УДК 004.032.26

DOI: 10.15827/0236-235X.127.368-376

Дата подачи статьи: 17.07.19

2019. Т. 32. № 3. С. 368–376

Прогноз состояния объекта на основе применения фильтра Калмана и глубоких нейронных сетей

А.Ю. Пучков¹, к.т.н., доцент, доцент, putchkov63@mail.ru

М.И. Дли¹, д.т.н., профессор, профессор, MiDli@mail.ru

Е.И. Лобанева¹, аспирант, lobaneva94@mail.ru

М.А. Василькова¹, студент, vasilkova_mariya00@mail.ru

¹ Смоленский филиал Национального исследовательского университета МЭИ, кафедра информационных технологий в экономике и управлении, г. Смоленск, 214013, Россия

В статье представлен алгоритм прогноза состояния объекта исходя из данных, поступающих в форме изображений от каких-либо источников, например, видеокамер, нацеленных на ответственные технологические зоны. В основе предлагаемого алгоритма лежит последовательное использование глубокой искусственной нейронной сети и фильтра Калмана. Нейронная сеть предназначена для уменьшения размерности входных данных (изображений), реализуя функцию энкодера, с выхода которого снимается вектор наблюдений за состоянием объекта. На основании этих наблюдений осуществляется оценка состояния объекта рекуррентным фильтром. Использование фильтра непосредственно для изображений привело бы к большой размерности задачи и практической невыполнимости из-за вычислительных трудностей.

Программа, реализующая предложенный алгоритм, разработана на языке Python 3.6 с использованием интегрированной среды Spyder из сборки Anaconda для операционной среды Linux. Выбор языка программирования обусловлен наличием для него мощных библиотек машинного обучения TensorFlow от компании Google, а также удобного фреймворка Keras для создания и работы с глубокими нейронными сетями.

Приведены результаты модельного эксперимента по использованию предложенного алгоритма для прогноза состояния объекта, который заключался в отнесении полученных наблюдений к тому или иному классу. В рамках эксперимента были сгенерированы наборы изображений, относящихся к различным классам и отличающихся своей текстурой. Для имитации шума на изображениях применялся построчный сдвиг пикселей по горизонтали. Сравнительный анализ результатов прогноза с применением фильтра Калмана и без него показал, что фильтрация позволяет снизить количество ложных классификаций.

Разработанный алгоритм может найти применение в системах поддержки принятия решений и автоматизированных системах управления технологическими процессами.

Ключевые слова: компьютерное зрение, фильтр Калмана, глубокие нейронные сети.

Методы и алгоритмы прогнозирования состояния различных объектов и систем в большинстве случаев используют данные, представленные в числовом виде. Однако современные средства контроля и наблюдения за объектами все больше применяют видеоданные, поэтому актуальна задача разработки и совершенствования методов видеоаналитики, в том числе алгоритмов, позволяющих делать прогноз на основе информации, представленной в форме изображений.

Распознавание визуальных образов считается классической задачей для интеллектуальных систем и оформилось в отдельное направление – компьютерное зрение. В современных системах компьютерного зрения, решающих задачи классификации объектов по их изобра-

жениям, широкое применение находят методы машинного обучения, основанные на использовании глубоких искусственных нейронных сетей, особенно на таких их разновидностях, как *сверточные нейронные сети* (convolutional neural network, CNN). Архитектура CNN была разработана еще в 1988 году, но стремительное распространение получила лишь с 2012 года, когда сеть AlexNet одержала победу в ежегодной олимпиаде по машинному зрению ImageNet [1]. Сеть улучшила качество распознавания примерно в полтора раза по сравнению с победителем предыдущего конкурса, решающим эту задачу с использованием векторов Фишера и SIFT [2]. Помимо новой архитектуры, этой победе способствовали и возросшие вычислительные возможности современных компьюте-

ров, в том числе персональных, а также разработка и применение библиотек машинного обучения (например, Theano, TensorFlow), программно-аппаратной архитектуры параллельных вычислений CUDA, позволившей существенно ускорить обучение нейронной сети за счет переноса этого процесса на видеокарты, содержащие тысячи микропроцессоров. Все это привлекло в сферу разработки систем компьютерного зрения большое количество исследователей по всему миру, что способствовало появлению широкого спектра практических применений CNN.

