# Оптимизация для вычислительной архитектуры Эльбрус модифицированного метода Виола и Джонса*

Н.А. Бочаров, Е.Е. Лимонова, Б.Н. Парамонов, С.А. Усилин

**Аннотация.** В работе рассматривается современные модификации оригинального обучающего алгоритма Виолы и Джонса. Предложен модифицированный алгоритм каскадного обучения алгоритма Виолы и Джонса, основанный на новой структуре высокоуровневого классификатора. Предложенный алгоритм позволяет повысить точность распознавания, а также обеспечивает возможность дообучения. Описывается оптимизация реализованного усовершенствованного алгоритма Виолы и Джонса для архитектуры Эльбрус, позволяющая существенно увеличить быстродействие реализованного алгоритма для архитектуры Эльбрус.

**Ключевые слова:** *детектирование объектов, метод Виола и Джонса, архитектура Эльбрус, алгоритм классификации.*

## Введение

В последнее время поиск и распознавание объектов на цифровых изображениях и в видеопотоке из стадии лабораторных исследований используются промышленных системах обработки видеопотока. Для таких систем закономерно требование высокой скорости работы алгоритмов локализации объектов в изображениях. Каскадная структура классификатора, применяемая в оригинальном алгоритме Виолы и Джонса, позволяет добиться высокой производительности за счет быстрого отсеивания пустых скользящих окон, поскольку их количество существенно превышает количество подокон, содержащих объект. Время обработки пустого подокна отличается от времени обработки подокна с объектом в несколько раз (пропорционально длине каскада). Поскольку количество подокон, содержащих объект, отличается от кадра к кадру, отличается и время обработки каждого кадра. За счет того, что подокон с объектом на кадре заметно меньше подокон без объекта, различие во времени обработки измеряется не десятками раз, но десятками процентов, что, тем не менее, существенно в системах реального времени. Особенно неприятно это обстоятельство в силу того, что задача поиска объекта обычно предполагает дальнейшее распознавание или классификацию, а именно на кадрах, содержащих объект, быстродействие наихудшее.

Другая проблема использования метода Виолы-Джонса возникает при работе с достаточно вариативными объектами. Оригинальный метод строился в предположении, что карта яркостных контрастов на объекте не меняется при вариациях. Однако это не верно, например, для случаев наличия контрастных теней от других объектов. В качестве одного из решений может быть использована предварительная кластеризация обучающей выборки и последующее обучение независимого каскада для каждого кластера.

Немало работ посвящено модификациям метода Виолы и Джонса в части алгоритма обучения сильных классификаторов. Большое количество работ посвящено различным вариантам аппроксимации функционала качества композиции непрерывно дифференцируемыми оценками сверху [1-2]. Является важным алгоритм LogitBoost [3], сформулированный Фридманом, Хасти и Тибширани, который представляет собой адаптацию подхода логистической регрессии к идее бустинга. В результате этого построенный классификатор обладает целым набором интересных последствий: обученный таким образом классификатор оказывается оптимальным с точки зрения байесовской классификации, позволяет строить более устойчивый к шуму классификаторы, возникает возможность получать численные оценки вероятности принадлежности исследуемых объектов к найденным классам.

Несмотря на наличие приведенных выше модификаций алгоритма Виолы и Джонса, применение его для решения большого диапазона прикладных задач в индустриальных системах распознавания

* Работа выполнена при поддержке РФФИ (проекты № 17-29-03297 и № 15-29-06081)

часто оказывается затруднительным. Во-первых, значение классических признаков Хаара, даже в известных нормализованных модификациях, оказываются неинвариантны к изменению освещенности, а модификации признакового пространства, оперирующие с граничными точками, оказываются либо вычислительно трудоемкими, либо чувствительны к шумовым выбросам, либо неприспособленными к масштабированию. Во-вторых, алгоритм Виолы и Джонса, как и представленные модификации, решают задачу поиска объектов в «лабораторной» постановке: наборы прецедентов известны и зафиксированы, детектирование объектов выполняется на отдельных стационарных изображениях, скорость построенных детекторов оценивается по средней оценке. Однако индустриальные системы распознавания предъявляют к алгоритмам поиска объектов дополнительные требования. Так, например, в качестве источника данных часто выступает видеокамера, обеспечивающая вместо отдельных стационарных изображений коррелированную последовательность кадров, которые могут быть использованы для повышения производительности алгоритма. Хотя в соответствии с оригинальным методом обучение детектора ведется в «пакетном режиме» (обучающий набор прецедентов известен и зафиксирован), на практике регулярно возникает задача «дообучения» детектора в связи с появлением новых данных. Наконец, несмотря на то, что в исследовательской сфере важным атрибутом алгоритма является время его работы в среднем, в прикладных индустриальных системах (особенно для тех, которые возвращают ответ распознавания в синхронном режиме) не менее важным считается время работы алгоритма в худшем случае (из-за наличия физического ограничения на время обработки одного изображения).

В настоящей работе предложена модификация метода каскадного обучения алгоритма Виолы и Джонса, позволяющая существенно повысить как точность, так и быстродействие детектирования объектов.

## 1. Обзор модификаций алгоритма Виолы и Джонса в части каскадной структуры классификатора

Несмотря на то, что каскадная структура классификатора позволяет существенно ускорить производительность детектирования объектов, она не лишена недостатков.

Во-первых, в соответствии с архитектурой каскада, информация, полученная на текущем уровне каскада, ни в каком виде не передается на следующие уровни. Соответственно, решение о том, отбраковывать ли на очередном уровне каскада рассматриваемый участок изображения никак не зависит от того, насколько хорошо он был распознан на предыдущих уровнях. Такой подход иногда приводит к построению «хрупкого» классификатора, который «полностью ломается» при малейших колебаниях на отдельных уровнях. Для решения данной проблемы ученые предлагают использовать в качестве степень уверенности классификатора на предыдущем уровне в качестве слабого классификатора при построении классификатора текущего уровня [4, 5].

