ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНОРМАЦИОННЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

КАФЕДРА МАТЕМАТИЧЕСКОГО И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

АВТОМАТИЗИРОВАННАЯ СИСТЕМА АНАЛИЗА ИЗОБРАЖЕНИЙ НА ПРЕДМЕТ СКРЫТОЙ ИНФОРМАЦИИ

Магистерская диссертация
обучающегося по направлению подготовки 02.04.01 Математика и компьютерные науки
очной формы обучения, группы 07001531
Ставрова Евгения Александровича

Научный руководитель
к.т.н., доцент
Бурданова Е.В.

Рецензент
к.т.н., доцент
Прохоренко Е.И.

БЕЛГОРОД 2017

Оглавление

Введение

Цифровая стеганография – наука о скрытой передаче информации, которая часто осуществляется за счет встраивания передаваемого сообщения в некий не вызывающий подозрения цифровой объект путем незначительной его модификации. Результат встраивания передается по каналу связи получателю, который извлекает встроенное сообщение. Это эффективное средство защиты информации, становящееся особенно актуальным в случае, когда применение криптографических методов невозможно или ограничено. Наиболее популярным направлением в стеганографии является развитие методов встраивания сообщений в изображения. В противовес стеганографии развивается стегоанализ.

Стеганализ широко изучался в последнее десятилетие. Его основная цель – обнаружить присутствие секретных сообщений в цифровых контейнерах, таких как цифровые изображения, поступающие из известного источника. Обычно эта задача формулируется как задача двоичной классификации, чтобы различать пустой и стего контейнер. В общем случае существующие методы в основном делятся на два этапа: выделение признаков и классификация. На этапе выделения из каждого изображения извлекается набор вручную подобранных признаков, чтобы зафиксировать влияние операций внедрения. Успех стегоанализа в целом зависит от подбора признаков. Однако отсутствие точных моделей естественных изображений усложняет эту работу. Поэтому предлагаются различные эвристические методы. С увеличением сложности методов стеганографии учитываются более сложные статистические зависимости между отдельными элементами. В последние годы именно направление статистического стегоанализа развивается интенсивнее всего.

На этапе классификации методы, такие как метод опорных векторов или ансамбль классификаторов, обучаются на основе подобранных признаков. Поскольку этапы извлечения признаков и классификации разделены, они не могут быть оптимизированы одновременно. Это означает, что на этапе извлечения признака невозможно собирать полезную информацию.

Это обусловливает актуальность задачи построения автоматизированной системы для стегоанализа на основе свёрточной нейронной сети. В данной системе реализуется возможность объединения и автоматизации этапов извлечения признаков и классификации.

Глубокие модели обучения - это такой класс моделей, который может автоматически извлекать признаки. Вдохновленные тем фактом, что человеческий мозг обрабатывает информацию иерархически с глубокой архитектурой, исследователи ожидали в течение десятилетий обучать глубокие многослойные нейронные сети. Но не было достигнуто никакого успеха до 2006 года, когда Джефф Хинтон сделал прорыв[1]. В этой работе Хинтон представил Deep Belief Networks (DBNs) с жадной поэтапной неконтролируемой предварительной подготовкой, которая может изучить иерархию признаков на одном уровне, за раз. Аналогичным образом были предложены многие другие модели глубокого обучения, такие как Глубинные Машины Больцмана[2], глубокие автоэнкодеры[3] и свёрточные нейронные сети (CHC)[4]. Эти модели имеют глубокие архитектуры, которые состоят из нескольких уровней нелинейных операций и могут обучаться с использованием либо контролируемых, либо неконтролируемых подходов для изучения иерархических представлений путем создания высокоуровневых признаков из низкоуровневых. Как правило, глубокие архитектуры могут представлять некоторые признаки, которые не могут быть эффективно представлены с помощью неглубоких архитектур. Они фактически оказались более мощными обучающими схемами для многих

задач искусственного интеллекта (ИИ), таких как распознавание объектов и обработка естественного языка.

Целью данной диссертационной работы является разработка и реализация автоматизированной системы для анализа изображений на предмет скрытой информации на основе модели СНС разработанной для стегоанализа. Для достижения этой цели были поставлены и решены следующие задачи:

1) Исследовать и проанализировать существующие методы стеганографии и обнаружение скрытых сообщений в рамках статистического и нейросетевого подходов в стегоанализе.

2) Разработать архитектуру свёрточной нейронной сети для нужд стегоанализа и архитектуру системы для анализа изображений.

3) Создать программный продукт.

4) Провести анализ эффективности системы по сравнению с сущетвующими методами статистического стегоанализа.

Объектом исследования является применение глубокого обучения в стегоанализе.

Предметом исследования является применение свёрточной нейронной сети для стегоанализа изображений.

Научная новизна данной работы состоит в предложенной архитектуре свёрточной нейронной сети.

Научно-практическая ценность результатов диссертации заключается в возможности применения предложенной архитектуры нейронной сети при практической разработке стегоаналитических систем, обладающих большей точностью и надёжностью. Например, систем фильтрации трафика для выявления и устранения скрытых каналов передачи информации.

ГЛАВА 1. МОДЕЛИ, МЕТОДЫ И АЛГОРИТМЫ СТЕГАНОГРАФИЧЕСКОГО СОКРЫТИЯ И ОБНАРУЖЕНИЯ СКРЫТОЙ ИНФОРМАЦИИ

1.1 Основные сведения о стеганографии

Стеганография - это метод организации связи, который собственно скрывает само наличие связи. В отличие от криптографии, где неприятель точно может определить является ли передаваемое сообщение зашифрованным текстом, методы стеганографии позволяют встраивать секретные сообщения в безобидные послания так, чтобы невозможно было заподозрить существование встроенного тайного послания.

Слово "стеганография" в переводе с греческого буквально означает "тайнопись" (steganos - секрет, тайна; graphy - запись). К ней относится огромное множество секретных средств связи, таких как невидимые чернила, микрофотоснимки, условное расположение знаков, тайные каналы и средства связи на плавающих частотах и т. д.

Стеганография занимает свою нишу в обеспечении безопасности: она не заменяет, а дополняет криптографию. Сокрытие сообщения методами стеганографии значительно снижает вероятность обнаружения самого факта передачи сообщения. А если это сообщение к тому же зашифровано, то оно имеет еще один, дополнительный, уровень защиты.

В настоящее время в связи с бурным развитием вычислительной техники и новых каналов передачи информации появились новые стеганографические методы, в основе которых лежат особенности представления информации в компьютерных файлах, вычислительных сетях и т. п. Это дает нам возможность говорить о становлении нового направления - компьютерной стеганографии.

Несмотря на то, что стеганография как способ сокрытия секретных данных известна уже на протяжении тысячелетий, компьютерная стеганография - молодое и развивающееся направление.

Как и любое новое направление, компьютерная стеганография, несмотря на большое количество открытых публикаций и ежегодные конференции, долгое время не имела единой терминологии.

До недавнего времени для описания модели стеганографической системы использовалась предложенная 1983 году Симмонсом [5] так называемая "проблема заключенных". Она состоит в том, что два индивидуума (Алиса и Боб) хотят обмениваться секретными сообщениями без вмешательства охранника (Вилли), контролирующего коммуникационный канал. При этом имеется ряд допущений, которые делают эту проблему более или менее решаемой. Первое допущение облегчает решение проблемы и состоит в том, что участники информационного обмена могут разделять секретное сообщение (например, используя кодовую клавишу) перед заключением. Другое допущение, наоборот, затрудняет решение проблемы, так как охранник имеет право не только читать сообщения, но и модифицировать (изменять) их.

Позднее, на конференции Information Hiding: First Information Workshop в 1996 году было предложено использовать единую терминологию и обговорены основные термины[6].

Стеганографическая система или стегосистема - совокупность средств и методов, которые используются для формирования скрытого канала передачи информации.

При построении стегосистемы должны учитываться следующие положения:

Обобщенная модель стегосистемы представлена на рис. 1.1.

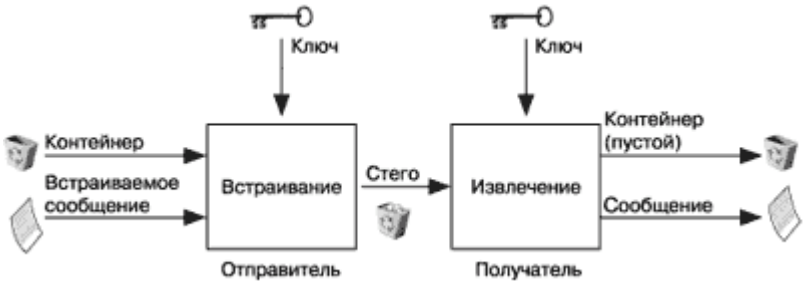


Рис. 1.1. Обобщённая модель стегосистемы.

В качестве данных может использоваться любая информация: текст, сообщение, изображение и т. п.

В общем же случае целесообразно использовать слово "сообщение", так как сообщением может быть как текст или изображение, так и, например, аудиоданные. Далее для обозначения скрываемой информации, будем использовать именно термин сообщение.

Контейнер - любая информация, предназначенная для сокрытия тайных сообщений.

Пустой контейнер - контейнер без встроенного сообщения; заполненный контейнер или стегоконтейнер, содержащий встроенную информацию.

Встроенное (скрытое) сообщение - сообщение, встраиваемое в контейнер.

Стеганографический канал или просто стегоканал - канал передачи стего.

Стегоключ или просто ключ - секретный ключ, необходимый для сокрытия информации. В зависимости от количества уровней защиты (например, встраивание предварительно зашифрованного сообщения) в стегосистеме может быть один или несколько стегоключей.

1.1.1. Классификация стеганографических систем

По аналогии с криптографическими системами, в стеганографии различают системы с секретным ключом и системы с открытым ключом.

В стеганографической системе с секретным ключом используется один ключ, который должен быть заранее известен абонентам до начала скрытого обмена секретными сообщениями либо переслан по защищенному каналу.

В стегосистеме с открытым ключом для встраивания и извлечения тайного сообщения используются разные ключи, причем вывести один ключ из другого с помощью вычислений невозможно. Один из ключей (открытый) может передаваться свободно по незащищенному каналу связи, а второй, секретный ключ, – по защищенному каналу. Данная схема хорошо работает при взаимном недоверии отправителя и получателя.

Учитывая все многообразие стеганографических систем, выделяют четыре основных типа стегосистем:

1) Безключевые стегосистемы

Бесключевая стегосистема определяется следующим образом. Совокупность $\sum = (C, M, S, E, D)$, где C – множество контейнеров-оригиналов, M – множество секретных сообщений, причём $|M| \leq |C|$, S – множество контейнеров – результатов, причём $sim(C, S) \to 1$, $E: C \times M \to S$ и $D: S \to M$ – соответственно функции прямого(встраивание) и обратного(извлечение) стегопреобразований, причём $D[E(c, m)] = m$ для любых $m \in M$ и $c \in C$, называется бесключевой стегосистемой.

2) Система с секретным ключом

Совокупность $\sum = (C, M, K, S^K, E, D)$, где C – множество контейнеров-оригиналов, M – множество секретных сообщений, причём $|M| \leq |C|$, S^K – множество контейнеров – результатов, причём $sim(C, S^K) \to 1$, K – множество секретных стегоключей, $E: C \times M \times K \to S^K$ и $D: S^K \times K \to M$ – функции прямого и обратного стегопреобразования со свойством $D[E(c, m, k), k] = m$ для любых $m \in M$, $c \in C$ и $k \in K$.

3) Стегосистема с открытым ключом.