В своей структуре CNN содержат многократное чередование слоев свертки и подвыборки, а на выходе имеют полносвязный слой. Общее количество нейронов в сети может достигать миллионов, что сильно затрудняет математическое описание процесса формирования пространства признаков сетью и ее работы, поэтому глубокое обучение основывается почти исключительно на инженерных решениях, а идеи чаще доказываются экспериментальным путем, а не теоретическим [3].

Применение CNN позволяет не затрачивать усилия на выделение пространства признаков для классификации объектов, так как сеть сама формирует карты признаков при обучении и моделирует высокоуровневые абстракции в данных. Задачами исследователя становятся только приведение наборов исходных данных к виду, понятному сети, и организация ее тренировки. Их решение позволяет применять CNN в самых различных прикладных областях: в медицине [4], социальном инжиниринге [5], обработке текстов [6], распознавании жестов [7], автоматической идентификации транспортных средств [8], обнаружении трещин в бетоне [9] и других.

Для многих областей применения CNN интерес представляет не только текущий контроль состояния объекта, но и прогноз его развития, например, для высокоответственных технологических процессов, систем медицинского мониторинга. Поэтому актуальными научными задачами являются разработка и совершенствование алгоритмов прогноза состояния объектов с использованием глубоких нейронных сетей на основе обработки данных, представленных в форме изображений.

Материалы и методы

По применяемому формальному аппарату методы прогнозирования весьма разнообразны

(моделирование, экспертные оценки, балансовый метод, экстраполяция и другие). Далее будут рассматриваться экстраполяционные методы, алгоритмы которых записываются в форме $X_{i+1} = f(X_i, G_i)$, где X_{i+1} – прогнозируемый вектор состояния объекта; f – некоторая функция от вектора текущего состояния объекта X_i и коррекции состояния G_i .

Различия алгоритмов выражаются в разных подходах к вычислению G_i и видах функции $f(\cdot)$, что учитывает особенности постановки задачи прогнозирования, предметной области, объем априорной информации. Одним из таких методов прогнозирования при наличии шумов с известными статистическими характеристиками в измерениях является фильтр Калмана. Он позволяет получать оценку состояния динамической системы по зашумленным наблюдениям неполного вектора состояния, используя описание объекта в форме векторно-матричного дифференциального или разностного уравнения [10]. Если данные об объекте представлены в виде изображений, для применения фильтра необходим их препроцессинг, обеспечивающий формирование вектора состояния и снижение его размерности для осуществления дальнейшего прогноза.

Авторами предложен алгоритм прогнозирования состояния технологического объекта на основе обработки изображений, поступающих, например, с видеокамеры, нацеленной на заданную технологическую зону. В основе алгоритма лежит использование прогнозирующего калмановского алгоритма оценивания вектора состояния, который восстанавливается по вектору наблюдений, формируемому глубокой нейронной сверточной сетью. Разработанный алгоритм базируется на следующих предположениях:

- постановку задачи визуального сопровождения технологического объекта или процесса можно рассматривать как оценку состояния системы на основании последовательности зашумленных измерений (наблюдений), применяемую в теории управления [11];
- изображение технологической зоны монотонно изменяется в течение всего времени наблюдения, что исключает появление резких модификаций ее изображения;
- интервал временной дискретизации Δt , через который поступают изображения, остается постоянным в течение всего периода наблюдений.