Второй недостаток классического каскада – это отсутствие оптимального способа построения уровней каскада. Дело в том, что при обучении классического каскадного классификатора для каждого уровня должны указываться несколько коррелирующих между собой параметров: количество слабых классификаторов, долю ложных срабатываний, долю ложных пропусков. Использование различных наборов данных параметров позволяют достигать компромисса между производительностью и качеством детектирования. В 2005 году был предложен новый вид каскадного классификатора, получивший название «легкий каскад» (Soft Cascade) [6, 7]. Структурно, легкий каскад похож на сильный классификатор, обученный с бустинга, однако способный отклонить отрицательные области после вычисления очередного слабого классификатора. Достигается это за счет сравнения частичных сумм линейных комбинаций слабых классификаторов с пороговым значением в процессе вычисления значения сильного классификатора. Процедура построения легкого каскада состоит из двух этапов: построение сильного классификатора и поиск пороговых значений отсечения.

Третий недостаток, присущий классическому каскадному классификатору – это неприменимость в случае большой вариативности обучающих прецедентов. В этом случае возможно решение путем кластеризации обучающей выборки, обучения различных каскадов для каждого кластера, иногда используя пре-классификатор для выбора целевого каскада [8, 9].

## 2. Алгоритм обучения классификатора Виолы и Джонса в виде решающего дерева сильных классификаторов

В данном разделе описывается разработанный алгоритм обучения классификатора Виолы и Джонса в виде решающего дерева, обеспечиваю-

щего несколько выходов с положительным результатом, благодаря чему обеспечивается качество детектирования объектов с сохранением времени работы в худшем случае.

Решающее дерево сильных классификаторов представляет собой вид бинарного решающего дерева: узел дерева – это сильный классификатор, на правое ребро которого попадают подокна, предположительно содержащие объект, а на левое – те, которые не распознались как объект, соответственно [10]. Окончательный ответ дается только в листьях, поскольку оригинальный каскадный классификатор, описанный в оригинальной работе Виолы и Джонса – это, по сути, древовидный классификатор, содержащий лишь один «положительный» выход (лист) и множество «отрицательных» выходов.

Предлагаемый метод построения (обучения) классификатора в виде решающего дерева основан на следующих предположениях:

– построение каскадного классификатора происходит итерационно. Каждый следующий уровень обучается, используя знания предыдущих уровней.
– обученный каскадный классификатор представим в виде конъюнкции сильных классификаторов (с одной лишь оговоркой о значимости порядка вычисления).

Следовательно, каскадный классификатор, состоящий из $N$ уровней, формально можно представить следующим образом:

$$Cascade(x) = S_1(x) \wedge S_2(x) \wedge S_3(x) \wedge \ldots \wedge S_N(x). \quad (1)$$

Будем считать, что каждый уровень $S_i$ возвращает +1, если искомый объект найден, и -1 в противном случае. Тогда формула (1) может быть переписана следующим образом:

$$Cascade(x) = [S_1(x) > 0] \cdot [S_2(x) > 0] \cdot \ldots \cdot [S_N(x) > 0] = \prod_{i=1}^{N} [S_i(x) > 0],$$

где $[S_i(x) > 0] = \begin{cases} 1, \text{если } S_i(x) > 0 \\ 0, \text{если } S_i(x) \le 0 \end{cases}$ – индикатор найденного объекта.

Также предполагаем, что любой путь от корня до самого нижнего узла древовидного классификатора может быть представлен как каскад, в котором отдельные сильные классификаторы входят в конъюнкцию с отрицанием.

Под взвешенным каскадом будем подразумевать классический линейный каскад, у которого для каждого уровня (сильного классификатора) $S_i(x)$ дополнительно задан вес (знаковый множитель) $w_i = \pm 1$ с которым данный уровень участвует в произведении при вычислении значения каскада.

Формула вычисления (1) взвешенного каскада записывается следующим образом:

$$WeightedCascade(x) = \prod_{i=1}^{N} [w_i \cdot S_i(x) > 0].$$

Для упрощения записи в дальнейших рассуждениях, будем схематично обозначать взвешенный каскад как последовательную конъюнкцию уровней, дополненных оператором отрицания тех их них, которые входят в произведение отрицательным весом.

Как и в случае с классическим линейным каскадом, взвешенный каскад строится итерационно. Но в отличии от классического каскада, где при обучении очередного уровня обновляется только отрицательная выборка, состоящая из ложных срабатываний уже обученного каскада, при обучении очередного уровня взвешенного каскада обновляется так же и положительная выборка, состоящая из успешно пройденных сквозь уже обученный каскад положительных примеров.

Опишем разработанный алгоритм обучения решающего дерева сильных классификаторов, состоящего из следующих шагов:

– *инициализация*. Пусть есть обучающая выборка $\mathbb{X}$ и валидационная выборка $\mathbb{X}^v$, состоящие из положительных образцов и отрицательных наборов изображений (изображений, не содержащих целевой объект). Пусть задан параметр $FPR^*$ – доля ложного срабатывания. Также пусть задано значение $N_{wc}$ – мощность уровня дерева (количество слабых классификаторов для каждого уровня дерева).
– *обучение вершины дерева*. Обучаем очередную вершину решающего дерева методом «в ширину», то есть, последовательно, справа налево, начиная с вершины-источника. Для подготовки обучающей выборки используем взвешенный каскад, построенный из уже обученных вершин, находящихся на пути к обучаемой вершине. Так, например, для обучения вершины $S_4$ соответствующий взвешенный каскад определяется как $WC_{S_4}=S_1 \wedge S_2$, для обучения вершины $S_5$ соответствующий взвешенный каскад определяется как $WC_{S_5}=S_1 \wedge \overline{S}_2$, для обучения вершины $S_6$ соответствующий взвешенный каскад имеет вид $WC_{S_6}=\overline{S}_1 \wedge S_3$ и так далее. Если при подготовке обучающей выборки для очередного уровня древовидного классификатора множество положительных или отрицательных примеров оказалось пустым, то рассматриваемая ветвь считается обученной полностью и дальнейшего обучения не требуется.

