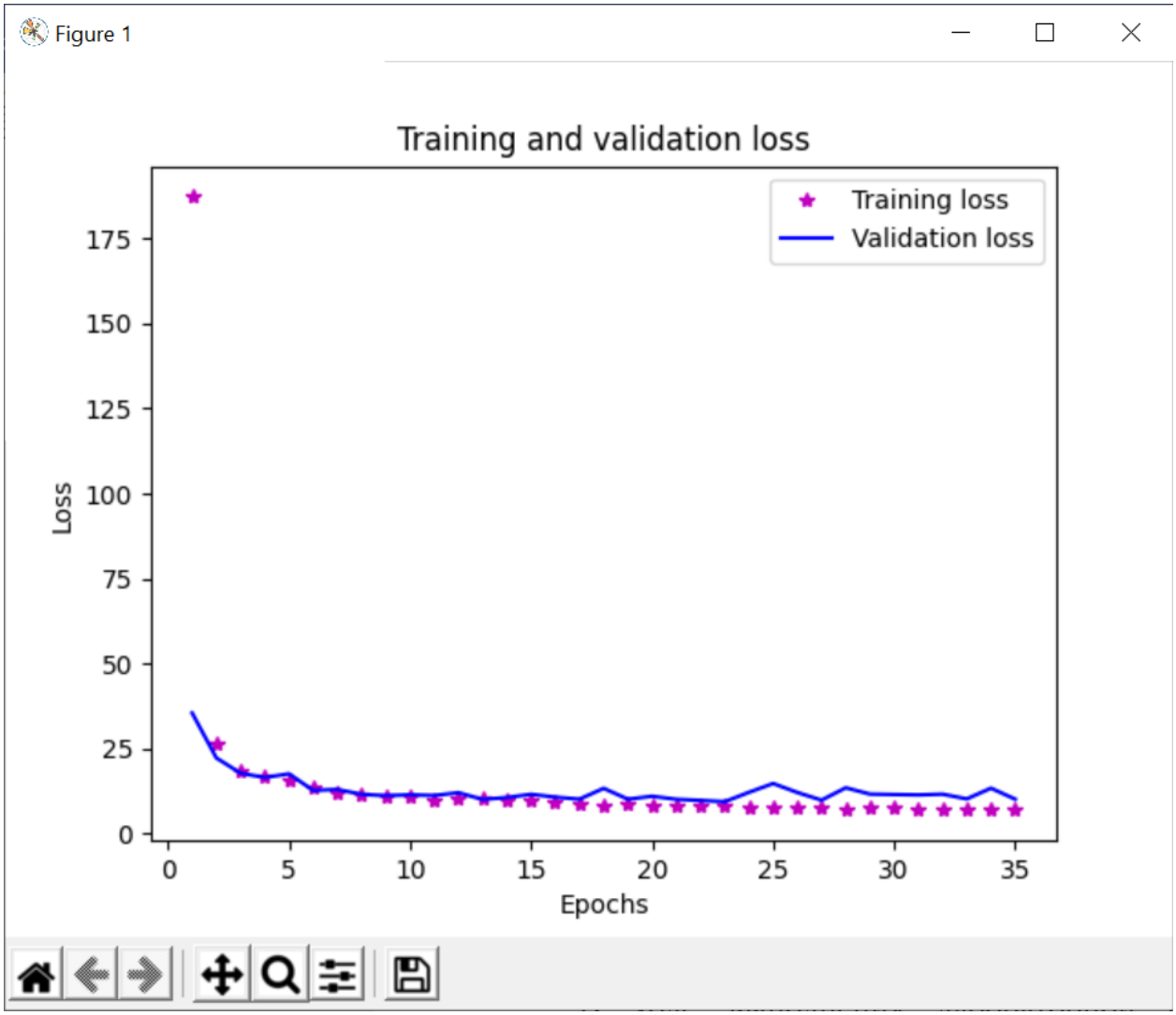

Рисунок 36 – графики потери сети на обучающих данных и данных, не участвовавших в обучении второго блока.