Кодирование изображений в компьютере предполагает отображение их в виде много-

мерных массивов. Например, изображение модели RGB содержит три канала (по количеству цветов), поэтому будет представлено многомерным массивом (тензором), имеющим форму $3 \times M_x \times M_y$, где M_x – количество пикселей по оси ОХ; M_y – количество пикселей по оси ОУ. Даже при низком разрешении 640×480 это приводит к тому, что вектор состояния будет содержать $3 \times 640 \times 480 = 921\,600$ элементов и расчет коррекции по калмановскому алгоритму практически невозможен в силу такой большой размерности и необходимости решать матричные уравнения. Выходом из данной ситуации может быть предварительное снижение размерности вектора состояний. Для этого успешно применяются CNN в различных архитектурах, например, в сетях SPP-net [12].

В предлагаемом алгоритме изображение технологической зоны сначала обрабатывается CNN с целью понижения размерности входных данных (изображения). В результате формируется вектор наблюдения фиксированной длины, позволяющий применять калмановский алгоритм прогноза состояния технологического объекта. Предполагается, что наблюдения содержат каналные шумы, влияние которых на дальнейшую оценку состояния желательно уменьшить. CNN работает как энкодер: получает на входе данные (изображение) и выдает их в сжатом виде (кодирует), при этом информации в выходных данных достаточно для их последующего использования. Укрупненная структура предлагаемого алгоритма представлена на рисунке 1.

Применение фильтра Калмана в связке с CNN при анализе изображений уже описывалось в литературе, например в [13], где под компонентами вектора состояний понимались такие параметры, как координаты характерных точек обрамляющего прямоугольника, его площадь и соотношение сторон, скорость изменения положения центра прямоугольника [14, 15]. Формирование обрамляющих прямоугольников требует дополнительных вычислительных затрат и не всегда необходимо при обработке изображений, где интерес представляет текстура, а не обособленный объект, движение которого анализируется.

Предлагаемая на рисунке 1 структура обеспечивает обработку каждого цветового канала отдельной CNN, что позволяет для каждого из них использовать индивидуальный фильтр Калмана, подстраивая его параметры и исходные данные с учетом канальных особенностей. В структурно более простом случае можно ограничиться одной CNN сразу на все каналы и увеличить размерность получаемого вектора состояний, однако это приведет к вычислительным трудностям из-за необходимости находить обратные матрицы большой размерности при расчете фильтра.

Входными данными для разработанного прогнозирующего алгоритма является наблюдаемая последовательность изображений размером $M_x \times M_y$, которая попадает на вход энкодера. На его выходе присутствует полносвязный слой, формирующий для каждого изображения вектор наблюдения u , имеющий

форму $(n \times 1)$, где $n \ll M_x \times M_y$. В результате обработки I изображений получается многомерная матрица наблюдений (тензор) U , имеющая форму $(n, I, 3)$, где 3 – количество цветовых каналов.

Вектор состояния, подлежащий восстановлению и прогнозированию, связан с вектором наблюдения для i -го изображения: $x(i, r) = C(i, r) u(i, r) + N(i, r)$, где $C(i, r)$ – матрица измерений для i -го изображения и r -го цветового канала; $u(i, r) = U(:, i, r)$, символ «:» принят во многих языках программирования (например Python, MatLAB) для обозначения всего диапазона возможных значений индекса; $N(i, r)$ – шум измерений для i -го изображения и r -го цветового канала с матрицей интенсивностей $Q(i, r)$ размера $(n \times n)$.

Далее рассмотрим ситуацию, когда наблюдению постоянно доступны все элементы изображения, получаемого после работы энкодера, поэтому матрица измерений единичная и отображать ее в формулах не будем.

Предполагается, что поступающие с выхода энкодера векторы – это наблюдения состояния контролируемого по видеоданным объекта, которое удовлетворяет выражению (формирующему фильтру)

$$x(i+1, r) = A(i+1|i, r) x(i, r) + b(i, r) V(i, r), \quad (1)$$

где $A(i+1|i, r)$ – матрица сдвига размера $(n \times n)$, структура которой определяется требованием интерполяции изображений [16]; $V(i, r)$ – порождающий белый шум с матрицей интенсивностей $R(i, r)$ размера $(n \times n)$; $b(i, r)$ – матрица входа.