– *проверка текущего состояния обучения*. Если достигнуты целевые параметры обучения ($FPR<FPR^*$), оцененные на валидационной выборке $\mathbb{X}^v$, то закончить процедуру обучения. Иначе перейти на повторное обучение..

Блок-схема представленного алгоритма изображена на рисунке 1.

В отличии от классического каскада, для которого справедливо сильное монотонное падение доли верных срабатываний с каждым новым уровнем (напомним, что доля верных срабатываний для каскада, состоящего из $K$ уровней, оценивается как $TPR = \prod_{i=1}^{K} tpr_i$, где $tpr_i$ – доля верных срабатываний для $i$-го уровня), древовидный классификатор обеспечивает более медленное падение.

Связано это с наличием нескольких путей в классификаторе, возвращающих положительный ответ (сумма всех долей верного срабатывания взвешенных каскадов, данный составляющих древовидный классификатор). Исходя из определения взвешенного каскада, доли верных срабатываний для него определяется следующим образом:

$$TPR_{WC} = \prod_{i:w_i>0} trp_i \cdot \prod_{j:w_j<0} (1 - tpr_j),$$

где $tpr_i$ – доля верных срабатываний $i$-го уровня взвешенного каскада.

Тогда доля верных срабатываний для всего древовидного классификатора задается следующим образом:

$$TPR = \sum_{WC} \left( \prod_{i:w_i>0} trp_i \cdot \prod_{j:w_j<0} (1 - tpr_j) \right).$$

В качестве дополнительного развития описанного алгоритма можно использовать порог разделения положительной выборки при обучении очередного уровня древовидного классификатора. Введение такого порога позволяет уже на этапе обучения строить прореженное дерево решений без существенной потери в качестве детектирования.

Разработанный алгоритм позволяет обучать классификаторы, обладающие высокой вариативностью. Действительно, при обучении классического каскада для детектирования сложных объектов (слабо обобщаемых с точки зрения выбранного признакового пространства) начиная с некоторого момента мощности имеющихся признаков оказывается недостаточно для разделения положительной и отрицательных выборок. И хотя бустинг в конце концов позволит построить очередной уровень, обладающий высокой сложностью, то есть содержащий большое количество слабых классификаторов, с высокой степенью вероятности это приведет к переобучению. Описанный алгоритм построения древовидного классификатора, благодаря априорно установленному ограничению на сложность каждого обучаемого уровня, периодически «перебрасывает» положительные образцы на отрицательную ветку, выполняя тем самым естественную кластеризацию положительной выборки.

Еще одной важной особенностью изложенного алгоритма построения классификатора Виолы и Джонса в виде решающего дерева является возможность дообучения при расширении обучающей выборки. Классическая каскадная структура позволяет дообучение в части уменьшения доли ложных срабатываний за счет добавления новых уровней при расширения отрицательного обучающего набора. Улучшение же доли верного обнаружения возможно только путем полного переобучения. В противоположность каскаду, дообучение древовидного классификатора при расширении обучающей выборки может быть выполнено как с точки зрения улучшения ложных срабатываний, так и точки зрения повышения доли верного обнаружения за счет наличия нескольких путей с положительным ответом.

Эффективность предложенной высокоуровневой структуры классификатора Виолы и Джонса в виде решающего дерева оценивалась в задаче поиска логотипа платежной системы VISA на изображениях банковских карт. Несмотря на кажущеюся однотипность, используемые на банковских картах логотипы с точки зрения обучения классификаторов достаточно разнообразны (см. рисунок 2). Это связано как с большой вариативностью фона, так и со способом нанесения логотипа и печати банковской карты. В рамках эксперимента обучались два типа классификаторов Виолы и Джонса: классический каскадный классификатор и разработанный классификатор в виде решающего дерева.В качестве положительной обучающей выборки использовался набор изображений, состоящий из 1744 предварительно вырезанных цветных образцов логотипов платежной системы. В качестве отрицательной обучающей выборки использовались 7020 фрагментов цветных фотографий банковских карт, не содержащая логотипа платежной системы VISA. Оба детектора использовали в качестве пространства признаков семейство яркостных 2-прямоугольных признаков Хаара. Для упрощения подсчетов каждый уровень (сильный классификатор) каскада и древовидного классификатора состоял из 5 слабых классификаторов.