Совокупность $\sum = (C, M, K, S^K, E, D)$, где C – множество контейнеров-оригиналов, M – множество секретных сообщений, причём $|M| \leq |C|$, S^K – множество контейнеров – результатов, причём $sim(C, S^K) \to 1$, $K = (k_O, k_C)$ – множество пар стегоключей (открытый ключ k_O используется для скрытия информации, а секретный ключ k_C – для её извлечения), $E: C \times M \times k_O \to S^K$ и $D: S^K \times k_C \to M$ – функции прямого и обратного стегопреобразования со свойством $D[E(c, m, k_O), k_C] = m$ для любых $m \in M$, $c \in C$ и $k_O, k_C \in K$.

4) Стегосистемы смешанного типа.

В большинстве приложений более предпочтительными являются безключевые стегосистемы, хотя такие системы могут быть сразу скомпрометированы в случае, если противник узнает применяемое стеганографическое преобразование. В связи с этим в безключевых стегосистемах часто используют особенности криптографических систем с открытым и (или) секретным ключом. Рассмотрим один такой пример.

Для обмена секретными ключами стегосистемы введем понятие протокола, реализованного на основе криптосистемы с открытыми ключами. Сначала Алиса генерирует случайную пару открытого и секретного ключа, а затем передает открытый ключ Бобу по скрытому каналу, созданному безключевой системой. Ни Боб, ни Вили, ведущий наблюдение за каналом, не могут определить, какая информация передавалась в скрытом канале: ключ или же случайные биты. Однако Боб может заподозрить, что стеганограмма от Алисы может содержать ее открытый ключ и постарается его выделить. После этого он шифрует с помощью выделенного ключа секретный стегоключ k, проводит сокрытие результата шифрования в контейнер и его передачу Алисе. Вили может попытаться извлечь секретную информацию из стеганограммы, но получит только случайный шифртекст. Алиса извлекает из стеганограммы скрытую криптограмму и расшифровывает ее своим секретным ключом. Таким образом, стороны обменялись секретным стегоключом k для совместного использования.

Рассмотренная стегосистема не лишена недостатков и приведена лишь в качестве примера смешанной системы.

1.1.2. Классификация методов компьютерной стеганографии

В современной стеганографии, в целом, можно выделить в направления: технологическую стеганографию и информационную стеганографию (рис 1.2).

Рис. 1.2. Классификация методов стеганографической защиты.

В рамках компьютерной стеганографии рассматриваются вопросы, связанные с сокрытием информации, которая хранится на носителях или передается по сетям телекоммуникаций, с организацией скрытых каналов в компьютерных системах и сетях, а также с технологиями цифровых водяных знаков и отпечатка пальца.

Существуют определенные отличия между технологиями цифровых водяных знаков и отпечатка пальца, с одной стороны, и собственно стеганографическими технологиями сокрытия секретной информации для ее последующей передачи или хранения. Самое главное отличие – это то, что

цифровые водяные знаки и отпечатки имеют целью защиту самого цифрового объекта (программы, изображения, музыкального файла и пр.), куда они внедряются, и обеспечивают доказательство прав собственности на данный объект.

1.2. Алгоритмы стеганографического сокрытия информации в графических файлах

1.2.1 Алгоритм HUGO

HUGO (Highly Undetectable steGO) – это стегоалгоритм, который оптимизирован для противодействия обнаружению против конкретного алгоритма стегоанализа SPAM [7]. Как показали результаты исследования авторов статьи [8], такой стегоалгоритм является так же сложным для обнаружения для некоторых других методов стегоанализа.

Основная идея HUGO заключается в том [9], что в процессе вложения, каждый пиксель изменяется с вероятностью обратно пропорциональной его “вкладу” в суммарное искажение изображения, который определяется стоимостью изменения пикселя.

Специальным образом выбирается матрица $\mathbb{H}$ размерности $k \times n$, где k – длина цепочки бит $\mathbf{M}$, которую требуется вложить, а n – полное количество отсчётов(пикселей) контейнера, затем решается уравнение[9]

$$\mathbb{H}\mathbf{Y} = \mathbf{M} \tag{1.1}$$

относительно вектора стего $\mathbf{Y}$, причём выбирается из множества решений (1) такое, которое при заданном $\mathbf{X}$ и $\rho_i = 1, \dots, n_1 \times n_2$ минимизирует суммарное искажение контейнера: $D(\mathbf{X}, \mathbf{Y}) = \sum_{i=1}^{n_1 \times n_2} \rho_i |x_i - y_i|$, где $\mathbf{X}$ – контейнер длинной n, $\mathbf{Y}$ – стегосигнал длиной n, ρ_i– цена изменения i - го пикселя с точки зрения стегоанализа. В простейшем случае она равна

единице для каждого пикселя. В этом случае, будет минимизировано количество измененных пикселей. Один из более сложных подходов, используемый в HUGO, является нахождение цены изменения исходя из Марковских вероятностей между соседними пикселями по 8 направлениям: в горизонтальных, вертикальных и диагональных направлениях. Для примера рассмотрим вычисления для горизонтального направления. Для остальных направлений, функционалы вычисляются похожим образом.

Пусть I изображение размера $n_1 \times n_2$. Разностный массив вычисляется по следующему правилу[9]:

$$D_{ij} = I_{ij} - I_{(i,j+1)}, \tag{1.2}$$

где $i \in \{1, \dots, n_1\}, j \in \{1, \dots, n_2 - 1\}$.

Затем, в зависимости от желаемого порядка функционалов, вычисляются:

функционалы первого порядка

$$\mathrm{C}_{\overrightarrow{d_1 d_2}} = Pr(D_{\overrightarrow{ij}} = d_1, D_{\vec{i},j+1} = d_2), \tag{1.3}$$

функционалы второго порядка

$$\mathrm{C}_{\overrightarrow{d_1 d_2 d_3}} = Pr(D_{\overrightarrow{ij}} = d_1, D_{\vec{i},j+1} = d_2, D_{\vec{i},j+2} = d_3), \tag{1.4}$$

где d_1, d_2, d_3 – расстояние, на которое отличаются соседние пиксели, варьируются в пределах $[-T, T]$,

T – параметр (в данном алгоритме используется T = 3).

На практике эффективнее использовать функционалы второго порядка, так как скрытность системы получается лучше, чем при использовании

функционалов первого порядка. Функционалы 1-го порядка, в свою очередь удобны для наглядности, при описании алгоритма.

Для противодействия SPAM стоимость изменения ρ_i не постоянна, а требует вычисления. Для этого функция искажения $D(\mathbf{X}, \mathbf{Y})$ определяется следующим образом [10]:

$$D(\mathbf{X}, \mathbf{Y}) = \sum_{d_1,d_2,d_3=-T}^{T} \left[w(d_1, d_2, d_3) \left| \sum_{k\in\{\rightarrow,\leftarrow,\uparrow,\downarrow\}} C_{d_1 d_2 d_3}^{\mathbf{X},k} - C_{d_1 d_2 d_3}^{\mathbf{Y},k} \right| + w(d_1, d_2, d_3) \left| \sum_{k\in\{\searrow,\nwarrow,\swarrow,\nearrow\}} C_{d_1 d_2 d_3}^{\mathbf{X},k} - C_{d_1 d_2 d_3}^{\mathbf{Y},k} \right| \right], \quad (1.5)$$

где $C_{d_1 d_2 d_3}^{\mathbf{X},k}$ – матрица смежности функционалов второго порядка, $w(d_1, d_2, d_3)$ – взвешивающая функция количественного обнаружения изменения матриц смежности, которая находится по следующей формуле [10]:

$$w(d_1, d_2, d_3) = \frac{1}{\left[\sqrt{d_1^2 + d_2^2 + d_3^2} + \sigma\right]^{\gamma}}, \quad (1.6)$$

где $\sigma, \gamma > 0$ – параметры, изменяя которые возможно минимизировать вероятность обнаружения.

Для того, что бы получить аддитивную меру искажений, примем стоимость изменения пикселя (i, j) как

$$\rho_{i,j} = D(\mathbf{X}, \mathbf{Y}^{(i,j)}), \quad (1.7)$$

где $\mathbf{Y}^{(i,j)}$ – стегосообщение, полученное при изменении одного пикселя контейнера $\mathbf{X}$, с координатами (i, j).[11]

Для упрощения решения задачи минимизации веса стегосигнала по сравнению с контейнером используется:[12]

1. Проверочная матрица $\mathbb{H}$ решётчатого кода разреженного вида (состоит из копий подматрицы $\mathbb{H}$ размера $h \times w$)(рис. 1.3)

$$\hat{\mathbb{H}} = \begin{pmatrix} 1\,0 \\ 1\,1 \end{pmatrix}^{\mathrm{h}}_{\mathrm{w}} \qquad \mathbb{H} = \begin{pmatrix} 1\,0 & & & & \\ 1\,1\,1\,0 & & & & \\ & 1\,1\,1\,0 & & & \\ & & 1\,1 & & \\ & & & \ddots & 1\,0 \\ & & & & 1\,1\,1\,0 \end{pmatrix}$$

Рис. 1.3. Пример построения матрицы $\mathbb{H}$.

2. Кодирование представляется в виде решетки [13], пример которой представлен на рис. 1.4. Между процедурой кодирования и решеткой имеется взаимно однозначное соответствие.

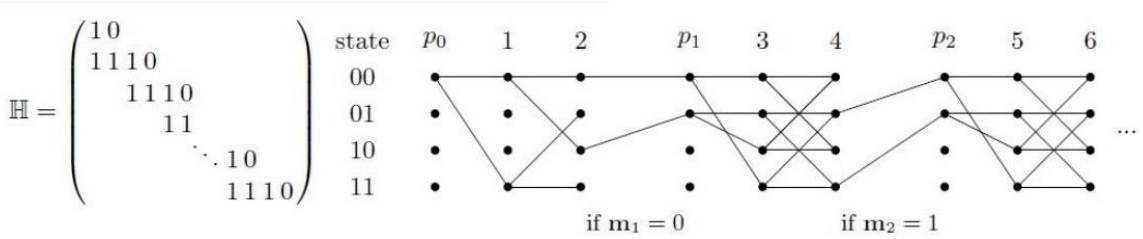


Рис. 1.4. Пример построения решётки.

Ключевой момент алгоритма Витерби состоит в том, что для каждой вершины на i-ом шаге имеется множество метрик (весов), соответствующих соединенным с ней ребрами вершинам на (i-1)-ом шаге, из которых можно оставить только одно ребро, которое минимизирует сумму весов на всех предыдущих шагах.

Данное направление перспективное и на сегодняшний день считается самым лучшим для высокоскоростных систем, однако строго доказательства, что оно будет строго секретным при различных методах стегоанализа нет.

1.2.2 Алгоритм WOW

WOW (Wavelet Obtained Weights) – стеганографический алгоритм, направленный на более эффективное скрытие информации в изображениях. Внедрение информации происходит в специально определенные места. После внедрения происходит небольшое изменение окресности точки изображения для большей скрытности алгоритма.

Суть алгоритма состоит в том что, вместо того, чтобы использовать взвешенную норму в некоторой стеганалитической модели для вычисления стоимости пикселей, используется набор направленных высокочастотных фильтров для получения так называемых направленных остатков, которые связаны с предсказуемостью пикселя в определенном направлении. Измеряя влияние внедрения на каждую направленную остаточную нагрузку и соответствующим образом агрегируя эти воздействия, стоимость внедрения остаётся высокой, когда содержание предсказуемо по крайней мере в одном направлении (гладкие области и вдоль краев) и низкой, где содержание непредсказуемо в каждом направлении (например, в текстурированных или шумных зонах). Таким образом, полученный алгоритм становится очень адаптивным и лучше сопротивляется стеганализу с использованием богатых моделей(SRM).