Дальнейшая последовательность шагов прогнозирования определяется калмановским рекуррентным алгоритмом:

- оценка состояния (предварительный априорный прогноз) перед поступлением нового изображения:

$$y(i+1|i, r) = A(i+1|i, r) y(i, r); \quad (2)$$

- расчет априорной матрицы ошибок:

$$P(i+1|i, r) = A(i+1|i, r) P(i, r) A^T(i+1|i, r) + b(i+1|i, r) R(i, r) b^T(i+1|i, r); \quad (3)$$

- определение оптимального коэффициента усиления фильтра с учетом ранее упоминавшегося допущения, что матрица измерений является единичной:

$$K(i+1, r) = P(i+1|i, r) [P(i+1|i, r) + Q(i+1, r)]^{-1}; \quad (4)$$

- вычисление апостериорной матрицы ошибок полученной оценки состояния:

$$P(i+1, r) = (E - K(i+1, r)) P(i+1|i, r), \quad (5)$$

где E – единичная матрица;

- уточнение начальной оценки состояния с учетом поступивших данных (изображений):

$$y(i+1) = y(i+1|i, r) + K(i+1, r) [u(i+1, r) - y(i+1|i, r)]; \quad (6)$$

- при поступлении новых изображений повторение шагов алгоритма.

Чтобы запустить рекуррентную процедуру (2)–(6), необходимо задать начальные значения $y(0) = M[x(0, r)]$, $P(0) = M[x(0, r) \cdot x^T(0, r)]$. Если математические ожидания $M[x(0, r) \cdot x^T(0, r)]$ и $M[x(0, r)]$ не точны, при увеличении i влияние неточности практически не сказывается на результатах вычислений.

Укрупненная последовательность этапов разработанного алгоритма прогнозирования состоит в следующем:

- обучение энкодера на эталонном наборе данных;
- определение параметров шума и начальных условий для фильтра Калмана;
- выполнение рекуррентной процедуры (2)–(6) для прогнозирования состояния контролируемого технологического объекта; точность выполняемого прогноза отражается в апостериорной матрице ошибок оценивания $P(i+1, r)$.

Результаты и обсуждение

С целью апробации алгоритма был проведен имитационный эксперимент для прогнозирования развития объекта, изменение которого задавалось изменением текстуры на изображении. Изменение текстуры, как предполагается, может служить классифицирующим признаком для определения состояния.

Для создания различных текстур было предложено использовать какую-либо функцию $F(x, y, \zeta)$, заданную на прямоугольной области изображения, управляя параметром ζ которой, можно было бы изменять текстуру. В качестве такой функции была выбрана проекция поверхности, получаемой в результате решения задачи триангуляции Делоне. Параметр ζ в эксперименте обеспечивал разбиение текстур на десять классов (рис. 2, верхний ряд) за счет формирования разного количества областей в прямоугольной области изображения. Шум на изображениях моделировался горизонтальными сдвигами пикселей относительно исходного положения (рис. 2, нижний ряд), причем параметры шума определяли количество пикселей, на которое осуществлялся сдвиг. Определение необходимых исходных данных для

Рис. 2. Изменение текстуры изображений в эксперименте

Fig. 2. Changing an image texture during the experiment

работы фильтра Калмана проведено экспериментально при обработке изображений, на которые не был наложен шум. В реальных условиях это можно сделать во время настройки контрольно-измерительного оборудования, используя, например, аппаратуру более высокого класса, чем та, которая будет применена в рабочем цикле.

Большое количество классов предполагает высокую чувствительность качества классификации к шумам, поэтому применение фильтра Калмана должно позволить снизить это влияние. Размерность вектора состояния при использовании фильтра Калмана определяет вычислительную сложность алгоритма и обычно не превышает трех или четырех элементов. Поэтому в данном эксперименте, чтобы ограничиться вектором состояний из четырех элементов для каждого канала, выходной слой энкодера содержал четыре нейрона.