В качестве тестового набора данных использовались цветные фотографии банковских карт

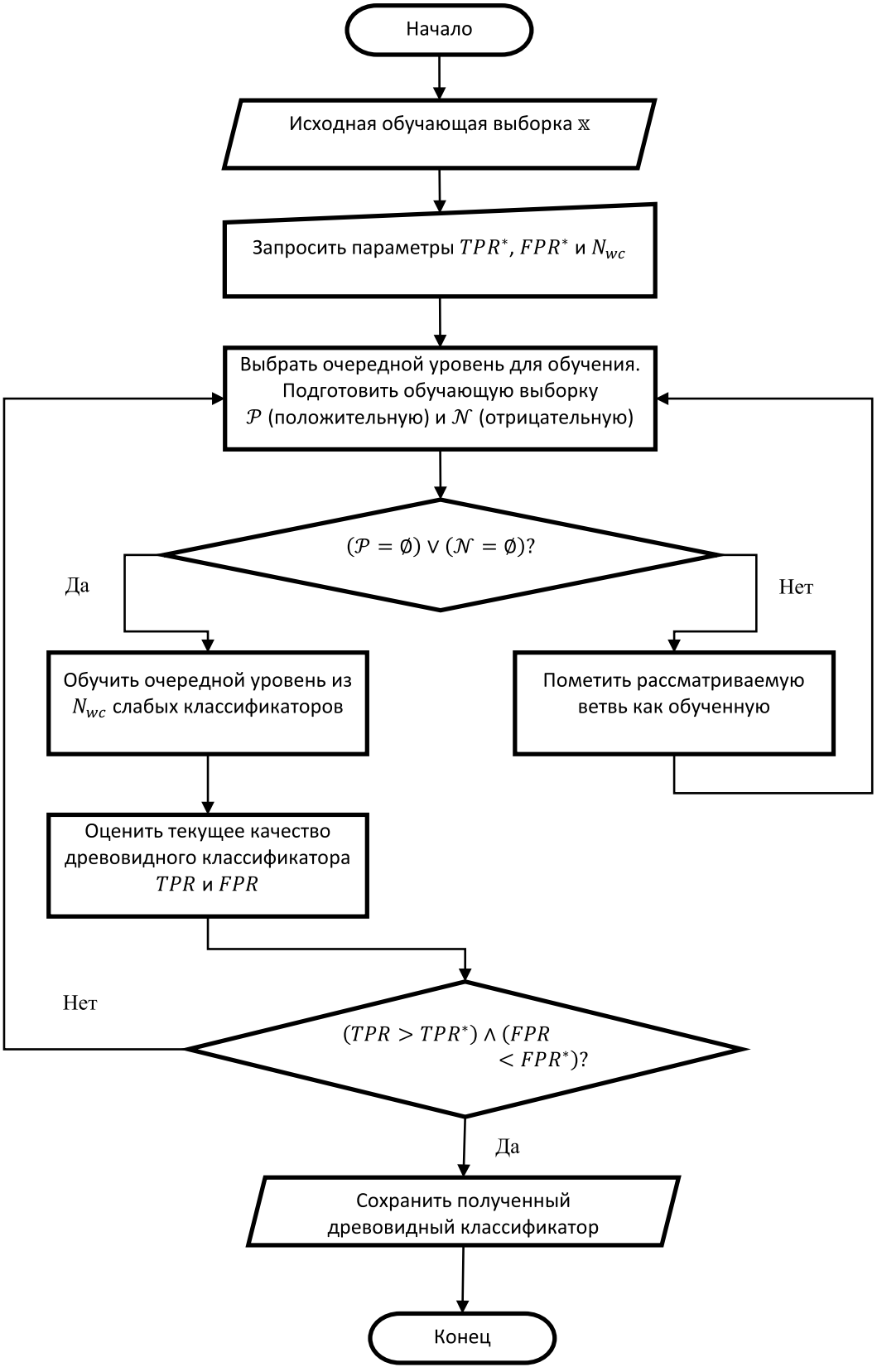


**Рис. 1** Блок-схема алгоритма обучения высокоуровневого классификатора в виде решающего дерева

платежной системы VISA разрешением 800x600 пикселей. Площадь банковской карты на каждом изображении составляло не менее 60% общей площади фотографии. Объем тестового набора составлял 1176 фотографий, примеры приведены на рисунке 2.

Было проведено несколько экспериментов.

Первый эксперимент (E1) состоял в обучении классического линейного каскада. Мощность (общее количество уровней, содержащихся в каскаде) обученного каскада составила 10 уровней.

Второй эксперимент (E2) представлял собой процедуру «дообучения» линейного каскада до общей древовидной структуры. Целью эксперимента была попытка повышения качества детектирования за «восстановления» потерянных в процессе обучения каскада положительных образцов. В результате был обучен древовидный классификатор, мощность которого составила 21 уровня, сохранив при этом длину максимального пути до положительного ответа как у исходного каскада.

Третий эксперимент (E3) заключался в обучении древовидного классификатора «с нуля» в соответствии с алгоритмом. Получившийся в результате древовидный классификатор обладает мощностью 22 уровня, обеспечивая при этом наименьшую по сравнению с предыдущими экспериментами длину максимального пути до положительного ответа (9 сильных классификаторов).

Для оценки качества использовалась $F$-мера [11], представляющая собой взвешенное гармоническое среднее между точностью и полнотой:

$$F = \frac{1}{\alpha \cdot \frac{1}{Precision} + (1-\alpha) \cdot \frac{1}{Recall}},$$

где $\alpha \in [0,1]$ определяет взаимный вес точности и полноты. Также -меру записывают в следующем виде:

$$F_\beta = \frac{(\beta^2+1) \cdot Precision \cdot Recall}{\beta^2 \cdot Precision + Recall},$$

где $\beta^2 = \frac{1-\alpha}{\alpha}, \beta \in [0,\infty)$. При $\beta \in [0,1]$ предпочтение отдается точности, при $\beta \in (1,\infty)$ больший вес приобретает полнота, а при $\beta = 1$ $F$ -мера придает одинаковый вес точности и полноте. Для экспериментов по сравнению классификаторов мы будем использовать $F$-меру с параметром $\beta = 10$, символизирующим факт, что полнота важнее точности в 10 раз.