Вложение обрабатывается аддитивными искажениями в форме:

$$D(\mathbf{X}, \mathbf{Y}) = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \rho_{ij}\left(\mathbf{X}, Y_{ij}\right)\left|X_{ij} - Y_{ij}\right|, \qquad (1.8)$$

где ρ_{ij} - затраты на изменение пикселя X_{ij} на Y_{ij}. Аддитивность означает, что алгоритм не рассматривает влияние отдельных изменений внедрения, влияющих друг на друга. Определив затраты на пиксель ρ_{ij}, вложение последовательности бит с минимальным ожидаемым искажением (8) эквивалентно исходному кодированию с критерием верности.

Функция искажения HUGO концентрирует влияние вложения, в основном в текстурах и краях. Однако содержание по краю обычно хорошо

моделируется с использованием локально полиномиальных моделей, что помогает обнаружить [13, 14, 15]. Таким образом, по возможности алгоритм встраивания должен встраиваться в текстурированные/шумные области, которые нелегко модифицируются в любом направлении. С этой целью при встраивании оценивается гладкость в нескольких направлениях с использованием набора фильтров $\mathcal{B}_n = \{\mathbf{K}^{(1)}, \dots, \mathbf{K}^{(n)}\}$, состоящий из n многонаправленных высокочастотных фильтров, представленных их ядрами, нормированными так, что все L_2-нормы $\|\mathbf{K}^{(k)}\|_2$ одинаковы. k-й остаток $R^{(k)}, k = 1, \dots, n$ вычисляется как $R^{(k)} = \mathbf{K}^{(k)} * \mathbf{X}$, где «⋆» является сверточным зеркальным дополнением, так что $R^{(k)}$ снова имеет $n_1 \times n_2$ элементов. (Зеркальное заполнение предотвращает введение вставки артефактов на границе изображения.) Если остаточные значения $R_{ij}^{(k)}$ велики для некоторого ij и для всех k, это означает, что локальное содержимое в пикселе x_{ij} не является гладким в любом направлении и, следовательно, его сложно моделировать. Поскольку необходимо обнаружить ребра во всех направлениях, естественно использовать предустановленные ребра-детекторы для набора фильтров. Ненаправленный фильтр **KB** равный[16]

$$\mathbf{K}^{(1)} = \begin{pmatrix} -1 & 2 & -1 \\ 2 & -4 & 2 \\ -1 & -2 & -1 \end{pmatrix}$$

часто используется в стеганализе, тогда как оператор Собеля является общим детектором кромок.

$$\mathbf{K}^{(1)} = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}, \mathbf{K}^{(2)} = \left(\mathbf{K}^{(1)}\right)^T$$

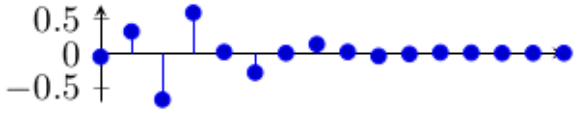
Вейвлет-ориентированные наборы фильтров WDFB-H и WDFB-D (рисунок 4) используют вейвлет-сигналы Haar и Daubechies 8-tap.

WDFB-H $\mathbf{h}$ = Haar wavelet decomp. low-pass

$\mathbf{g}$ = Haar wavelet decomp. high-pass

WDFB-D $\mathbf{h}$ = Daubechies 8 wavelet decomp. low-pass

$\mathbf{g}$ = Daubechies 8 wavelet decomp. high-pass



$$\mathbf{K}^{(1)} = \mathbf{h} \cdot \mathbf{g}^{\mathrm{T}},\ \mathbf{K}^{(2)} = \mathbf{g} \cdot \mathbf{h}^{\mathrm{T}},\ \mathbf{K}^{(3)} = \mathbf{g} \cdot \mathbf{g}^{\mathrm{T}}$$

Рис. 1.5. Вейвлет-ориентированные наборы фильтров WDFB-H и WDFB-D.

Вычисление остатка совпадает с декомпозицией вейвлета первого уровня без прореживания. Вейвлет-наборы состоят из трех фильтров: $\mathbf{K}^{(1)}$, $\mathbf{K}^{(2)}$, $\mathbf{K}^{(3)}$ с использованием которых получены LH, HL и HH остатки. При использовании одномерного фильтра низкочастотного декомпозиции вейвлета h и фильтра декомпозиции высоких частот g 2-D-направленные фильтры рассчитываются, как показано на рис. 1.5.

При вложении изменяются большие значения направленных остатков, там где текстуры и ребра, и сохраняются небольшие значения, где содержание предсказуемо. Одним из способов достижения этого является взвешивание разницы между $\mathbf{R}^{(k)}$ и тем же остатком после изменения только одного пикселя в ij (обозначаемого $\mathbf{R}^{(k)}_{[ij]}$) самим коэффициентом вейвлет:

$$\xi_{ij}^{(k)} = \left|\mathbf{R}^{(k)}\right| * \left|\mathbf{R}^{(k)} - \mathbf{R}^{(k)}_{[ij]}\right| \curvearrowleft \overline{\overline{(a)}}\left|\mathbf{R}^{(k)}\right| * \left|\mathbf{K}^{(k)}\right| \curvearrowleft. \qquad (1.9)$$

Величина $\xi_{ij}^{(k)}$, которая называется вложением «пригодности», является формально корреляцией между абсолютной величиной остатка покрытия с абсолютным значением остаточного изменения. Поскольку $\mathbf{R}^{(k)} - \mathbf{R}^{(k)}_{[ij]}$-

пространственно сдвинутый направленный фильтр $\mathbf{K}^{(k)}$, величина $\xi_{ij}^{(k)}$, может быть вычислена для всех пикселей сразу (равенство (а)). Затем вычисляются затраты на вложение $\rho_{i,j}$ путем агрегирования всех соответствий $\xi_{ij}^{(k)}$, $k = 1, \dots, n$. Поскольку необходимо ограничить внесение изменений в эти пиксели сложным содержанием во всех направлениях, то правилу агрегации ρ: $\mathbb{R}^n \longrightarrow \mathbb{R}_0^+$, $\rho_{ij} = \rho(\xi_{ij}^{(1)}, \dots, \xi_{ij}^{(n)})$ требуются следующее свойства:

1. Чем больше значения $\left|\xi_{ij}^{(k)}\right|$, тем меньше должна быть ρ_{ij}.

2. Если существует $k \in \{1, \dots, n\}$ такие, что $\xi_{ij}^{(k)} = 0$, тогда $\rho_{ij} = +\infty$.

Простой функцией, отвечающей обоим требованиям, является обратная норма Гёльдера с $p < 0$:

$$\rho_{ij}^{(p)} = \left(\sum_{k=1}^{n} \left|\xi_{ij}^{(k)}\right| p\right)^{\frac{1}{p}}.$$

Стоит ограничивать изменения вложения на ± 1, $\left|X_{ij} - Y_{ij}\right|$.

1.2.3 Алгоритм UNIWARD

UNIWARD (universal wavelet relative distortion) - стеганографический алгоритм, основанная на наработках HUGO и WOW, использующая наиболее зашумлённые места в изображении для улучшения скрытности алгоритма.

Для данного изображения $\mathbf{X}$, представленного в пространственной области, оценивается его гладкость в разных направлениях с помощью 8-кратного набора вейвлет-направленных фильтров Добеши (D-WDFB), $\mathcal{B} = \left\{\mathbf{K}^{(1)}, \mathbf{K}^{(2)}, \mathbf{K}^{(3)}\right\}$ состоящий из высокочастотных фильтров LH, HL и HH (ядер *K*). Эти три фильтра построены из одномерных низкочастотных (h) и высокочастотных (g) фильтров декомпозиции, показанных на рис. 1.5:

$$\mathbf{K}^{(1)} = h \cdot g^{\mathrm{T}}, \mathbf{K}^{(2)} = g \cdot h^{\mathrm{T}}, \mathbf{K}^{(3)} = g \cdot g^{\mathrm{T}}. \quad (1.10)$$

Из рисунка 4 видно, что поддержка каждого одномерного фильтра равна 16, что дает ядрам размер 16 × 16. Определим k-ое направленное остаточное значение как $\mathbf{R}^{(k)} = \mathbf{K}^{(k)} * \mathbf{X}, k = 1,2,3$, где * - зеркальная свертка, которая дает каждому $\mathbf{R}^{(k)}$ размерность $n_1 \times n_2$. Цель зеркального заполнения заключается в том, чтобы предотвратить введение вставки артефактов на границе изображения. Также стоит обратить внимание на то, что наклонные остатки представляют собой, по существу, 2-х неразрешенные вейвлеты LH, HL и HH направленного разложения **X**. Причина выбора этого набора фильтров для построения UNIWARD приведена в [11], где среди всплесков Добеши, авторы изучили несколько различных наборов фильтров, в том числе краевой детектор Sobel, ненаправленные ядра и вейвлеты Хаара. Так как вейвлеты Добеши дали последовательно лучшие результаты,этот набор используют чаще всего.

Дана пара контейнер-изображений, X и Y, пустой и заполненный соответственно, обозначим $W_{uv}^{(k)}(\mathbf{X})$ и $W_{uv}^{(k)}(\mathbf{Y})$ uv- й вейвлет-коэффициент в k-м разложении, полученным с использованием ядер (10). Если **X** и **Y** являются изображениями JPEG, они сначала декомпрессируются в пространственную область, а затем применяются вейвлет-преобразования. Искажение между обоими изображениями представляет собой сумму относительных изменений вейвлет-коэффициентов w.r.t. контейнер-изображения:

$$D(\mathbf{X}, \mathbf{Y}) \triangleq \sum_{k=1}^{3} \sum_{u,v} \frac{\left|W_{uv}^{(k)}(\mathbf{X}) - W_{uv}^{(k)}(\mathbf{Y})\right|}{\varepsilon + \left|W_{uv}^{(k)}(\mathbf{X})\right|}, \quad (1.11)$$

где сумма по uv берется по всем $n_1 \times n_2$. коэффициентам поддиапазонов, а $\varepsilon > 0$ является стабилизирующей константой, чтобы

избежать деления на ноль. Безопасность внедрения с использованием UNIWARD довольно нечувствительна к точному значению ε.

Чтобы понять логику этого определения, нужно понять, что отношение в (1.11) меньше при изменении большого вейвлет-коэффициента контейнера, которое произойдет в текстурах / шумных областях и вблизи краев. С другой стороны, если хотя бы один небольшой коэффициент, который изменяется на сравнительно большую величину, величина искажения также окажется большой. Таким образом, (1.11) препятствует внесению изменений в регионы, где содержание является гладким (и, следовательно, его можно моделировать) по меньшей мере в одном направлении.