Предложенный алгоритм прогнозирования состояния технологического объекта по изображениям был реализован на языке Python 3.6 в IDE Spyder из сборки Anaconda для операционной системы Linux. Энкодер построен с использованием специализированной нейросетевой библиотеки Keras, являющейся надстройкой над фреймворком тензорных вычислений TensorFlow. В данном эксперименте использовалась одна сеть, содержащая пять чередующихся слоев свертки и подвыборки, на выходе применен полносвязный слой с четырьмя выходами, что соответствует количеству элементов в векторе состояния.

Задание всего временного интервала наблюдения за технологическим объектом имитировалось получением 1 000 изображений, соответствующих линейному изменению параметра ζ от $\zeta_{\min}$ до $\zeta_{\max}$. Для обу-

чения энкодера применялась выборка из 50 000 изображений, разбитых на указанные классы и сгенерированных в специальной программе [17]. Вся выборка была разбита на две части: 80 % выделено для обучения, 20 % – для тестирования. Тренировка сети осуществлялась на ноутбуке Asus FX502VM с CPU IntelCore i7-7700HQ и GPU GeForce GTX 1060. Использование при обучении видеокарты обеспечило ускорение этого процесса более чем в десять раз по сравнению с обучением на центральном процессоре. Для контроля обучения CNN использовались метрика точности (`metrics=['accu-racy']`) и функция потерь, в качестве которой была выбрана категориальная энтропия (`loss='categorical_crossentropy'`). Графики зависимости изменения этих величин от номера эпохи обучения сети показаны на рисунке 3.

Текст программы функции на Python, реализующей оценку состояния, приведен на рисунке 4.

Проверка работоспособности предлагаемого алгоритма выполнялась для режима `offline`, при котором на вход принимается уже заполненный массив наблюдений U . Массив прогнозов оценок Y на вход функции подается заполненный нулями, кроме самого первого среза, который определяет начальные условия. Учитывая, что на вход алгоритма в рабочей выборке было подано 1 000 изображений и вектор

Рис. 3. Графики изменения точности и потерь при обучении сети

Fig. 3. Graphs of changes in accuracy and losses during network training

```
def F_Kalman_img (A,R,U,Y,P,K):
# проверка соответствия размерностей массивов
if A.shape==R.shape and R.shape==P.shape and P.shape==K.shape and U.shape==Y.shape:
    E=np.eye(len(A[1,:,0])) # единичная матрица
    for kr in range(len(A[0,0,:])): # цикл по количеству цветовых каналов
        i=0
        print(kr)
        while i<len(U[0,0,:])-1:
            # предварительный прогноз, @ – обозначение матричного умножения
            Y[:,i+1,kr]=A[:,i+1,kr]@Y[:,i,kr]

            # априорная матрица ошибок
            P[:,i+1,kr]=A[:,i+1,kr]@P[:,i,kr]@A[:,i+1,kr].T+b@R[:,i,kr]@b.T

            # расчет оптимального коэффициента усиления фильтра
            K[:,i+1,kr]=P[:,i+1,kr]@np.linalg.inv((P[:,i+1,kr]+Q[:,i,kr]))

            # вычисление апостериорной матрицы ошибок полученной оценки состояния
            P[:,i+1,kr]=(E-K[:,i+1,kr])@P[:,i+1,kr]

            # уточнение начальной оценки состояния с учетом поступивших данных (изображений)
            Y[:,i+1,kr]=Y[:,i+1,kr]+K[:,i+1,kr]@(U[:,i+1,kr]-Y[:,i+1,kr])
            i=i+1
        return (Y)
    else:
        print ('Выявлено несоответствие размеров массивов!')
        sys.exit(1)
```

Рис. 4. Программа функции на Python, реализующей оценку состояния

Fig. 4. Python function program implementing state evaluation

состояний для каждого канала содержал четыре элемента, формы (на языке Python этот метод называется shape) входных тензоров функции имели вид: A.shape= (4, 4, 1000, 3), R.shape= (4, 4, 1000, 3), U.shape= (4, 1000, 3),

Y.shape= (4, 1000, 3), P.shape= (4, 4, 1000, 3), K.shape= (4, 4, 1000, 3).