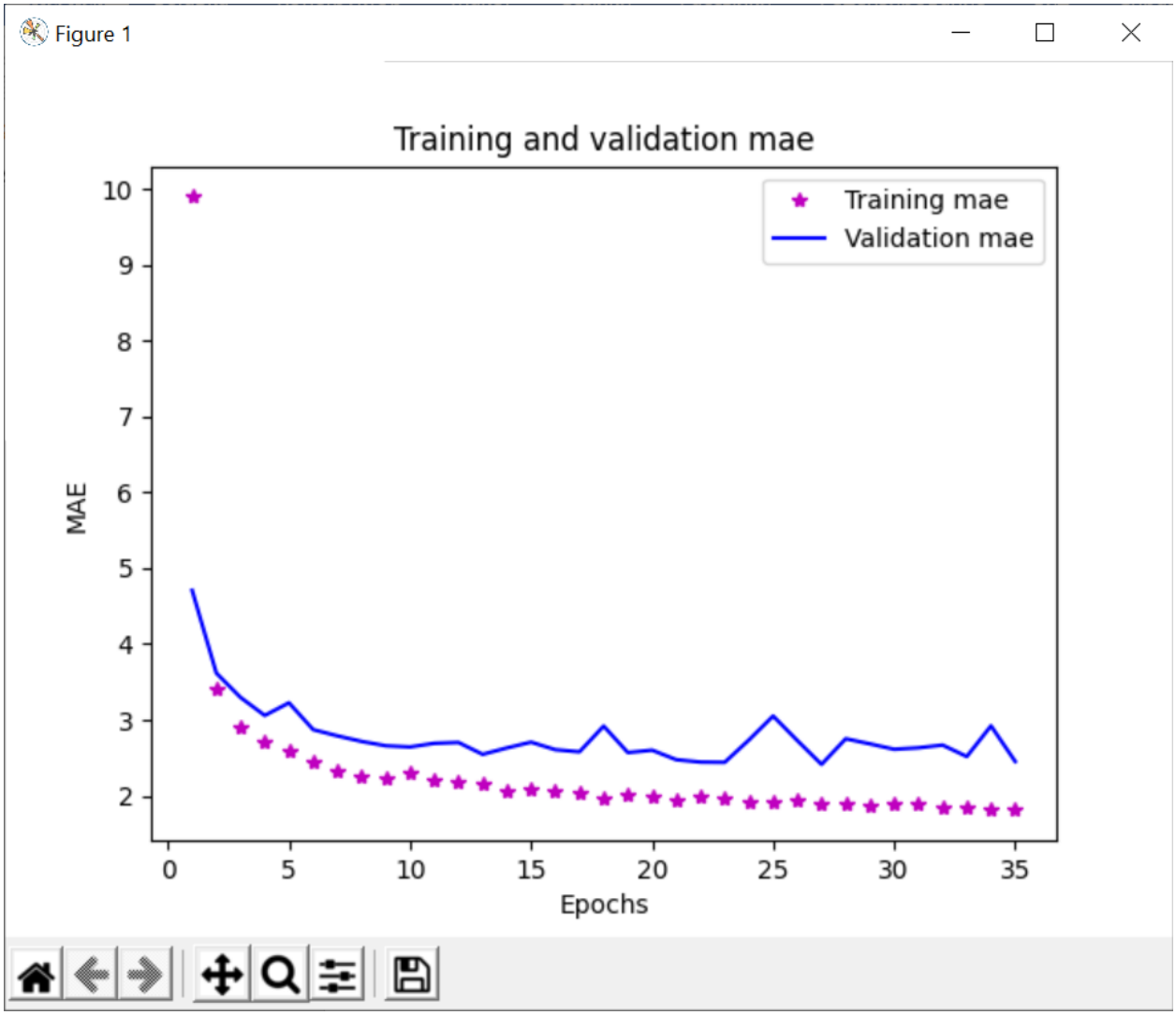

Рисунок 37 – графики оценки средней абсолютной ошибки второго блока.