Качество детектирования всех обученных классификаторов было проверено на подготовленном тестовом наборе, состоящем из 1176 фотографий банковских карт платежной системы VISA. Для подсчета верно найденных логотипов использовалась методика, предложенную в рамках соревнования The PASCAL Visual Object Classes (VOC) Challenge [12].

Результаты экспериментов приведены в таблице 1. С точки зрения сравнения качества детектирования с точки зрения выбранной $F$-метрики наилучшим оказался древовидный классификатор,


**Рис. 2** Пример используемых логотипов платежной системы VISA

построенный «с нуля» (эксперимент E3), а наихудшим – классический линейный каскад (эксперимент E1). Также заметим, что:

– дообучение линейного каскада до древовидной структуры позволяет повысить полноту детектора, немного потеряв при этом в точности;
– рост количества положительных выходов в классификаторе отрицательно сказывается на точности древовидного классификатора;
– обученный «с нуля» древовидный классификатор позволяет выполнить лучшую кластеризацию положительной обучающей выборки, улучшая полноту детектора.

**Табл. 1**

Сравнение качества детектирования классификаторов

| Эксперимент | Precision | Recall | $F$ (при $\beta$=10) |
|---|---|---|---|
| E1 | 0,9068 | 0,9184 | 0,9183 |
| E2 | 0,9046 | 0,9269 | 0,9266 |
| E3 | 0,8893 | 0,9294 | 0,9290 |

В таблице 2 представлены значения, позволяющие оценить вычислительную сложность обученных детекторов, оцененную с помощью . Из представленных данных видно, что в среднем наилучшая производительность у классического линейного каскада. Однако обученный в рамках эксперимента E3 древовидный классификатор обладает наименьшей длиной максимального пути до положительного ответа, следовательно, обеспечит лучшую производительность в худшем случае.

**Табл. 2**

Вычислительная сложность обученных детекторов

| Эксперимент | Мощность | Длина максимального пути до положительного ответа | Уровней в среднем |
|---|---|---|---|
| E1 | 10 | 10 | 2,00 |
| E2 | 21 | 10 | 2,52 |
| E3 | 22 | 9 | 2,48 |

### 3. Особенности архитектуры Эльбрус

Архитектура Эльбрус относится к категории архитектур, использующих принцип широкого командного слова (Very Long Instruction Word, VLIW). На процессорах с VLIW-архитектурой компилятор формирует последовательности групп команд (широкие командные слова), в которых отсутствуют зависимости между командами внутри каждой группы и сведены к минимуму зависимости между командами в разных группах. Далее команды внутри каждой группы исполняются параллельно, что обеспечивает высокий уровень параллелизма на уровне команд.

Благодаря тому, что распараллеливание на уровне команд целиком обеспечивается оптимизирующим компилятором, аппаратура для исполнения команд значительно упрощается, поскольку теперь она не решает задач распараллеливания, как в случае, например, с архитектурой x86. Энергопотребление VLIW-процессора снижается: от него больше не требуется анализировать зависимости между операндами или переставлять операции, поскольку все эти задачи возложены на компилятор. Стоит отметить, что компилятор располагает значительно большими вычислительными и временными ресурсами, чем аппаратные анализаторы исходного кода, и поэтому может выполнять анализ тщательнее и находить больше независимых операций [13].

Кроме того, особенностью архитектуры Эльбрус являются методы работы с памятью. Часто доступ во внешнюю память может занимать значительное время и замедлять вычисления. Для решения этой проблемы используется кэширование, однако оно имеет свои недостатки, поскольку кэш имеет строго ограниченный размер и типичные алгоритмы кэширования направлены на сохранение в кэше только часто используемых данных.

Другим способом повышения эффективности доступа в память является использование методов предварительной подкачки данных. Эти методы позволяют прогнозировать обращения в память и производить подкачку данных в кэш или другое специальное устройство за некоторое время до их использования. Они делятся на программные и аппаратные. В программных методах специальные инструкции предварительной подкачки планируются на этапе компиляции. Они обрабатываются аналогично обычным обращениям в память. Аппаратные методы работают во время исполнения программы и используют для прогнозирования динамическую информацию об обращениях в память. Они требуют наличия дополнительных модулей в составе микропроцессора, однако не нуждаются в специальных инструкциях подкачки и могут работать асинхронно.

Процессоры Эльбрус поддерживают программно-аппаратный метод подкачки. При этом в аппаратную часть микропроцессора включено специальное устройство для обращения к массивам (Array Access Unit, AAU), но необходимость подкачки определяется компилятором, генерирующим

специальные инструкции (инициализация AAU, запуск и останов программы предварительной подкачки, асинхронные инструкции предварительной подкачки и синхронные инструкции пересылки данных из буфера подкачки массивов (Array Prefetch Buffer, APB) в регистровый файл). Использование устройства подкачки эффективнее помещения элементов массива в кэш, поскольку элементы массивов чаще всего обрабатываются последовательно и редко используются более одного раза [14]. Однако необходимо отметить, что использование буфера предварительной подкачки на Эльбрусе возможно только при работе с выровненными данными. За счет этого чтение/запись выровненных данных происходят заметно быстрее, чем соответствующие операции для невыровненных данных.

Также микропроцессоры Эльбрус поддерживают несколько видов параллелизма помимо параллелизма на уровне команд: векторный параллелизм, параллелизм потоков управления, параллелизм задач в многомашинном комплексе.

## 4. Средства повышения производительности программного кода на платформе Эльбрус

Повысить быстродействие разработанного алгоритма на процессоре архитектуры Эльбрус можно путем использования высокопроизводительной библиотеки EML и интринсиков (англ. Intrinsics), реализующих векторный параллелизм.