В общем, под боковым информированным внедрением понимается любой метод, в котором отправитель имеет более качественную версию доступного контейнера (так называемый «предварительный»). Исторически первым методом, который использовал предварительный тест, был алгоритм Embedding by Dithering [17], в котором на его входе было внедрено изображение с истинным цветом, чтобы преобразовать изображение в 256-цветную палитру GIF. Термин «предел» происходит от Кера [18]. В частности, в области JPEG предварительная проверка достигает формы неквантизированных ДКП-коэффициентов D_{ij}, полученных из необработанного предварительного изображения **P**. В этом случае встраиватель может выбирать около D_{ij} «вверх» или «вниз» для модуляции его четности (например, младший значащий бит округленного значения). При сжатии предварительного **P** на изображение-контейнер **X** погрешность округления для ij-х ДКП-коэффициентов равна

$$e_{ij} = \left|D_{ij} - X_{ij}\right|, e_{ij} \in [0,0.5]. \quad (1.12)$$

При округлении «с другой стороны» отправитель вводит следующее изменение вложения

$$Y_{ij} = X_{ij} + sign(D_{ij} - X_{ij}), \tag{1.13}$$

Что соответствует «погрешности округления»$1 - e_{ij}$. Поэтому каждое изменение вложения увеличивает искажение w.r.t. За счет разницы между ошибками округления:

$$\left|D_{ij} - Y_{ij}\right| - \left|D_{ij} - X_{ij}\right| = 1 - 2e_{ij}.$$

Таким образом, естественно определить искажение для стороннего внедрения в JPEG-области как разность:

$$D^{(\mathrm{SI})}(\mathbf{X}, \mathbf{Y}) \triangleq D(\mathbf{P}, \mathbf{Y}) - D(\mathbf{P}, \mathbf{X}) = \sum_{k=1}^{3} \sum_{u,v} \frac{\left|W_{uv}^{(k)}(\mathbf{P}) - W_{uv}^{(k)}(\mathbf{Y})\right| - \left|W_{uv}^{(k)}(\mathbf{P}) - W_{uv}^{(k)}(\mathbf{X})\right|}{\varepsilon + \left|W_{uv}^{(k)}(\mathbf{P})\right|}. \tag{1.14}$$

Стоит отметить, что линейность ДКП и вейвлет-преобразований гарантирует, что $D^{(\mathrm{SI})}(\mathbf{X}, \mathbf{Y}) \geq 0$. Это связано с тем, что округление ДКП-коэффициента для получения покрытия X соответствует добавлению некоторого двумерного шаблона 8 × 8 в пространственной области, который зависит от модифицированного режима ДКП и, следовательно, шаблон 23 × 23 в области вейвлет, поскольку поддержка 8-кратных всплесков Добеши в 16 × 16. С другой стороны, округление «до другая сторона» для получения stego-изображения **Y** соответствует вычитанию одного и того же шаблона, но с большей амплитудой, поэтому $\left|W_{uv}^{(k)}(\mathbf{P}) - W_{uv}^{(k)}(\mathbf{Y})\right| - \left|W_{uv}^{(k)}(\mathbf{P}) - W_{uv}^{(k)}(\mathbf{X})\right| \geq 0$.

Уравнение (1.14) имеет некоторое сходство с искажением, используемым в предложенном нормированном возмущенном квантовании (NPQ) [20]. Там авторы также предложили вычислить искажение вложения

как относительное изменение коэффициентов ДКП покрытия. UNIWARD отличается от функции искажения NPQ тем, что мы вычисляется искажение с использованием набора направленных фильтров в области вейвлет, что придает очень важную чувствительность к встраиванию - направленность и, следовательно, потенциально лучшую адаптивность содержания.

Кроме того, искажение (1.11) построено так же, как и искажение вложения, используемое в WOW [20], оно также способно оценивать локальное содержимое контейнера с использованием направленных невязок, вычисленных в области вейвлета.

Обратите внимание, что оба (1.11) и (1.14) не являются аддитивными, потому что изменение пикселя X_{ij} будет влиять на 16 × 16 окрестности коэффициентов вейвлетов (размер поддержки вейвлета 8-кратных зондов Добеши). Для изображений, представленных в области JPEG, изменение коэффициента JPEG X_{ij}будет влиять на блок размером 8 × 8 пикселей и, следовательно, 23 × 23 вейвлет-коэффициентов. Поэтому при изменении соседних пикселей (или коэффициентов ДКП) шаблоны внедрения перекрываются и изменения «взаимодействуют», вызывая неаддитивность *D*. Даже если существуют методы для встраивания с использованием неаддитивных функций искажения (например, конструкция Гиббса [21]), реализация внедрения с использованием аддитивных искажений значительно проще. Более того, в случае UNIWARD кажется, что взаимодействия между соседними изменениями вложений достаточно сильны, чтобы на практике построение Гиббса было неэффективным. Конструкция Гиббса способна встраивать так называемую энтропию стирания, но с искажением, соответствующим фактической энтропии марковского поля. Чем сильнее взаимодействие между изменениями внедрения, тем больше разница между обеими энтропиями, и тем менее эффективна конструкция Гиббса.

Как показано в [23], любая функция искажения $D(\mathbf{X}, \mathbf{Y})$ может быть использована для вложения в её так называемую аддитивную аппроксимацию с помощью D для вычисления стоимости изменения

коэффициента каждого пикселя/ДКП-коэффициента. В частности, стоимость, ρ_{ij}, изменения X_{ij} на Y_{ij} при оставлении всех остальных элементов покрытия неизменна:

$$\rho_{ij}(\mathbf{X}, Y_{ij}) \triangleq D(\mathbf{X}, X_{\sim ij}Y_{ij}), \tag{1.15}$$

где $X_{\sim ij}Y_{ij}$ - контейнер $\mathbf{X}$, только с изменением ее ij-го элемента: $X_{ij} \longrightarrow Y_{ij}$. Заметим, что $\rho_{ij} = 0$, когда $\mathbf{X} = \mathbf{Y}$. Будем обозначать аддитивные приближения к (1.11) и (1.14) с индексом «А.» Например, аддитивное приближение к $D(\mathbf{X}, \mathbf{Y})$

$$D_A(\mathbf{X}, \mathbf{Y}) = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \rho_{ij}(\mathbf{X}, Y_{ij}) \left[X_{ij} \neq Y_{ij}\right] \tag{1.16}$$

Заметим, что наличие абсолютных значений в $D(\mathbf{X}, \mathbf{Y})$ (1.11) влечет

$$\rho_{ij}(\mathbf{X}, X_{ij} + 1) = \rho_{ij}(\mathbf{X}, X_{ij} - 1), \forall i, j \text{ и } X_{ij}, \tag{1.17}$$

Что позволяет использовать трехмерную операцию вложения для пространственных и JPEG-областей. Практические алгоритмы внедрения могут быть построены с использованием тройной многоуровневой версии STC (раздел IV в [23]). Можно, по-видимому, справедливо утверждать, что стоимость внедрения должна зависеть от полярности изменения, однако равная стоимость обоих возможных изменений определяется функцией искажения. Более того, поскольку UNIWARD ограничивает внесение изменений в текстуры, потенциальный недостаток наличия равных затрат для обеих полярностей снижается, и это позволяет нам уменьшить искажение внедрения для фиксированной полезной нагрузки за счет использования более сильных троичных кодов. Это, как ожидается, станет особенно выгодным для больших полезных нагрузок. Наконец, обратите внимание, что

(SI) для стеганографии JPEG со стороны, D A (X, Y) по своей сути ограничивается бинарной операцией встраивания, поскольку отправитель имеет только два варианта: либо округление X ij вверх, либо вниз.

Методы внедрения, которые используют аддитивные аппроксимации UNIWARD для пространственного, JPEG и сторонней области JPEG, называются S-UNIWARD, J-UNIWARD и SI-UNIWARD, соответственно.

1.3. Методы выявления стеганографически скрытой информации

1.3.1. Основные принципы стегоанализа.

Даже при оптимальных условиях для атаки задача извлечения скрытого сообщения из контейнера может оказаться очень сложной. Однозначно утверждать о факте существования скрытой информации можно только после ее выделения в явном виде. Иногда целью стеганографического анализа является не восстановление алгоритма вообще, а поиск, например, конкретного стеганоключа, используемого для выбора битов контейнера в стеганопреобразовании.

Основной целью стеганоанализа является моделирование стеганографических систем и их исследование для получения качественных и количественных оценок надежности использования стеганопреобразования, а также построение методов выявления скрываемой в контейнере информации, ее модификации или разрушения.

Одной из главных задач стеганоанализа является исследование возможных следов применения стеганографических средств и разработка методов, которые позволяли бы обнаруживать факты их использования. Применение конкретного стеганографического преобразования требует от стеганоаналитика индивидуального подхода к его исследованию.

Исследование сообщений, скрытых одним из множества существующих стеганографических методов, процесс довольно трудоемкий. Для успешного проведения стеганоанализа необходимо:

- иметь для анализа стеганосредство, с помощью которого осуществляется скрытие сообщений;
- иметь возможность восстанавливать используемые в системе стеганографический и, возможно, криптографический алгоритмы; выполнять их экспертный анализ и разрабатывать алгоритм определения ключей;
- иметь возможность использовать для проведения стеганоанализа вычислительные ресурсы необходимой мощности;
- поддерживать на должном уровне теоретические и практические знания в области компьютерной стеганографии.

Выделяют следующие направления практического развития стеганографического анализа:

1) Разработка вероятностно-статистических методов распознавания, применение элементов искусственного интеллекта для получения оценок надежности стеганографических преобразований, а также при создании детекторов (фильтров) – для анализа информационных потоков с целью обнаружения и перекрытия скрытых каналов связи (рис. 1.5). В таком случае проверка наличия скрытой информации сводится к определенной оценке с использованием статистических критериев (последовательной корреляции, энтропии изображения, дисперсии младшего бита и т. д.) – это называется пассивным стегоанализом. Разрабатываемые с этой целью средства должны не только обеспечивать низкий уровень погрешности во время распознавания скрытых сообщений (особенно в тех случаях, когда используется предварительное шифрование), но и быть универсальными, то есть должна существовать возможность детектирования сообщений встроенных разными стеганографическими методами.

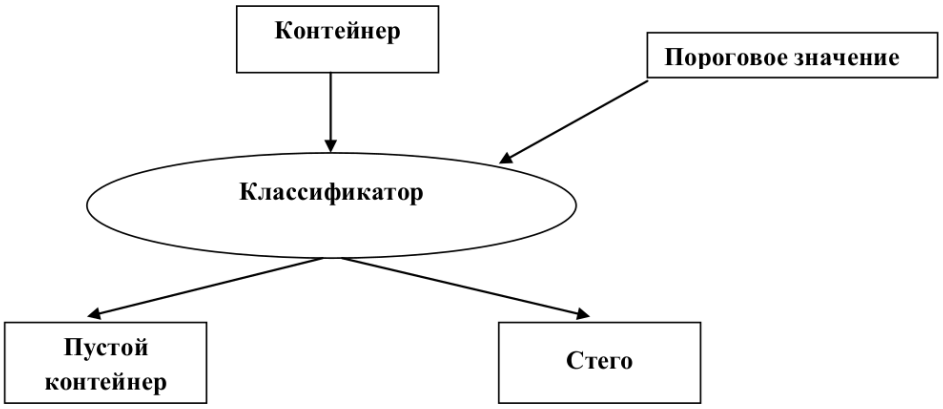


Рис. 1.6. Общая схема пассивного стегоанализа.

2) Анализ конкретных программных стеганографических средств с целью восстановления алгоритмов и разработки оптимальных методов их исследования. Основная сложность в данном случае заключается в большой трудоемкости, обусловленной необходимостью индивидуального подхода к каждому конкретному алгоритму, реализующему тот или иной метод скрытия информации, а также значительным объемом вычислений, необходимых для восстановления стеганоключей.

3) Разработка технологий активных и злонамеренных атак для внесения невосстанавливаемых искажений в предполагаемую стеганограмму с целью спровоцировать ее повторную передачу в другом контейнере, что подтвердило бы факт использования стеганосредств.

1.4. Статистические методы выявления стеганографического скрытия информации

1.4.1 Гистограммный анализ

Гистограммный анализ, также известный как статистический анализ пар значений, был предложен Андреасом Фитцманом и Андреасом Вестфелдом [24] для детектирования метода LSB. Они отметили, что стеганографические системы, основанные на последовательной замене

наименее значимых бит, порождают определенного рода искажения, которые могут быть выявлены с помощью классификатора. В процессе внедрения информации с высокой энтропией (т.е. свысокой мерой неопределенности) происходит предсказуемое изменение гистограммы распределения частот [25].

Пусть в качестве контейнера стеганографической системы используются цифровые фотографии, с каждым пикселем которого связан индекс i в таблице цветов. Обозначим через n_i и n_i^* частоты появления цвета с индексом i до и после внедрения. Предположим, что встраиваемое сообщение имеет равномерное распределение. Например, сообщение предварительно было зашифровано. Тогда если выполняется неравенство:

$$n_{2i} > n_{2i+1}, \tag{1.18}$$

то пиксели с цветом $2i$ в процессе внедрения будут меняться чаще, чем пиксели с цветом $(2i+1)$. Это приводит к очевидному результату:

$$|n_{2i} - n_{2i+1}| > |n_{2i}^* - n_{2i+1}^*|. \tag{1.19}$$

Другими словами, внедрение равномерно распределенного сообщения уменьшает разницу между частотами распределения соседних цветов, имеющих различие в наименьшем бите.