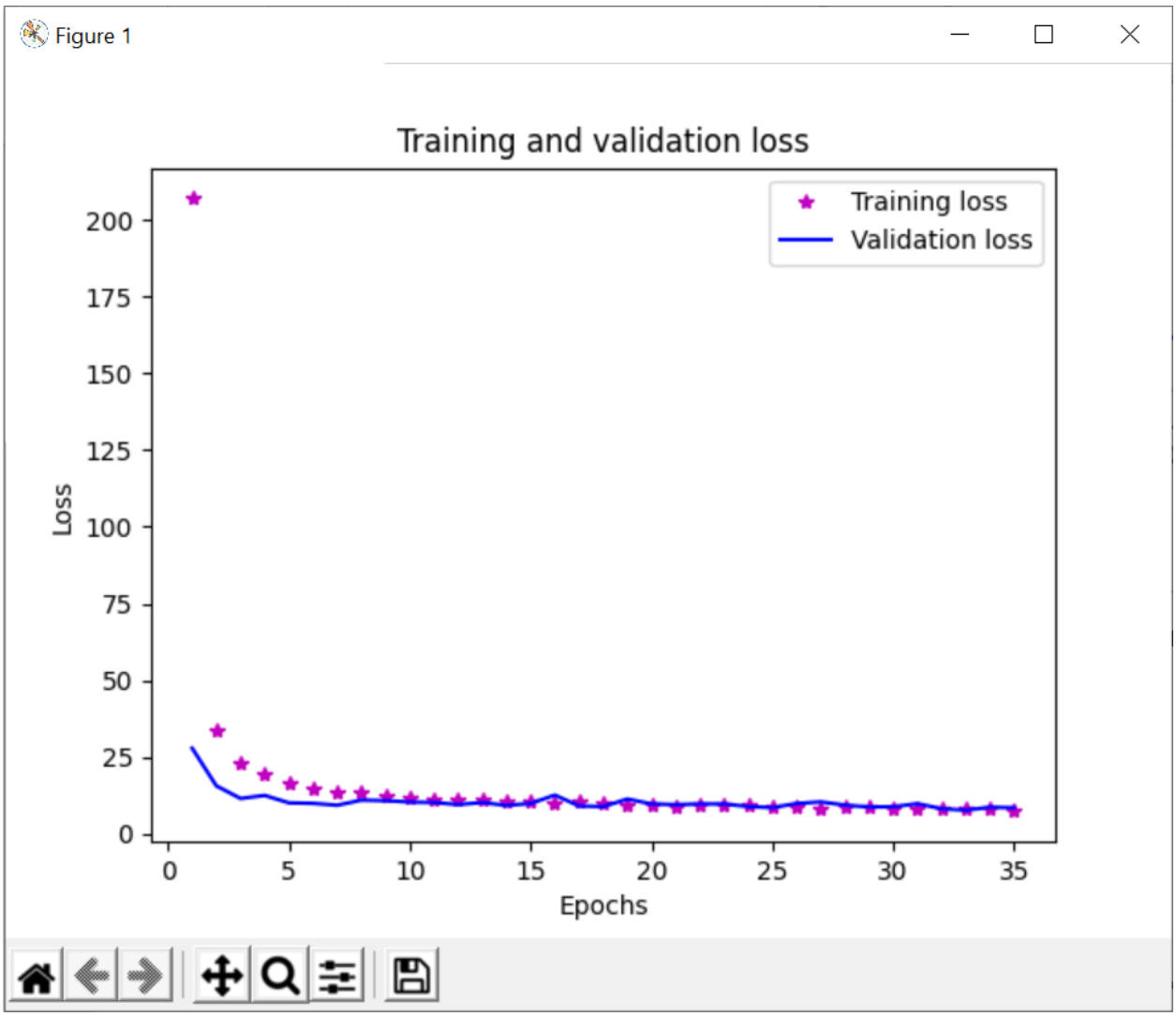

Рисунок 38 – графики потери сети на обучающих данных и данных, не участвовавших в обучении третьего блока.