Библиотека EML – высокопроизводительная библиотека, предоставляющая пользователю набор разнообразных методов для обработки сигналов, изображений, видео, а также математические функций и операции [15]. Она предназначена для использования в программах, написанных на языках C/C++. Библиотека EML включает в себя несколько следующие группы функций:

- Vector – функции для работы с массивами (векторами) данных;
- Algebra – функции линейной алгебры;
- Signal – функции обработки сигналов;
- Image – функции обработки изображений;
- Video – функции обработки видео;
- Volume – функции преобразования трехмерных структур;
- Graphics – функции для рисования фигур.

Для массивов данных определены самые востребованные операции, например, сложение, поэлементное перемножение, подсчет среднего значения массива и т.д.

Разработчики могут использовать векторный параллелизм на платформе Эльбрус напрямую, то есть выполнять одну и ту же операцию над одним регистром, содержащим сразу несколько элементов данных. Для этого используются функции-интринсики, вызовы которых заменяются компилятором на высокоэффективный код для данной платформы. Микропроцессор Эльбрус-4С поддерживает набор инструкций версии 3, в котором размер регистра составляет 64 бита. С помощью регистра в 64 бита можно одновременно обработать 2 вещественных 32-битных числа, 4 16-битных целых числа или 8 8-битных целых чисел и таким образом повысить производительность. Набор интринсиков микропроцессора Эльбрус во многом схож с набором интринсиков SSE, SSE2 и включает в себя операции для преобразования данных, инициализации элементов вектора, арифметических операций, побитовых логических операций, перестановки элементов вектора и др.

## 5. Оптимизация классификатора Виолы и Джонса для архитектуры Эльбрус

Экспериментальная проверка разработанного алгоритма классификации Виолы и Джонса была выполнена в задаче детектирования третьей страницы паспорта Российской Федерации, содержащей номер и серию паспорта, а также фамилию, имя, отчество, пол, дату и место рождения гражданина РФ. Для решения этой задачи был обучен бинарный классификатор, на вход которому поступает серое изображение, приведенное к размеру 480 на 360 пикселей. В случае, если приведение к данному размеру не сохраняет пропорций входного изображения, к соответствующему размеру приводится наименьшее измерение изображения, а второе уменьшается с сохранением пропорций. Выходы классификатора интерпретируются как наличие или отсутствие страницы паспорта, ориентированной таким образом, что ее стороны параллельны краям кадра.

Разработанный классификатор оперирует модифицированными двупрямоугольными признаками Хаара, которые вычисляются на основе интегральных изображений:

$$\begin{gathered} s_0 = I_{y+H,x+W} - I_{y,x+W} - I_{y+H,x} + I_{y,x} \\ s_1 = I_{y+q+H,x+p+W} - I_{y+q,x+p+W} - I_{y+q+H,x+p} + I_{y+q,x+p} \\ r = 255\, s_0/(s_0 + s_1 + 1) \end{gathered} \quad (2)$$

где $I$ - интегральное изображение, $W$, $H$ - ширина и высота прямоугольника, по которому вычисляется сумма, ($x$, $y$) - координата верхнего левого угла подокна, ($p$, $q$) - сдвиг второго прямоугольника для вычисления суммы, $r$ - результат вычисления признака. В отличие от классического двупрямоуголь-

ного признака Хаара, такой признак более устойчив к изменениям яркости на исходном изображении.

В ходе детектирования эти вычисления производятся для каждого подокна исходной картинки. Поскольку для нашего классификатора шаг, с которым берутся эти подокна, равен 1 как по горизонтали, так и по вертикали, операции вида (2) для каждой строки исходной картинки легко реализовать с использованием таких векторных операций EML как:

– eml_Status eml_Vector_AddShift_32S(const eml_32s *pSrc1, const eml_32s *pSrc2, eml_32s *pDst, eml_32s len, eml_32s len) - выполняет поэлементное сложение двух массивов типа 32-битное целое Src1 и pSrc2 с домножением на 2^shift и записывает результат в массив pDst.
– eml_Status eml_Vector_SubShift_32S(const eml_32s *pSrc1, const eml_32s *pSrc2, eml_32s *pDst, eml_32s len, eml_32s len) - выполняет поэлементное вычитание двух массивов типа 32-битное целое Src1 и pSrc2 с домножением на 2^shift и записывает результат в массив pDst.
– eml_Status eml_Vector_AddCShift_32S(const eml_32s *pSrc, eml_32s val, eml_32s *pDst, eml_32s len, eml_32s shift) - выполняет прибавление константы val к каждому элементу массива типа 32-битное целое pSrc с домножением на 2^shift и записывает результат в массив pDst.
– eml_Status eml_Vector_DivShift_32S(const eml_16s *pSrc1, const eml_16s *pSrc2, eml_16s *pDst, eml_32s len, eml_32s shift) - выполняет поэлементное деление двух массивов типа 32-битное целое Src1 и pSrc2 с домножением на 2^shift и записывает результат в массив pDst.

Таким образом, вычисление 2-прямоугольных признаков Хаара было векторизовано. Следует также отметить, что операции библиотеки EML не требуют специального выравнивания данных. На рисунке 3 показано время вычисления такого признака в зависимости от количества применений классификатора по горизонтали. Эксперименты выполнялись на машине с процессором Эльбрус-4С. Можно видеть, что EML позволяет до 10 раз ускорить время вычисления 2-прямоугольного признака Хаара в процессе детектирования объекта, причем наиболее эффективно это вычислений при количестве приложений классификатора от 64 и более.

Далее оценим время классификации одного подокна в зависимости от длины исходной картинки (см. рисунок 4). Можно видеть, что использование EML позволило уменьшить время детектирования страницы паспорта РФ до 30 раз.