Этот подход может быть применен не только к цифровым изображениям с индексацией цвета, но и для формата JPEG [26]. В этом случае вместо индексов цвета необходимо использовать коэффициенты ДКП.

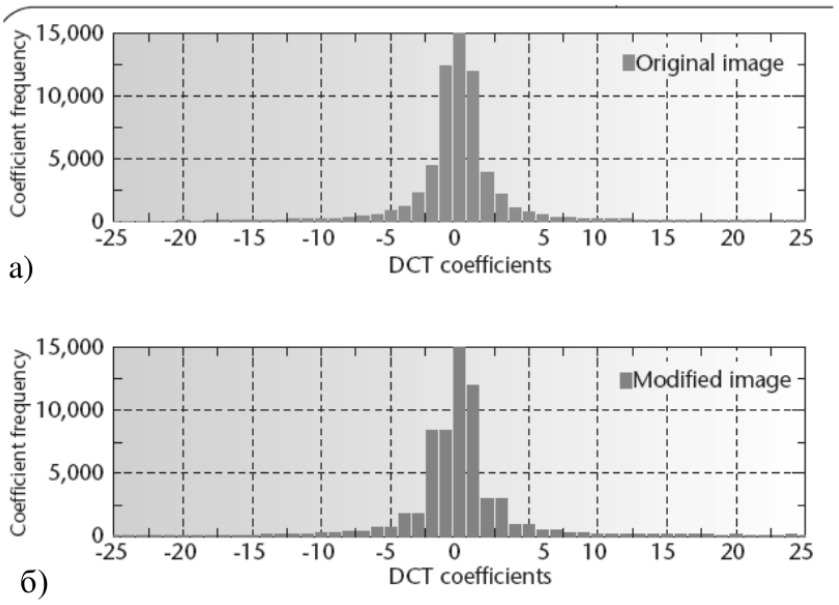


Рисунок 1.7. Гистаграммы распределения коэффициентов ДКП для пустого (а) и заполненного (б) контейнеров.

Для того чтобы определить факт изменения гистограммы распределения частот, Фитцман и Вестфелд предложили использовать критерий согласия Хи-квадрат [27]. Так как в процессе внедрения методом LSB сумма распределения частот соседних пар остается неизменной, то ожидаемое распределение может быть получено в виде среднего арифметического значений соседних пар:

$$y_i^* = \frac{y_{2i} - y_{2i+1}}{2}. \quad (1.20)$$

Величина Хи-квадрат для сравнения ожидаемого распределения и распределения исследуемой последовательности имеет вид:

$$\chi^2 = \sum_{i=1}^{\nu} \frac{y_i - y_i^*}{y_i^*}, \quad (1.21)$$

где ν – число степеней свободы, равное количеству столбцов гистограммы минус один.

Вероятность p, что два распределения окажутся одинаковыми, определяется следующим образом:

$$p = \int_o^{\chi^2} \frac{t^{\nu-2} e^{-\frac{t}{2}}}{2^{\nu/2}\Gamma(\frac{\nu}{2})},$$

где Г – гамма функция Эйлера.

Таким образом, чем больше значение p, тем выше вероятность наличия скрытого сообщения в исследуемом контейнере. Гистограммный анализ позволяет не только выявить факт использования алгоритма LSB, но и определить границы сообщения внутри контейнера. Авторами гистограммного анализа было предложено проводить серию измерений вероятности p для фрагментов цифрового изображения. Каждый фрагмент представляет собой последовательность пикселей, идущих подряд с начала изображения.

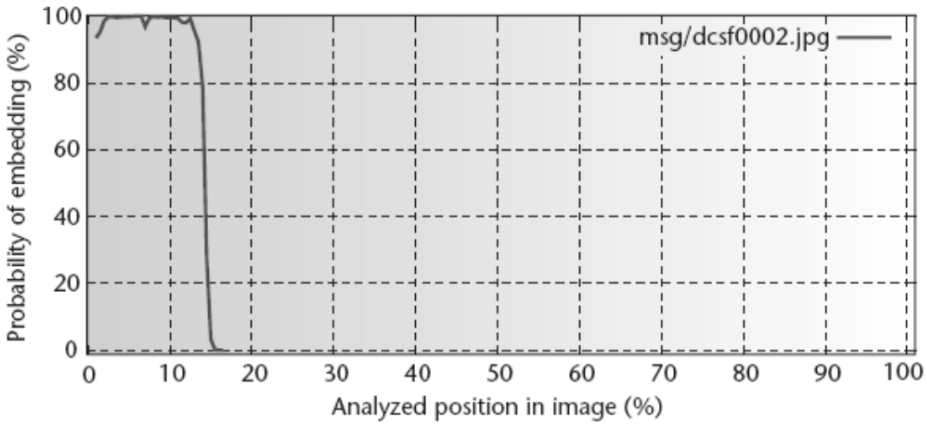


Рисунок 1.8. Анализ заполненного контейнера с помощью критерия согласия Хи-квадрат.

Предложенный авторами [28] метод можно отнести к методам стегоанализа, направленным против определенного поведения стеганографической системы. Другими словами, гистограммный анализ основан на недостатках метода LSB, эксплуатируя тем самым принцип Керкгоффса.

1.4.2 RS-анализ

Другим оригинальным методом статистического анализа является метод RS-анализа [30,31]. Данный метод был предложен в 2001 году коллективом исследователей из Бингемтонского университета под руководством Джессики Фридрих. Сокращение в название расшифровывается как Regular-Singular, то есть «регулярный-сингулярный».

В отличие от гистограммного анализа, RS-анализ основан на выявлении пространственной корреляции в пустых контейнерах [32,33].

Пусть изображение имеет размерность MxN пикселей, которые принимают значения из множества P. Например, в случае полутонового изображения, P = {0,...,255}. Изображение разбивается на связанные группы по n пикселей, $G = (x_1, \dots, x_n)$. На группах G определяется вещественная функция регулярности, или «гладкости», f(G).

В качестве такой функции принято выбирать сумму перепадов соседних пикселей:

$$f(G) = \sum_{i=1}^{n} |x_{i+1} - x_i|. \quad (1.22)$$

Чем больше значение f(G), тем более зашумленной является группа G.

Функция F(x) называется флиппингом, если F(F(x))=x. Флиппинг, таким образом, представляет собой двух тактовую перестановку. Определяются следующие функции флиппинга, F_1 – инверсия младшего бита, F_{-1} – инверсия с переносом в старший бит:

$$F_1: 0 \leftrightarrow 1, 2 \leftrightarrow 3, \dots, 254 \leftrightarrow 255 \quad (1.23)$$

$$F_{-1}: 255 \leftrightarrow 0, 1 \leftrightarrow 2, \dots, 253 \leftrightarrow 254 \quad (1.24)$$

Все группы изображения делятся на три типа групп: регулярные, сингулярные, неиспользуемые. Регулярные группы – это группы, для которых функция флиппинга увеличивает значение гладкости, $G \in R \leftrightarrow f(F(G)) > f(G)$. Сингулярные группы – это группы, для которых функция флиппинга уменьшает значение гладкости $G \in S \leftrightarrow f(F(G)) > f(G)$. Наконец, неиспользуемые группы – это группы, значение гладкости которых не меняется после применения функции флиппинга, $G \in U \leftrightarrow f(F(G)) > f(G)$.

Здесь F(G) означает применение флиппинга F к компонентам вектора G =(x_1,...,x_n). Можно применять разные функции флиппинга к разным пикселям внутри группы. Выбор функции флиппинга можно записать в виде маски M, которая представляет собой n-размерный вектор в пространстве {-1,0,1}. Тогда в результате применения флиппинга с маской M получается группа:

$$F(G) = (F_{M(1)}(x_1), F_{M(2)}(x_2), \dots, F_{M(n)}(x_n)) \quad (1.25)$$

Целью флиппинга F является искажение значений пикселей в обратимом порядке, тем самым моделируя процесс добавления шума. Для естественных изображений добавление небольшого шума ведет к возрастанию функции гладкости. Поэтому число регулярных групп будет больше числа сингулярных групп. Эта разница позволяет внедрять потенциально большое сообщение.

Пусть R_M– число регулярных групп для маски M (в процентах от всего количества групп). Аналогично, S_M – число сингулярных групп. Имеем $R_M + S_M \leq 1$ и $R_{-M} + S_{-M} \leq 1$, где -M – инвертированная маска. Статистическая гипотеза RS-анализа заключается в том, что для естественного изображения (пустого контейнера) число регулярных и сингулярных групп не меняется при инвертировании маски:

$$R_M \cong R_{-M}, S_M \cong S_{-M}. \quad (1.26)$$

Данная гипотеза основана на зависимости функции флиппинга: $F_{-1} = F_1(x+1) - 1$. Применение F_{-1} аналогично применению F_1 к изображению, значения пикселей которого были сдвинуты на единицу. Для естественного изображения не существует причины, из-за которой сдвиг значений пикселей на единицу приведет к заметному изменению числа регулярных и сингулярных групп.

Группой исследователей из Бингемтонского университета было получено экспериментальное подтверждение статистической гипотезы на выборке из цифровых фотографий [29]. В то же время было отмечено, что внесение случайных искажений в наименее значимые биты, нарушает данное соотношение.

Случайные искажения уменьшают разницу между R_M и S_M с увеличением длины внедряемого сообщения. Это можно объяснить тем, что разница между этими значениями характеризует потенциальный объем сообщения для внедрения. С другой стороны, искажения увеличивают разницу между R_{-M} и S_{-M}.

В процессе анализа изображения строится RS-диаграмма. На оси абсцисс откладывается процент количество инвертированных бит x, на оси ординат – относительные значения регулярных и сингулярных групп в процентах от общего числа групп. Если предположить, что длина сообщения p и при записи были изменены 50% бит, то получится статистика в точке p/2.

Если теперь инвертировать все младшие биты изображения и пересчитать статистики, получится статистика в точке 100-p/2. Заполнение наименее значимых бит изображения случайными значениями даст точку с абсциссой 50%. В данной точке пересекаются кривые R_M и S_M. Приняв p/2 за 0, 100-p/2 за 1 и аппроксимируя кривые R_M и S_M параболами, а кривые R_{-M} и S_{-M} прямой, получим координаты точки пересечения R_M и S_M . Пусть значение абсцисс равно x. Тогда длина сообщения p вычисляется по формуле:

$$p = \frac{x}{x-0,5}. \quad (1.27)$$

Таким образом, выходным значением RS-анализа является предполагаемая длина сообщения в исследуемом изображении. Затем это значение может быть сравнено с некоторым пороговым значением для получения маркировки классификатором.

RS-анализ был применен для детектирования метода LSB при использовании ряда стеганографических систем, таких как Steganos, Windstorm, STools, Hide4PGP. Ошибка классификации составила менее 1% [30].

Предложенный исследователями Бингемтонского университета метод стегоанализа направлен на обнаружение закономерностей в естественных изображениях, в отличие от гистограммного анализа.

1.5. Использование машинного обучения для стегоанализа

В качестве ответной меры на появление алгоритмов Outguess и F5, так как те основывались на минимизации вносимого внедрением искажения, в 2002 году Сьюви Лью и Хани Фарид представили новое направление в стегоанализе – метод машинного обучения [31,32]. Их подход заключается в построение сложных статистических моделей для естественных изображений и в нахождении устойчивых закономерностей.