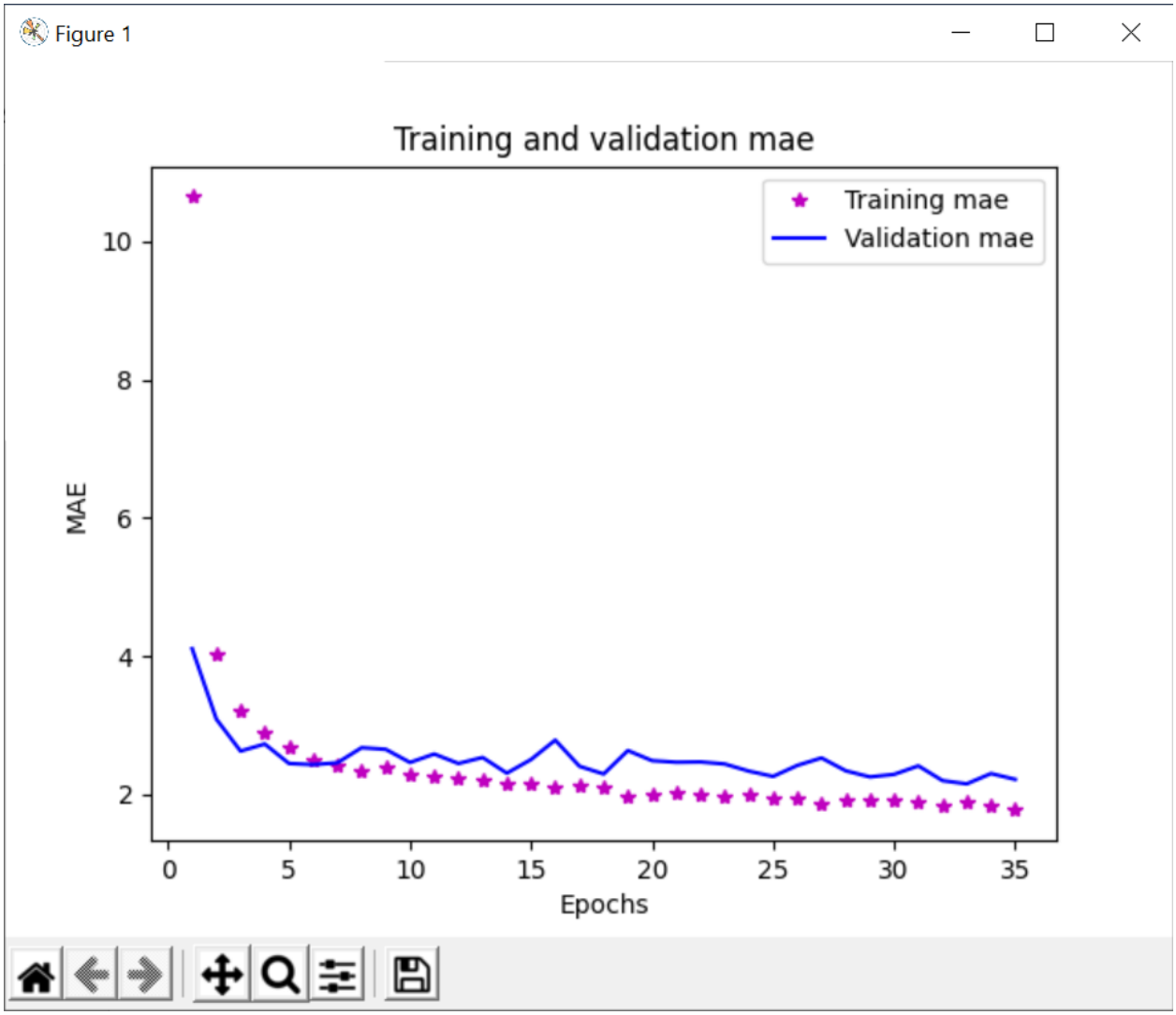

Рисунок 39 – графики оценки средней абсолютной ошибки третьего блока.