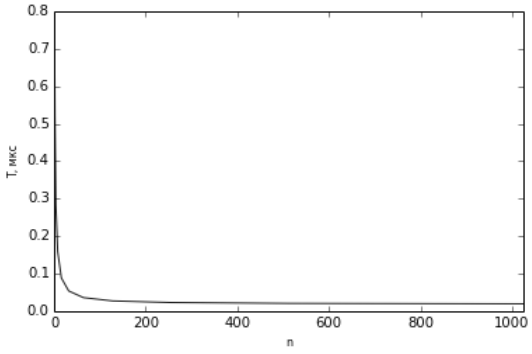


**Рис. 3** Зависимость времени вычисления модифицированного 2-прямоугольного признака Хаара *T* от количества применений классификатора по горизонтали *n*

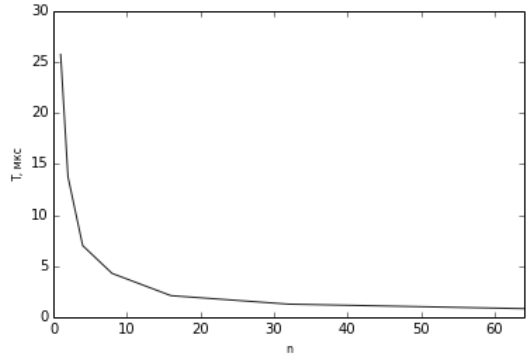


**Рис. 4** Зависимость времени вычисления классификации *T* от количества применений классификатора по горизонтали *n*

Кроме того, данный метод оптимизации быстродействия можно обобщить на некоторые случаи шага детектора, отличного от 1. Если размер окна классификатора кратен шагу, то вычисления с использованием векторных операций можно применить на модифицированной интегральной картинке, сформированной из соответствующих элементов полного интегрального изображения.

## Заключение

Предложенный в работе модифицированный метод каскадного обучения алгоритма Виолы и Джонса, позволяющая существенно повысить как точность, так и быстродействие детектирования объектов, удалось эффективно реализовать на платформе Эльбрус. Проведенные эксперименты показывают, что использование предложенных механизмов для платформы Эльбрус позволило уменьшить время детектирования страницы па-

спорта РФ до 30 раз по сравнению исходным вариантом реализации, не учитывающим особенности платформы Эльбрус.

## Литература


1. *Huang C. et al*. Vector boosting for rotation invariant multi-view face detection // Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on. 2005. Vol. 1. 446–453.
2. *Li S.Z., Zhang Z.Q*. FloatBoost learning and statistical face detection // IEEE Trans. Pattern Anal. Mach. Intell. 2004. Vol. 26, № 9. 1112–1123.
3. *Domingo C., Watanabe O*. MadaBoost: A Modification of AdaBoost // Conference on Computational Learning Theory (COLT). 2000. 180–189.
4. *Xiao R., Zhu L., Zhang H.-J*. Boosting Chain Learning for Object Detection // ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision. 2003. 709. P.
5. *Wu B. et al*. Fast rotation invariant multi-view face detection based on real AdaBoost // In Sixth IEEE International Conference on Automatic Face and Gesture Recognition. 2004. P. 79–84.
6. *Dollár P. et al*. Integral Channel Features // BMVC 2009 London Engl. 2009. 1–11.
7. *Bourdev L., Brandt J*. Robust Object Detection via Soft Cascade // Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2 - Volume 02. Washington, DC, USA: IEEE Computer Society, 2005. 236–243.
8. *Wu B. et al*. Fast rotation invariant multi-view face detection based on real AdaBoost // In Sixth IEEE International Conference on Automatic Face and Gesture Recognition. 2004. 79–84.
9. *Huang C.H.C. et al*. Vector boosting for rotation invariant multi-view face detection // Tenth IEEE Int. Conf. Comput. Vis. Vol. 1. 2005. Vol. 1.
10. *Норушис А.* Построение логических (древообразных) классификаторов методами нисходящего поиска (обзор) // Статистические проблемы управления. Вильнюс, 1990. Vol. 93. 131–158.
11. *Powers D.M.W*. Evaluation: From Precision, Recall and F-Measure To Roc, Informedness, Markedness & Correlation // J. Mach. Learn. Technol. 2011. Vol. 2, № 1. 37–63.
12. *Everingham M. et al*. The pascal visual object classes (VOC) challenge // Int. J. Comput. Vis. 2010. Vol. 88, № 2. 303–338.
13. *Ким А.К., Бычков И.Н.* и др. Российские технологии "Эльбрус" для персональных компьютеров, серверов и суперкомпьютеров // Современные информационные технологии и ИТ-образование, М.: Фонд содействия развитию интернет-медиа, ИТ-образования, человеческого потенциала «Лига интернет-медиа», 2014, № 10. 39-50.
14. *Ким А. К., Перекатов В. И., Ермаков С. Г.* Микропроцессоры и вычислительные комплексы семейства «Эльбрус». – СПб.: Питер, 2013. – 272 С.
15. *Ишин П.А., Логинов В.Е., Васильев П.П.* Ускорение вычислений с использованием высокопроизводительных математических и мультимедийных библиотек для архитектуры Эльбрус // Вестник воздушно-космической обороны, М.: Научно-производственное объединение «Алмаз» им. акад. А.А. Расплетина, 2015, № 4 (8). 64-68.



**Бочаров Никита Алексеевич.** Специалист АО «МЦСТ». Окончил в 2017 году МФТИ. Количество печатных работ: 4. Область научных интересов: вычислительная техника, информатика.
E-mail: bocharov.na@phystech.edu

**Лимонова Елена Евгеньевна.** Математик ИСА ФИЦ ИУ РАН. Окончила в 2017 г. МФТИ. Количество печатных работ: 13. Область научных интересов: обработка изображений, распознавание образов на мобильных устройствах. E-mail: limonova@smartengines.biz.