Для оценки качества прогноза ограничимся визуализацией его результатов (рис. 5). По оси абсцисс отложены номера изображений, упо-

Рис. 5. Результаты классификации изображений

Fig. 5. Image classification results

рядоченные по возрастанию номеров классов, к которым они принадлежат: каждые сто изображений относятся к одному классу, что отражает ступенчатая линия. Точками обозначены результаты прогнозирования принадлежности к классу без применения фильтра Калмана, а пунктирной линией – с применением фильтра. Визуальный анализ графиков позволяет сделать вывод о том, что наибольшие ошибки классификации (прогноза) состояния происходят в зоне переключения с одного класса на другой, однако применение фильтра позволяет снизить количество ложных классификаций.

Заключение

В результате проделанной работы был предложен алгоритм прогнозирования состояния технологического объекта на основе данных, представленных в виде изображений технологической зоны. В основе алгоритма заложено

применение глубокой нейронной сети, выполняющей функцию энкодера для понижения размерности данных, представленных в виде изображений, и формирования вектора наблюдений, на основе которого далее производится оценка состояния объекта с помощью фильтра Калмана.

Модельный эксперимент по применению предложенного алгоритма прогнозирования состояния технологического объекта на основе данных, представленных в форме изображений, показал, что применение фильтра Калмана позволяет повысить качество прогноза.

Разработанный алгоритм может найти применение в системах поддержки принятия решений и автоматизированных системах управления технологическими процессами для обеспечения более полного использования доступной информации за счет включения в обработку данных, представленных в форме изображений.

Исследование выполнено при финансовой поддержке РФФИ, проект № 19-01-00425.

Литература

1. Krizhevsky A., Sutskever I., Hinton G. Imagenet classification with deep convolutional neural networks. Proc. NIPS, 2012, pp. 1–9. DOI: 10.1145/3065386.
2. Sánchez J., Perronnin F. High-dimensional signature compression for largescale image classification. Proc. CVPR, 2011, pp. 1665–1672.
3. Шолле Ф. Глубокое обучение на Python. СПб: Питер, 2018. 400 с.
4. Rohit S., Chakravarthy S. BMC Neurosci. 2011. URL: <https://bmcneurosci.biomedcentral.com/article/10.1186/1471-2202-12-S1-P35> (дата обращения: 15.04.2019). DOI: 10.1186/1471-2202-12-S1-P35.
5. Severyn A., Moschitti A. Twitter sentiment analysis with deep convolutional neural networks. Proc. 38th Intern. ACM SIGIR Conf., Santiago, 2015, pp. 959–962. DOI: 10.1145/2766462.2767830.
6. Wang P., Xu J., Xu B., Liu C., Zhang H., Wang F. Semantic clustering and convolutional neural network for short text categorization. Proc. ACL-IJCNLP, Beijing, 2015, vol. 2, pp. 352–357.
7. Ahlawat S., Batra V., Banerjee S., Saha J., Garg A.K. Hand gesture recognition using convolutional neural network. Proc. ICICC, 2019, vol. 56, pp. 179–186. DOI: 10.1007/978-981-13-2354-6_20.
8. Xiang L., Zhao D., Yu K. Automatic vehicle identification in coating production line based on computer vision. Proc. Intern. Conf. CSET, 2016, pp. 260–267. DOI: 10.1142/9789814651011_0038.
9. Cha Y.J., Choi W. Vision-based concrete crack detection using a convolutional neural network. Proc. CPSEMS, 2017, vol. 2, pp. 71–73.
10. Балакришнан А.В. Теория фильтрации Калмана. М.: Мир, 1988. 168 с.
11. Sonka M., Hlavac V., Boyle R. Image Processing, Analysis and Machine Vision. Thomson Publ., 2008, 866 p.
12. He K., Zhang X. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. 2014. URL: <https://arxiv.org/abs/1406.4729> (дата обращения: 15.05.2019).
13. Salarpour A., Salarpour A., Fathi M. Vehicle tracking using Kalman filter and features. SIPIJ. 2011, no. 2, pp 1–8. DOI: 10.5121/sipij.2011.2201.
14. Bewley A., Ge Z. Simple online and realtime tracking. 2016. URL: <https://arxiv.org/abs/1602.00763> (дата обращения: 15.05.2019).
15. Wojke N., Bewley A., Paulus D. Simple online and realtime tracking with a deep association metric. 2017. URL: <https://arxiv.org/abs/1703.07402> (дата обращения: 15.05.2019).
16. Сирота А.А., Иванков А.Ю. Блочные алгоритмы обработки изображений на основе фильтра Калмана в задаче построения сверхразрешения // Компьютерная оптика. 2014. Т. 38. № 1. С. 118–126.