Машинное обучение – обширный подраздел искусственного интеллекта, изучающий методы построения алгоритмов, способных обучаться. Общая постановка задачи звучит следующим образом. Имеется множество объектов (ситуаций) и множество возможных ответов (откликов, реакций). Между объектами и ответами существует зависимость, которую необходимо выявить. Известна совокупность «прецедентов» - пар вида

«объект, ответ», называемая обучающей выборкой. Эти данные используются для построения алгоритма, способного для любого объекта выдать достаточно точный ответ.

Лью и его коллега для классификации использовали метод опорных векторов. Суть метода заключается в том, что каждый объект представляет собой радиус-вектор (точку) в многомерном линейном пространстве. Каждый объект принадлежит одному из двух классов. В процессе обучения вычисляется гиперплоскость, которая разделяет объекты разных классов.

В качестве представления изображения в линейном пространстве используется вектор признаков, который вычисляется на основании внутренних закономерностей. Например, Фарид предложили использовать статистики распределения групп пикселей, такие как математическое ожидание, дисперсия, среднеквадратичное отклонение и т.д. Пиксели группировались по вертикали, горизонтали и диагонали. Также, пиксели подвергались различным фильтрам и преобразованиям. В результате, был получен 72-х размерный вектор признаков [32].

Существует три разновидности метода опорных векторов: случай линейной разделимости, случай нелинейной разделимости и метод с заменой ядра, которые различаются ресурсоемкостью обучения и точностью классификации.

Пусть пара $(x_i, y_i), i = 1, \dots, N$– объект из обучающей выборки, где x_i – вектор признаков, y_i принимает значение 1 для пустых контейнеров и -1 для стего. В случае линейной разделимости строится гиперплоскость, которая разграничивает положительные и отрицательные экземпляры. Точки, лежащие на гиперплоскости, удовлетворяют выражению:

$$w^t x_i + b = 0, \tag{1.28}$$

где w – нормаль гиперплоскости,$b/\|w\|$ – расстояние от начала координат до гиперплоскости.

Из множества допустимых гиперплоскостей выбирается та, у которой зазор между положительными и отрицательными экземплярами максимален.

Если гиперплоскость существует, тогда выполняется условие:

$w^t x_i + b \geq 1$ при $y_i = 1$ и $w^t x_i + b \leq -1$ при $y_i = -1$.

При этом размер зазора определяется как $2/\|w\|$. Таким образом, задача сводится к минимизации $\|w\|^2$. Задача оптимизации решается через лагранжиан:

$$L(w, b, a) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} \alpha_i (w^t x_i + b) y_i + \sum_{i=1}^{N} \alpha_i. \qquad (1.29)$$

Полученные в результате значения w и b могут быть использованы для классификации нового экземпляра z, достаточно определить в каком полупространстве относительно гиперплоскости лежит соответствующий вектор признаков. Если $w^t x_i + b \geq 0$, то экземпляр классифицируется как стего. В противном случае – как пустой контейнер.

В некоторых случаях не существует гиперплоскости, способной полностью разделить экземпляры разных классов (рисунок 1.7, б). Некоторые экземпляры могут «лежать на неправильной стороне». В этом случае вводят мягкие границы: $w^t x_i + b \geq 1 - \varepsilon_i$ при $y_i = 1$ и $w^t x_i + b \geq -1 + \varepsilon_i$ при $y_i = -1$. При решении добавляется условие минимизации суммарной ошибки обучения, $\sum_{i=1} \varepsilon_i \to \min$.

В некоторых случаях эффективно разделить экземпляры нелинейной

гиперплоскостью. В этом случае применяется универсальный метод опорных векторов с заменой ядра. Здесь вместо скалярного произведения векторов используется стеганографии нелинейная функция ядра. В качестве ядра в цифровой принято выбирать радиальную базисную функцию: $(x, y) = e^{-\gamma\|x-y\|}, \gamma > 0$. Таким образом, вводится дополнительный оптимизационный параметр, который ищется перебором на заданном отрезке. Использование

замены ядра приводит к лучшей эффективности разделения классов, но требует больших ресурсоемких вычислений.

Лью и Фарид использовали метод машинного обучения на обучающей выборке из 1800 пустых контейнеров и случайного подмножества из 1800 стего. Используя 72-х размерный вектор признаков, они получили следующие результаты: для алгоритма LSB точность классификации составила 99%, для алгоритма Outguess – 95,6% [32].

Эффективность классификации зависит от выбора пространства признаков. Исследователи предложили несколько различных пространств признаков высокой размерности, например, SPAM (размерность 686)[33], калиброванное пространство признаков Певных (размерность 548)[34].

Благодаря принципу открытости с помощью метода машинного обучения удалось построить эффективные классификаторы для многих распространенных стеганографических систем: YASS [35], PQ [36], MOX [37] и других. Тем самым была показана универсальность метода машинного обучения в стегоанализе.

ГЛАВА 2. ПРОЕКТИРОВАНИЕ МОДЕЛИ НЕЙРОННОЙ СЕТИ

2.1. Модель свёрточной нейронной сети для стегоанализа

Являясь одной из наиболее представительных моделей глубокого обучения, CHC представляют собой иерархические нейронные сети, в которых обучаемые фильтры и операции объединения применяются поочередно к необработанным входным изображениям, приводя к все более иерархичным представлениям сложных объектов. Они были применены к задачам распознавания[38] и классификации[49] изображений и показали превосходную производительность.

CHC объединяет 3 архитектурных идеи, как показано на рис. 2.1: частичные области, совместные веса, и объединение[3].

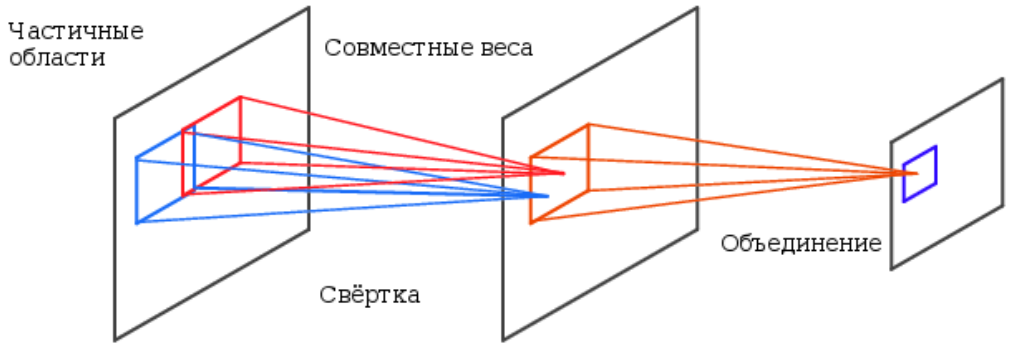


Рис. 2.1. Один типичный слой в CHC с операцией свёртки и объединения.

По сравнению со стандартными нейронными сетями прямого распространения с аналогичными размерами слоев, CHC имеют гораздо меньше соединений и параметров. Следовательно, их гораздо легче тренировать, в то время как теоретически наилучшая производительность, скорее всего, будет лишь немного хуже[39]. Типичная CHC состоит из двух видов слоев: свёрточного слоя, который обычно следует за операцией

объединения и слоя классификации. Они различаются в том, как реализуются операции свертки и объединения и как обучаются сети[40].

CHC принимается в качестве основы для модели стеганализа по определенным соображениям. Во-первых, CHC могут принимать необработанные данные в качестве входных данных без необходимости в этапе выделения признаков. В CHC процесс выделения и классификации признаков унифицирован в рамках единой структуры. Это непосредственно удовлетворяет цели, заключающейся в обучаемом формировании признаков встраивания с изображений вместо того, чтобы рассматривать CHC как еще один классификатор, построенный на существующих методах стегоанализа, таких как Spatial domain Rich Model(SRM). Во-вторых, CHC являются контролируемыми моделями. В отличие от некоторых задач ИИ, маркированные данные довольно легко получить в стеганализе. Контейнеры и стегосообщения могут рассматриваться как положительные и отрицательные образцы соответственно. В-третьих, при использовании метода свёрточного обучения можно обучать модели относительно крупномасштабными изображениями (например, 256x256 пикселей). В задачах ИИ, изображения могут быть поддискретизированы к малым (например, 32x32 пикселя), чтобы сделать процесс обучения быстрее. Однако эта операция стирает шум стегосообщения, следовательно, делает невозможным обнаружение. Поэтому обусловлена модель глубокого обучения, которая может работать с крупномасштабными изображениями.

Несмотря на успешное применение к некоторым задачам ИИ, действующие CHC не учитывают статистические свойства, которые важны для стегоанализа. Чтобы эффективно собирать полезную статистическую информацию для стеганализа, предлагается модель CHC с гауссовым нейроном, настраиваемую по глубине модель CHC. Архитектура сети представлена на рис. 2.2 (справа). Слева от рисунка показана блок-схема традиционных схем стеганолиза для сравнения. Эта модель принимает пиксели в качестве входных данных и состоит из трех типов слоев: слоя

обработки изображений, нескольких сверточных слоев для представления признаков и нескольких полностью связанных слоев для классификации. В отличие от традиционных схем параметры извлекаются автоматически.

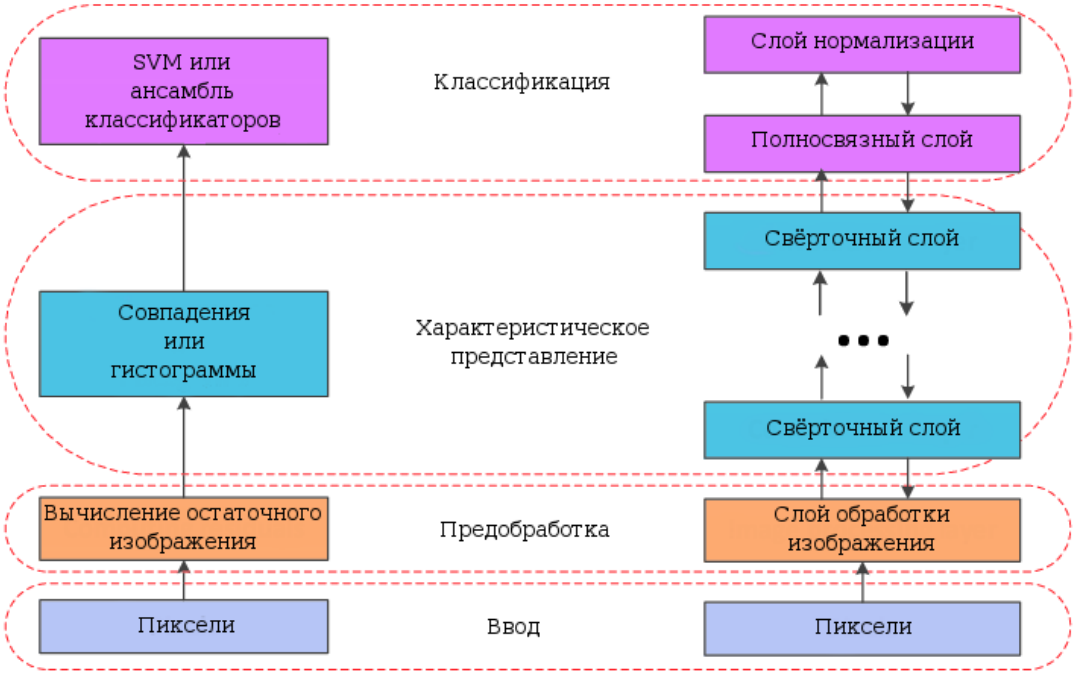


Рис. 2.2. CHC на основе гауссовой функции (справа) и традиционная архитектура стегоанализа основанная на характеристиках, составленых вручную(слева). Стрелками показаны направления этапов.