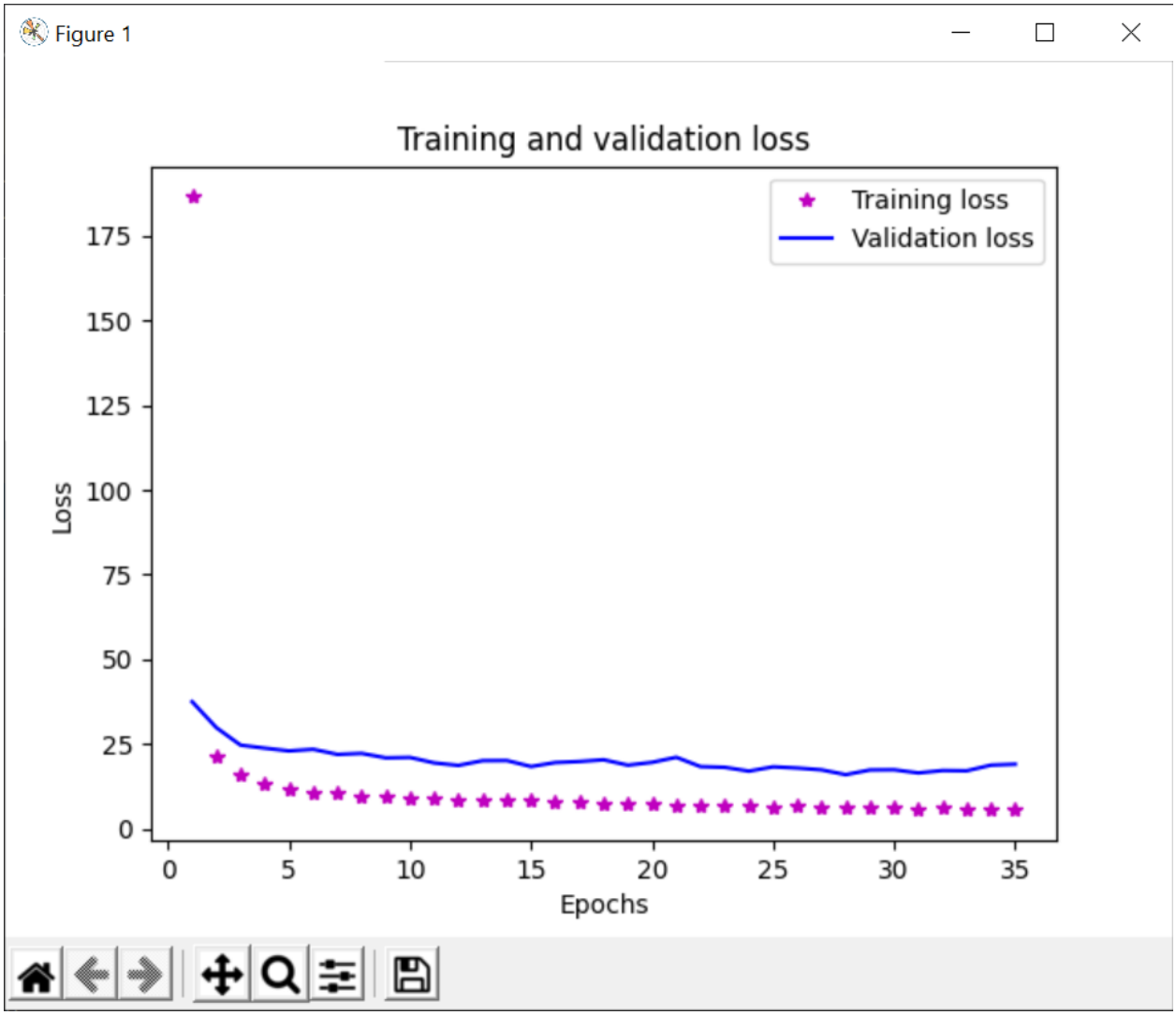

Рисунок 40 – графики потери сети на обучающих данных и данных, не участвовавших в обучении четвертого блока.