**Парамонов Борис Николаевич.** Г.н.с. ПАО «ИНЭУМ им. И.С. Брука».Д.т.н., профессор. Окончил в 1977 г. Киевское высшее инженерное радиотехническое училище ПВО. Количество печатных работ: 120. Область научных интересов: управление, вычислительная техника, информатика.
E-mail: paramonov_n_b@mail.ru

**Усилин Сергей Александрович.** Исполнительный директор ОАО «Смарт Энджинс Сервис». Окончил в 2009 г. МФТИ. Количество печатных работ: 24. Область научных интересов: цифровая обработка изображений, распознавание образов, поиск объектов на изображениях и в видеопотоке.
E-mail: usilin@smartengines.ru.

## Optimization for the Elbrus computing architecture of the modified method of Viola and Jones

*N.A. Bocharov, E.E. Limonova, B.N. Paramonov, S.A. Usilin*

**Abstract.** In the work, modern modifications of the original training algorithm by Viola and Jones are considered. A modified cascade learning algorithm for the Viola and Jones algorithm is proposed, based on the new structure of a high-level classifier. The proposed algorithm allows to increase the accuracy of recognition, and also provides the possibility of additional training. The optimization of the implemented improved algorithm of Viola and Jones for the Elbrus architecture is described, which allows to significantly increase the speed of the implemented algorithm for the Elbrus architecture.

**Keywords:** object detection, Viola and Jones method, Elbrus architecture, classification algorithm.

### References

1. *Huang C. et al*. Vector boosting for rotation invariant multi-view face detection // Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on. 2005. Vol. 1. 446–453.
2. *Li S.Z., Zhang Z.Q*. FloatBoost learning and statistical face detection // IEEE Trans. Pattern Anal. Mach. Intell. 2004. Vol. 26, № 9. 1112–1123.
3. *Domingo C., Watanabe O*. MadaBoost: A Modification of AdaBoost // Conference on Computational Learning Theory (COLT). 2000. 180–189.
4. *Xiao R., Zhu L., Zhang H.-J*. Boosting Chain Learning for Object Detection // ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision. 2003. 709. P.
5. *Wu B. et al*. Fast rotation invariant multi-view face detection based on real AdaBoost // In Sixth IEEE International Conference on Automatic Face and Gesture Recognition. 2004. P. 79–84.
6. *Dollár P. et al*. Integral Channel Features // BMVC 2009 London Engl. 2009. 1–11.
7. *Bourdev L., Brandt J*. Robust Object Detection via Soft Cascade // Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2 - Volume 02. Washington, DC, USA: IEEE Computer Society, 2005. 236–243.
8. *Wu B. et al*. Fast rotation invariant multi-view face detection based on real AdaBoost // In Sixth IEEE International Conference on Automatic Face and Gesture Recognition. 2004. 79–84.
9. *Huang C.H.C. et al*. Vector boosting for rotation invariant multi-view face detection // Tenth IEEE Int. Conf. Comput. Vis. Vol. 1. 2005. Vol. 1.
10. *Norushis A*. Construction of logical (tree-like) classifiers of methods of top-down search (review) // Statistical problems of management. Vilnius, 1990. Vol. 93. 131-158.
11. *Powers D.M.W*. Evaluation: From Precision, Recall and F-Measure To Roc, Informedness, Markedness & Correlation // J. Mach. Learn. Technol. 2011. Vol. 2, № 1. 37–63.
12. *Everingham M. et al*. The pascal visual object classes (VOC) challenge // Int. J. Comput. Vis. 2010. Vol. 88, № 2. 303–338.
13. *Kim A.K., Bychkov I.N.* Russian Technologies "Elbrus" for personal computers, servers and supercomputers // Modern Information Technologies and IT Education, Moscow: Foundation for the Promotion of Internet Media, IT Education, Human Capacity "League of Internet Media", 2014, No. 10. 39-50.
14. *Kim A.K., Bychkov I.N.*, *Ermakov S.G*. Microprocessors and computing systems of the Elbrus family. - SPb: Peter, 2013. - 272 C.
15. *Ishin P.A., Loginov V.E., Vasilyev P.P*. Acceleration of computations using high-performance mathematical and multimedia libraries for the architecture of Elbrus // Bulletin of Aerospace Defense, Moscow: Almaz Scientific and Production Association. acad. A.A. Raspletin, 2015, No. 4 (8). 64-68.

**Bocharov N. A.,** specialist "MCST" Ltd, Graduated in 2017 Moscow Institute of Physics and Technology. Number of publications: 5. Research interests: computer technology, computer science**.**
E-mail: bocharov.na@phystech.edu.

**Limonova E. E.,** mathematician of the Institute for Systems Analysis, Federal Research Center "Computer Science and Control" of Russian Academy of Sciences. Graduated in 2017 Moscow Institute of Physics and Technology. Number of publications: 13. Research interests: image processing, pattern recognition on mobile devices. E-mail: limonova@smartengines.biz.

**Paramonov B. N.**, Chief Researcher "INEUM named after is Brook" Corp., doctor of technical sciences, prof. Graduated in 1977 from the Kiev Higher Engineering Radio Engineering School of Air Defense. Number of publications: 1209. Area of scientific interests: management, computer technology, computer science. E-mail: paramonov_n_b@mail.ru.

**Usilin S. A.**, Executive Director of "Smart Engines Service" Ltd. Graduated in 2009 from the Moscow Institute of Physics and Technology. Number of publications: 24. Research interests: digital image processing, image recognition, image search and video streaming. E-mail: usilin@smartengines.ru.