Левая часть рисунка, для сравнения, показывает порядок работы традиционных схем стегоанализа. Разработанная модель берёт пиксели на входе, и состоит из трёх видов слоёв: слой обработки изображения, несколько свёрточных слоёв для выделения признаков встраивания, и несколько полносвязных слоёв для классификации. В отличии от традиционных схем, выделение признаков получаются автоматически.

2.1.1 Слой обработки изображения

В этом слое выполняется операция фильтрации с предопределенным фильтром верхних частот, который фиксируется во время тренировки. Как правило, высокочастотный стего-шум, добавленный к контейнеру, является

своего рода очень слабым сигналом, на который сильно влияет содержимое изображения. Следовательно, с помощью фильтрации высоких частот укрепляется слабый стегосигнал. Это может обеспечить хорошую инициализацию для всей сети, следовательно, добиться хорошей производительности по сравнению со случайной инициализацией. в традиционных схемах стегоанализа это распространённая практика предобработки. Математически фильтрация может быть выражена, как показано ниже.

$$R=K*I, \tag{2.1}$$

где I – это изображение, R - изображение после фильтрации верхних частот (обычно называемое остаточным изображением), символ * – означает операцию свёртки, а K – линейный фильтр с конечной импульсной характеристикой, не зависящий от сдвига, для вычисления остатка.

В данной модели вместо всего семейства шумовых остатков, используемых в современных наборах признаков, таких как SRM и PSRM, используется только один из соответствующих фильтров. Тем не менее, будет доказано, что узнанное представление функции сопоставимо с наборами пространственных объектов большой размерности.

В данной модели используется ядро свёртки kv, показанное ниже.

$$K_{kv} = \frac{1}{2}\begin{pmatrix} -1 & 2 & -2 & 2 & -1 \\ 2 & -6 & 8 & -6 & 2 \\ -2 & 8 & -12 & 8 & -2 \\ 2 & -6 & 8 & -6 & 2 \\ -1 & 2 & -2 & 2 & -1 \end{pmatrix}$$

2.1.2 Свёрточный слой

После применения операций фильтрации к стегоконтейнеру в слое обработки изображения, иерархически собирается вместе ответ стего сигнала из локального в глобальный в свёрточном слое.

Вход и выход каждого свёрточного слоя – это набор массивов называющийся картой признаков. На выходе, каждая карта признаков это конкретное представление признаков извлечённое из всех мест во входе. В свёрточном слое присутствуют три вида операций, свёртка, нелинейная функция активации и объединение, обычно применяются последовательно как представлено ниже.

$$X_j^i = pool(f(\sum_i X_i^{l-1} * K_{ij}^l + b_j^l)), \qquad (2.2)$$

где *f()* обозначает нелинейную операцию, *pool()* обозначает объединение, X_j^i это *j*-я карта признаков в слое *l*, X_i^{l-1} это i –я карта признаков в слое *l-1*, K_{ij} это обучаемое ядро свёртки соединяющее *j*-ю выходную карту и *i*-ю входную карту, b_j^l это обучаемый параметр смещения для *j*-й выходной карты.

Для операции свёртки, каждая выходная карта признаков обычно сочетает в себе свёртки с нескольких входных карт признаков. Сверточная структура включает идеи локальных регионов и общие веса. С локальными регионами, каждый низкоуровневый признак вычисляется только из подмножества ввода, такого как окрестность пикселя в заданной позиции изображения. Такой экстрактор локальных признаков совместно использует одни и те же параметры при применении в разных соседних местах ввода, что эквивалентно свертке значений пикселя изображения с ядром, содержащим весовые параметры. Доля параметров создает инвариантную относительно сдвига операцию, а также уменьшает число свободных переменных, следовательно, увеличивает производительность обобщения сети[41].

В стеганализе считается, что для естественных изображений полученных фотокамерой последующая внутрикамерная обработка во время получения изображения, такая как цветовая интерполяция, шумоподавление, цветокоррекция и фильтрация, вводит сложные зависимости в шумовую составляющую соседних пикселей. Но стеговый шум, вызванный стеганографическим внедрением, будет нарушать эти зависимости. В самом деле, большинство методов стегоанализа пытаются использовать эти зависимости для обнаружения присутствия стегового шума. Эвристически предполагается, что из-за сложных зависимостей хорошая оценка центрального пикселя может быть получена из соседних пикселей, за исключением оцениваемого пикселя. Затем, вычитая истинное значение центрального пикселя из оцененного, можно получить значение ошибки предсказания, которое непосредственно отражает, изменен или нет пиксель. В настоящее время успешные предсказательные методы, которые обычно представляют собой некоторые инвариантные к сдвигу фильтры, разрабатываются вручную. В рамках предлагаемой архитектуры предикторы вырабатываются автоматически. Принимая во внимание то, что операция фильтрации на уровне обработки изображений породила исходный остаточный шум, операция свертки на каждом свёрточном уровне заключается в иерархическом захвате зависимостей между большей окрестностью и обеспечении точности прогноза.

Затем нелинейная активационная функция применяется поэлементно к выходу операции свертки. Функция нелинейной активации имеет эффект ограничения амплитуды выходного сигнала. Что еще более важно, для многослойных сетей это делает возможным решение некоторых проблем, которые невозможны с однослойными. Важен выбор функции активации. Это определяется характером данных и предполагаемым распределением целевых переменных. Обычно это сигмоидальная функция, такая как логистическая сигмоида или сигмоидальная функция гипербалического

тангенса. Для стегоанализа, используется функция Гаусса, которая может быть выражена как

$$f(x) = \frac{-x^2}{e^{\sigma^2}}, \tag{2.3}$$

Где σ – параметр, определяющий ширину кривой. Нейрон с этой функцией активации производит значительный положительный ответ только тогда, когда входной сигнал попадает в небольшой интервал вокруг нуля. В частности, максимальный отклик будет получен в центре нуля. Насколько известно, гауссовая функция впервые используется в качестве функции активации в глубоких СНС, поэтому стоит называть этот вид СНС гауссова СНС (гСНС).

Как уже упоминалось, целью операции свёртки является вычисление значений ошибок прогнозирования путем использования зависимостей между соседними элементами. Мотивация, к использованию в основе гауссовой нелинейной функции, заключается в том, что она лучше отличает стего сигнал и сигнала контейнера от значений ошибки предсказания по сравнению с сигмоидальными функциями. Рис. 2.3 дает простой пример того, почему работает гауссова активация.

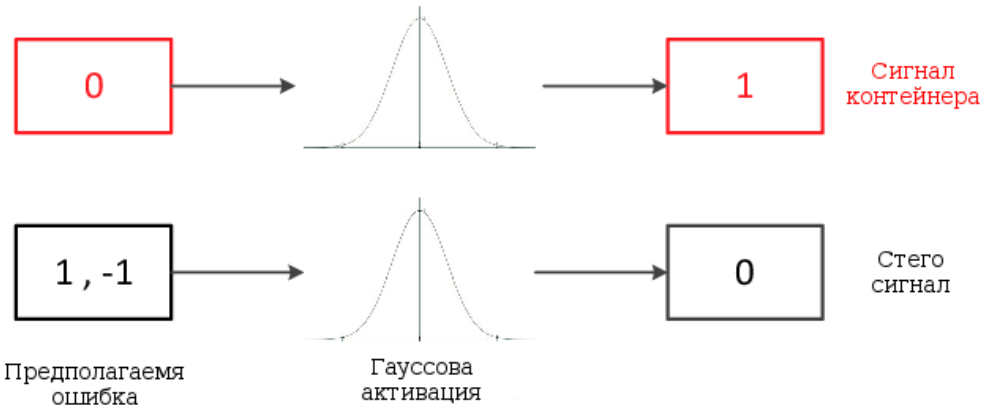


Рис. 2.3. Простой пример, иллюстрирующий, почему активационная функция Гаусса работает для стегоанализа в предложенной модели.

Гауссовская активация может различать сигнал стего и сигнал контейнера от значений ошибки предсказания.

В идеале должно быть три типа значений ошибки прогнозирования: 1, -1 и 0. Значения 1 и -1 означают, что пиксель модифицирован операцией вложения, и считается, что здесь есть сигнал стего. Значение 0 означает, что пиксель не менялся, и мы считаем, что это сигнал контейнера. При гауссовской активации значения ошибки предсказания, соответствующие сигналу обложки или стего сигнала, преобразуются в 0 или 1 соответственно. Следовательно, сигнал контейнера и стего сигнал разделены. На практике трудно получить такие точные значения дискретного предсказания для каждого пикселя, чтобы принимать решение «да» или «нет». Фактически, значения ошибки предсказания в структуре являются непрерывными, что отражает силу стего сигнала в соответствующих позициях. С гауссовской функцией активации значения, которые далеки от нуля, которые более релевантны для сигнала стего, преобразуются примерно до 0. Между тем, значения вокруг нуля, которые более релевантны сигналу контейнера, преобразуются примерно до 1. Следовательно, использование гауссовой функции активации позволяет разрабатывать на основе сети точные предикторы для стегоанализа. Полученные в результате активации данные затем передаются в объединяющую часть слоя. Операция объединения предназначена для преобразования низкоуровневого представления признаков в более полезное, которое сохраняет важную информацию и отбрасывает ненужные детали[42]. В общем, для представления признаков более высокого уровня требуется информация от постепенно больших входных регионов. Операция объединения приводит к объединению информации в пределах набора небольших локальных областей, одновременно сокращая время вычислений[43]. В операции объединения итоговые результаты соседних групп нейронов в одной и той же карте пространственных объектов суммируются.

Как правило, обычно существует два варианта объединения: усреднённое объединение и максимальное объединение. Первый вариант принимает среднее значение внутри области объединения:

$$pool(R_j) = \frac{1}{|R_j|}\sum_{i \in R_j} a_j, \quad (2.4)$$

В то время как операция максимального объединения выбирает максимальное значение:

$$pool(R_j) = max\ a_i, i \in R_j, \quad (2.5)$$

где R_j – область объединения в j-й карте признаков, a_i.- i – й элемент в ней.

Максимальное объединение только фиксирует самую сильную активацию в области объединения. Оно хорошо подходит к виду представления признака, которое очень разрежено (т.е. имеет очень низкую вероятность быть активным)[44]. Однако для стеганализа при такой операции может быть потеряна некоторая полезная информация. Стего сигнал является своего рода очень слабым сигналом поэтому недостаточно для точного предсказания, просто опираться на ответы от отдельных элементов. В данной модели используется усреднённое объединение. При усреднённом объединении учитываются все активации в области объединения, которые должны отбрасывать распределения, вызванные отдельными элементами. Благодаря объединению всего сигнала в пределах области объединения стего сигнал на всей области усиливается. Объединённая структура в каждом сверточном слое изобажена на рис. 2.4:

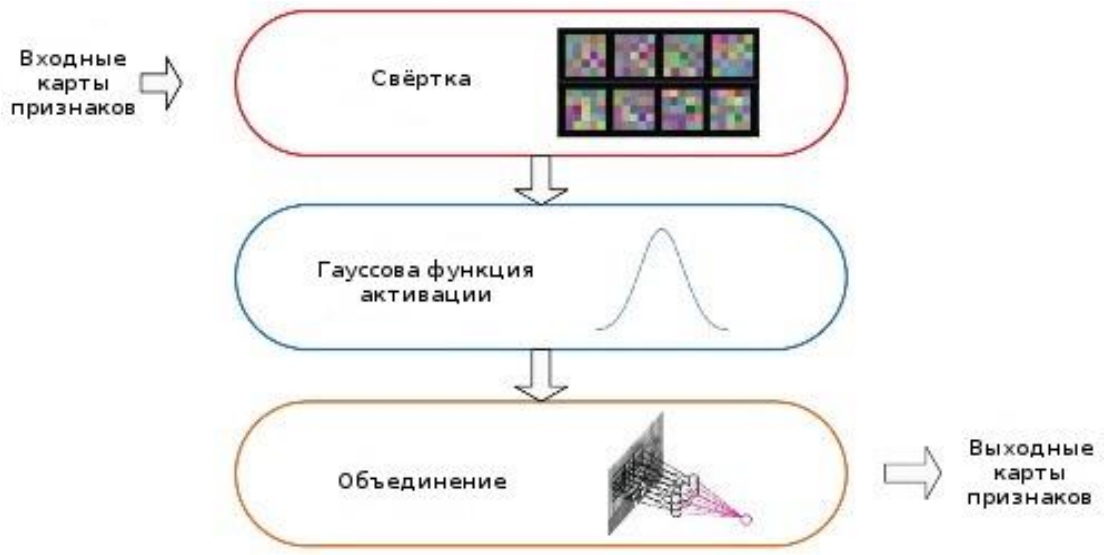


Рис. 2.4. Компоненты сверточного слоя в модели, включая свертку, гауссовскую нелинейность и усреднённое объединение.