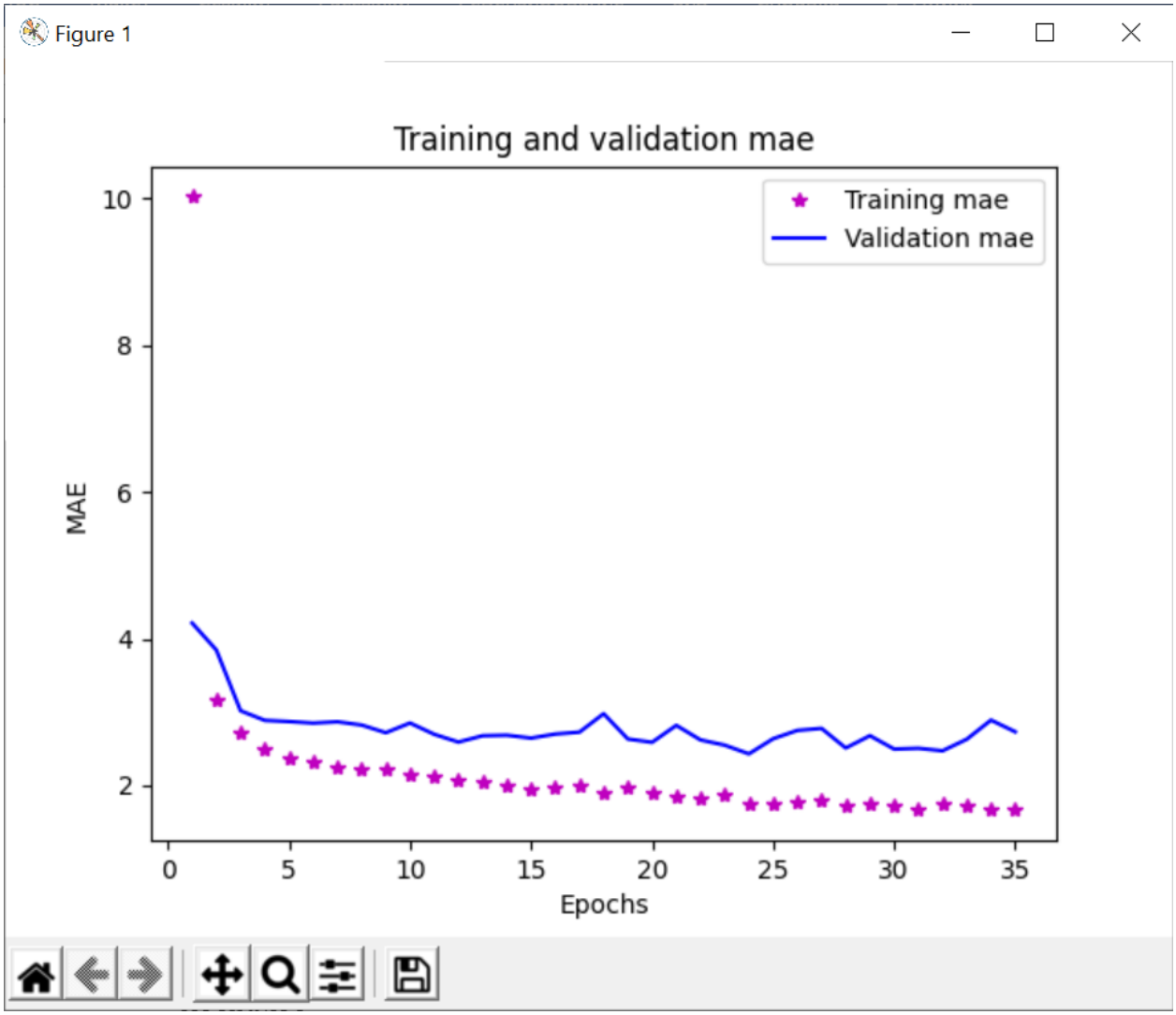

Рисунок 41 – графики оценки средней абсолютной ошибки четвертого блока.

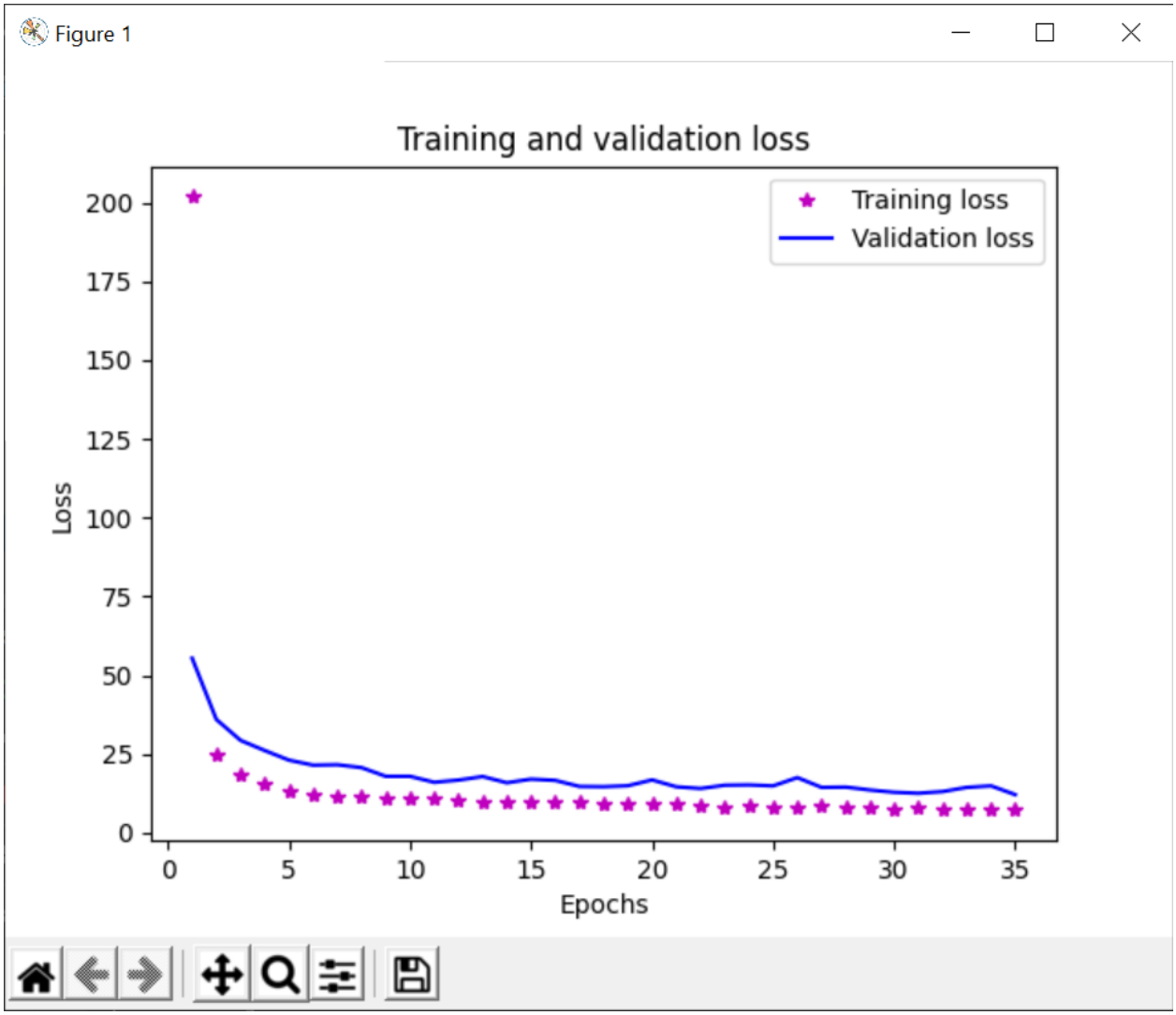

Рисунок 42 – графики потери сети на обучающих данных и данных, не участвовавших в обучении пятого блока.