17. Василькова М.А., Пучков А.Ю. Подготовка наборов входных данных для обучения глубоких нейронных сетей // Информационные технологии, энергетика и экономика: сб. тр. XVI Междунар. науч.-технич. конф. 2019. Т. 1. С. 262–265.

Software & Systems
DOI: 10.15827/0236-235X.127.368-376

Received 17.07.19
2019, vol. 32, no. 3, pp. 368–376

Predicting an object state based on applying the Kalman filter and deep neural networks

A.Yu. Puchkov¹, Ph.D. (Engineering), Assistant Professor, Associate Professor, putchkov63@mail.ru

M.I. Dli¹, Dr.Sc. (Engineering), Professor, Professor, MiDli@mail.ru

E.I. Lobaneva¹, Postgraduate Student, lobaneva94@mail.ru

M.A. Vasilkova¹, Student, vasilkova_mariya00@mail.ru

¹ Smolensk Branch of the Moscow Power Engineering Institute, Department of Information Technologies in Economics and Management Smolensk, 214013, Russian Federation

Abstract. The paper presents an algorithm for predicting an object state based on data from different sources (for example, video cameras) coming in the form of images aimed at critical technological zones. The proposed algorithm is based on the consistent use of a deep artificial neural network and the Kalman filter. A neural network is designed to reduce the input data dimension (images) performing the function of an encoder, which gives of an observation vector of the object state on the output. Based on these observations, the object state is evaluated by a recurrent filter. Using the filter directly for images would lead to a large dimension of the problem; it would be impossible to perform it practically due to computational difficulties.

The program that implements the proposed algorithm was developed in Python 3.6 using the Spyder integrated environment from the Anaconda assembly for the Linux operating environment. The choice of a programming language is due to the availability of powerful libraries for machine learning TensorFlow from Google, as well as the convenient Keras framework for creating and working with deep neural networks.

The paper describes the results of a model experiment on using the proposed algorithm for predicting an object state, which consisted in attributing the obtained observations to a particular class. The experiment also involved generating sets of images belonging to different classes, differing in their texture. A line-by-line horizontal pixel shift simulated the noise in the images. The comparative analysis of the predicted results with and without using the Kalman filter has shown that filtering reduces the number of false classifications.

The developed algorithm might be used in decision support systems and automated process control systems.

Keywords: computer vision, Kalman filter, deep neural networks.

Acknowledgements. The reported study was funded by RFBR according to the research project no. 19-01-00425.