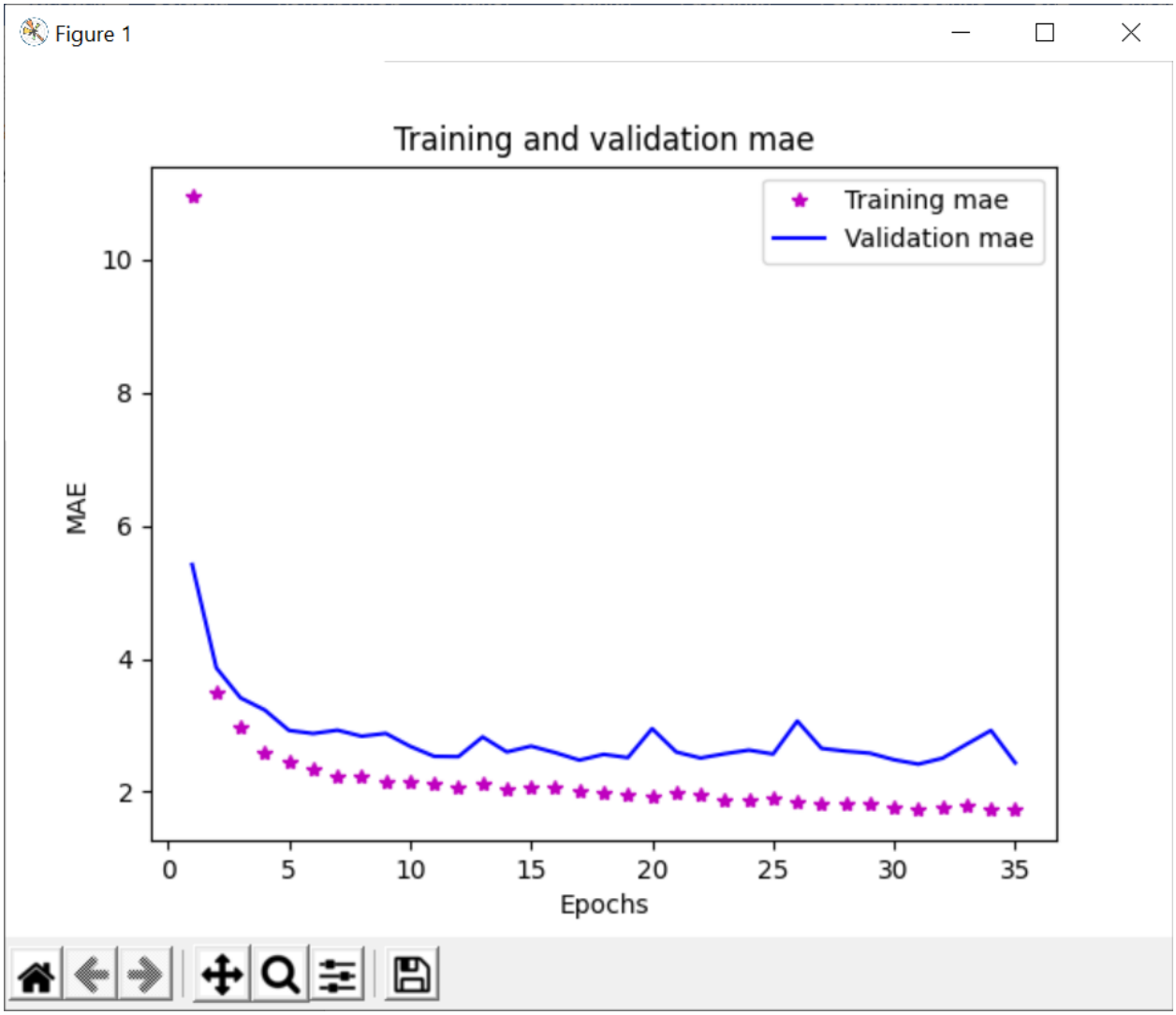

Рисунок 43 – графики оценки средней абсолютной ошибки пятого блока.

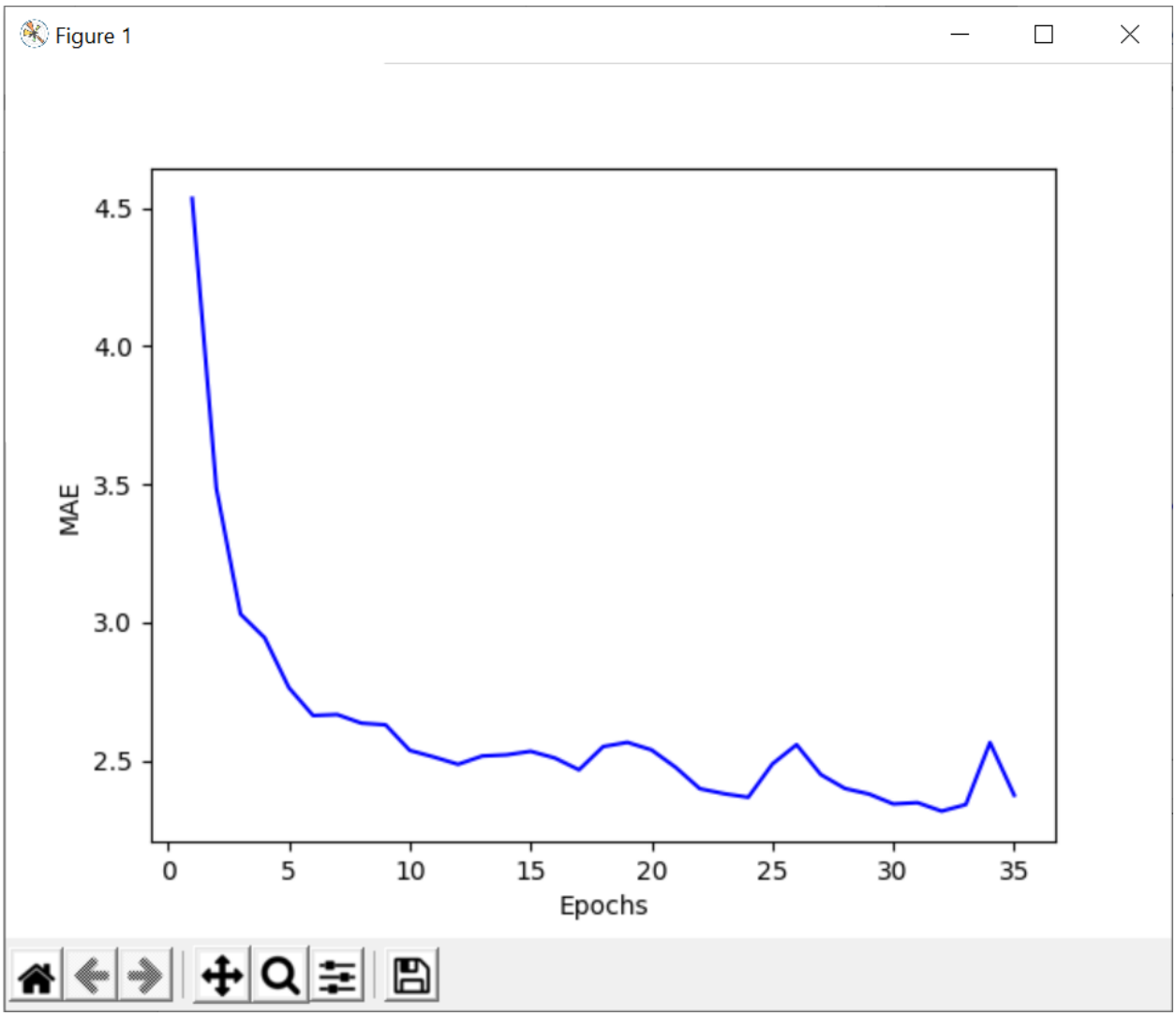

Рисунок 44 – график среднего значения оценки средней абсолютной ошибки

Средняя mae после полного обучения по k блокам равна 2.374253.

Сравнив среднее mae после полного обучения с предыдущим результатом при k = 4 (2.5744281), видно, что значение улучшилось. Значит, при k = 5 ИНС обучается наиболее эффективно.

Выводы.

В ходе выполнения лабораторной работы было реализовано предсказание медианной цены на дома в пригороде Бостона в середине 1970-х по таким данным, как уровень преступности, ставка местного имущественного налога и т. д, а также построены графики ошибок, mae в ходе обучения и усредненные графики по всем моделям.