References

1. Krizhevsky A., Sutskever I., Hinton G. Imagenet classification with deep convolutional neural networks. *Proc. NIPS*. 2012, pp. 1–9. DOI: 10.1145/3065386.
2. Sánchez J., Perronnin F. High-dimensional signature compression for largescale image classification. *Proc. CVPR*. 2011, pp. 1665–1672.
3. Scholle F. *Deep learning in Python*. St. Petersburg, Piter Publ., 2018, 400 p.
4. Rohit S., Chakravarthy S. *BMC Neurosci*. 2011. Available at: <https://bmcn neurosci.biomedcentral.com/articles/10.1186/1471-2202-12-S1-P35> (accessed April 15, 2019). DOI: 10.1186/1471-2202-12-S1-P35.
5. Severyn A., Moschitti A. Twitter sentiment analysis with deep convolutional neural networks. *Proc. 38th Intern. ACM SIGIR Conf. on Development and Information Retrieval*. ACM Press, Santiago, 2015, pp. 959–962. DOI: 10.1145/2766462.2767830.
6. Wang P., Xu J., Xu B., Liu C., Zhang H., Wang F. Semantic clustering and convolutional neural network. *Proc. 53rd Annual Meeting on Natural Language Processing*. ACL Press, Beijing, 2015, vol. 2, pp. 352–357.

7. Ahlawat S., Batra V., Banerjee S., Saha J., Garg A.K. Hand gesture recognition using convolutional neural network. S. Bhattacharyya, A. Hassaniien, D. Gupta, A. Khanna, I. Pan (Eds.). *Proc. ICICC*. Singapore, Springer Publ., 2019, vol. 56, pp. 179–186. DOI: 10.1007/978-981-13-2354-6_20.
8. Xiang L., Zhao D., Yu K. Automatic vehicle identification in coating production line based on computer vision. *Proc. Intern. Conf. CSET*. 2016, pp. 260–267. DOI: 10.1142/9789814651011_0038.
9. Cha Y.J., Choi W. Vision-based concrete crack detection using a convolutional neural network. *Proc. CPSEMS*. Caicedo J., Pakzad S. (eds.). Springer Publ., Cham, 2017, vol. 2, pp. 71–73.
10. Balakrishnan A.V. *Theory of Kalman filtration*. Moscow, Mir Publ., 1988, 168 p.
11. Sonka M., Hlavac V., Boyle R. *Image Processing, Analysis and Machine Vision*. Thomson Publ., 2008, 866 p.
12. He K., Zhang X. *Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition*. arXiv.org, 2014. Available at: <https://arxiv.org/abs/1406.4729> (accessed May 15, 2019).
13. Salarpour A., Salarpour A., Fathi M. Vehicle tracking using Kalman filter and features. *SIPJ*. 2011, no. 2, pp. 1–8. DOI: 10.5121/sipij.2011.2201.
14. Bewley A., Ge Z. *Simple Online and Realtime Tracking*. arXiv.org, 2016. Available at: <https://arxiv.org/abs/1602.00763> (accessed May 15, 2019).
15. Wojke N., Bewley A., Paulus D. *Simple Online and Realtime Tracking with a Deep Association Metric*. arXiv.org, 2017. Available at: <https://arxiv.org/abs/1703.07402> (accessed May 15, 2019).
16. Orphan A.A., Ivankov A.Yu. Block image processing algorithms based on the Kalman filter in the problem of constructing a super-resolution. *Computer Optics*. 2014, vol. 38, no. 1, pp. 118–126.
17. Vasilkova M.A., Puchkov A.Yu. Preparation of input data sets for training deep neural networks. *Information Technologies, Energy and Economics. Proc. 16th Intern. Sci. and Tech. Conf.* 2019, vol. 1, pp. 262–265 (in Russ.).

Для цитирования

Пучков А.Ю., Дли М.И., Лобанева Е.И., Василькова М.А. Прогноз состояния объекта на основе применения фильтра Калмана и глубоких нейронных сетей // Программные продукты и системы. 2019. Т. 32. № 3. С. 368–376. DOI: 10.15827/0236-235X.127.368-376.

For citation

Puchkov A.Yu., Dli M.I., Lobaneva E.I., Vasilkova M.A. Predicting an object state based on applying the Kalman filter and deep neural networks. *Software & Systems*. 2019, vol. 32, no. 3, pp. 368–376 (in Russ.). DOI: 10.15827/0236-235X.127.368-376.