Каждый сверточный слой извлекает объекты из всех карт предыдущего слоя. По мере того, как сеть становится все глубже, более сложные и более высокие зависимости моделируются прогрессивно с участием больших входных областей. Следовательно, стего сигнал агрегируется иерархически от локального до глобального. Эти узнаваемые высокоуровневые признаки позволяют нейронам верхнего уровня предсказывать изменена ли входная область или нет.

2.1.3 Классификационный слой

Слой классификации состоит из нескольких полносвязнных слоев. Полученные признаки передаются на эти слои. На верхнем слое используется нормированная экспоненциальная функция активации для создания распределения по всем меткам класса. В данной модели используется двухсторонняя нормированная экспоненциальная функция, как показано ниже:

$$y_i = \frac{e^{x}i}{\sum_{j=1}^{2} e^{x}i}, \tag{2.6}$$

Для i = 1,2, где x_i – суммарный входной сигнал к нейрону i в верхнем слое, а y_i.- его выход.

Чтобы уменьшить проблему переобучения, следует применяеть методика, называемую «отсев», для регуляции полносвязанных слоев[45]. В тренировка с выбыванием, выход каждого нейрона в соответствующих слоях устанавливается равным нулю с вероятностью 0,5. Этот метод улучшает обобщающую способность сети и дает улучшенную производительность теста.

ГЛАВА 3. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ И ТЕСТИРОВАНИЕ СИСТЕМЫ

3.1. Выбор средств реализации

На основании спроектированной модели свёрточной нейронной сети была разработана и реализована автоматизированная система для анализа изображений на предмет скрытой информации.

3.1.1. Выбор языка реализации

Реализация системы предъявляет некоторые требования к языку программирования, а именно расширенные возможности многопоточности на GPU и низкоуровневая работа с данными. В целях выбора были рассмотрены два, на сегодняшний день, популярных языка реализации нейронных сетей, C++ и Python. Для данных языков существует множество библиотек для реализации глубинного обучения.

C++ – компилируемый, статически типизированный язык программирования общего назначения.

Поддерживает такие парадигмы программирования, как процедурное программирование, объектно-ориентированное программирование, обобщённое программирование. Язык имеет богатую стандартную библиотеку, которая включает в себя распространённые контейнеры и алгоритмы, ввод-вывод, регулярные выражения, поддержку многопоточности и другие возможности. C++ сочетает свойства как высокоуровневых, так и низкоуровневых языков. В сравнении с его предшественником – языком C, – наибольшее внимание уделено поддержке объектно-ориентированного и обобщённого программирования.

C++ содержит средства разработки программ контролируемой эффективности для широкого спектра задач, от низкоуровневых утилит и драйверов до весьма сложных программных комплексов. В частности:

- Высокая совместимость с языком Си : код на Си может быть с минимальными переделками скомпилирован компилятором C++. Внешнеязыковой интерфейс является прозрачным, так что библиотеки на Си могут вызываться из C++ без дополнительных затрат, и более того – при определённых ограничениях код на C++ может экспортироваться внешне не отличимо от кода на Си (конструкция extern "C").
- Как следствие предыдущего пункта – вычислительная производительность. Язык спроектирован так, чтобы дать программисту максимальный контроль над всеми аспектами структуры и порядка исполнения программы. Один из базовых принципов C++ – «не платишь за то, что не используешь» – то есть ни одна из языковых возможностей, приводящая к дополнительным накладным расходам, не является обязательной для использования. Имеется возможность работы с памятью на низком уровне.
- Поддержка различных стилей программирования: традиционное императивное программирование (структурное, объектно-ориентированное), обобщённое программирование, функциональное программирование, порождающее метапрограммирование.
- Автоматический вызов деструкторов объектов в адекватном порядке (обратном вызову конструкторов) упрощает и повышает надёжность управления памятью и другими ресурсами (открытыми файлами, сетевыми соединениями, соединениями с базами данных и т. п.).
- Перегрузка операторов позволяет кратко и ёмко записывать выражения над пользовательскими типами в естественной алгебраической форме.
- Имеется возможность управления константностью объектов (модификаторы const, mutable, volatile). Использование константных

объектов повышает надёжность и служит подсказкой для оптимизации. Перегрузка функций-членов по признаку константности позволяет определять выбор метода в зависимости цели вызова (константный для чтения, неконстантный для изменения). Объявление mutable позволяет сохранять логическую константность при виде извне кода, использующего кэши и ленивые вычисления.

- Шаблоны C++ дают возможность построения обобщённых контейнеров и алгоритмов для разных типов данных. Попутно шаблоны дают возможность производить вычисления на этапе компиляции.
- Возможность встраивания предметно-ориентированных языков программирования в основной код. Такой подход использует, например библиотека Boost.Spirit, позволяющая задавать EBNF-грамматику парсеров прямо в коде C++. Boost.Spirit реализует рекурсивно-нисходящий алгоритм, что накладывает соответствующие ограничения (такие как недопустимость левой рекурсии).

Библиотеки C++ для работы с нейронными сетями:

- OpenNN[45]
- FANN[46]
- ConvNet[47]
- ALGLIB[48]

Для реализации многопоточности в свёрточных нейронных сетях на GPU в C++ хорошо подходит библиотека cuda-convnet2[49].

Python – высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объём полезных функций.

Python поддерживает несколько парадигм программирования, в том числе структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное. Основные архитектурные черты

– динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений и удобные высокоуровневые структуры данных. Код в Python организовывается в функции и классы, которые могут объединяться в модули (они в свою очередь могут быть объединены в пакеты).

Python – активно развивающийся язык программирования, новые версии (с добавлением/изменением языковых свойств) выходят примерно раз в два с половиной года. Вследствие этого и некоторых других причин на Python отсутствуют стандарт ANSI, ISO или другие официальные стандарты, их роль выполняет CPython.

Несомненным достоинством является то, что интерпретатор Python реализован практически на всех платформах и операционных системах. Первым таким языком был C, однако его типы данных на разных машинах могли занимать разное количество памяти и это служило некоторым препятствием при написании действительно переносимой программы. Python же таким недостатком не обладает.

Следующая немаловажная черта - расширяемость языка, этому придается большое значение и, как пишет сам автор, язык был задуман именно как расширяемый. Это означает, что имеется возможность совершенствования языка всеми всеми заинтересованными программистами. Интерпретатор написан на C и исходный код доступен для любых манипуляций. В случае необходимости, можно вставить его в свою программу и использовать как встроенную оболочку. Или же, написав на C свои дополнения к Python и скомпилировав программу, получить "расширенный" интерпретатор с новыми возможностями.

Следующее достоинство - наличие большого числа подключаемых к программе модулей, обеспечивающих различные дополнительные возможности. Такие модули пишутся на C и на самом Python и могут быть разработаны всеми достаточно квалифицированными программистами. В качестве примера можно привести следующие модули:

Numerical Python - расширенные математические возможности, такие как манипуляции с целыми векторами и матрицами;

Tkinter - построение приложений с использованием графического пользовательского интерфейса (GUI) на основе широко распространенного на X-Windows Tk-интерфейса;

OpenGL - использование обширной библиотеки графического моделирования двух- и трехмерных объектов Open Graphics Library фирмы Silicon Graphics Inc. Данный стандарт поддерживается, в том числе, в таких распространенных операционных системах как Microsoft Windows 95 OSR 2, 98 и Windows NT 4.0.

Библиотеки Python для работы с нейронными сетями:

PyBrain[50]

Theano[51]

Pylearn2[52]

Caffe[53]

Для организации многопоточности в Python используется стандартная библиотека threading.py[54]. Вызовы функций из неё используются в выше перечисленных библиотеках.

Из проведённого анализа языков программирования был сделан вывод, что для реализации системы лучше всего подходит язык C++. Python в виду своих достоинств является интерпретируемым языком, что накладывает существенные ограничения на скорость работы системы. В качестве основы для неронной сети была использована модель из библиотеки ConvNet. Для реализации многопоточности на GPU использовалась библиотека cuda-convnet2. Для организации графического интерфейса была использована библиотека Gtk+3[55].

3.1.2. Выбор среды разработки

В качестве среды для разработки на языке C++ были рассмотрены две популярные на сегодняшний день IDE в ОС GNU/Linux – CLion и Eclipse с плагином Eclipse CDT.

CLion - Интегрированная среда разработки для языка программирования «C++». Подходит для операционных систем «Windows», «macOS», и «Linux».

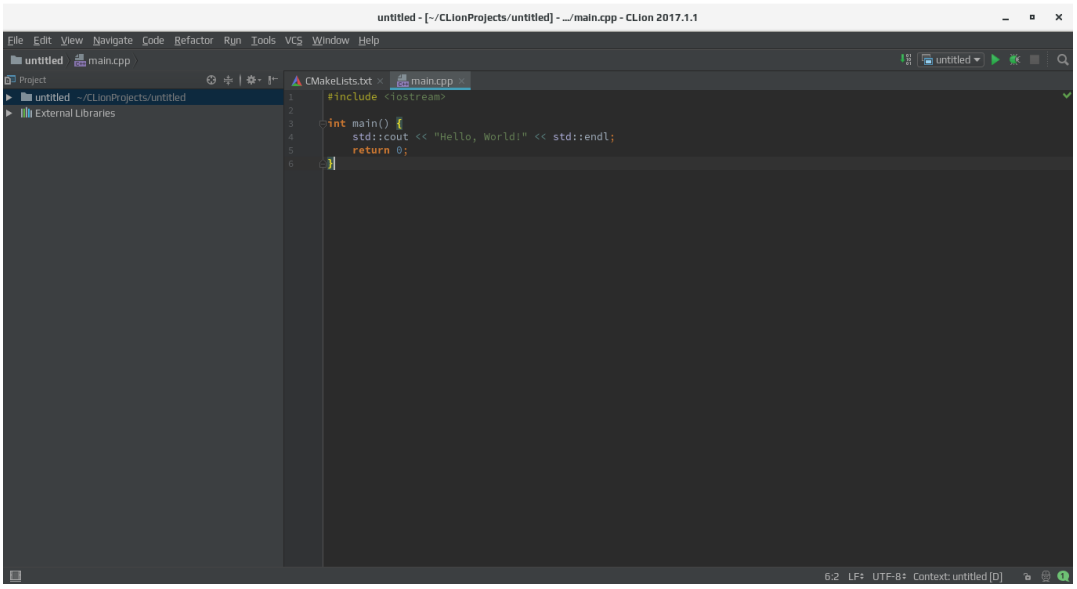


Рис. 3.1. Окно редактора CLion при начале работы с программой.

Многофункциональный, умный редактор кода, предоставляющий возможности автодополнения, множественных курсоров, автоформатирования кода. Мгновенная навигация под коду в один клик, в том числе переход на любой класс / символ / файл по его имени, переход на определение символа в родительском классе и другие. Быстрые сочетания клавиш практически для всех действий и команд. Стандартные раскладки IntelliJ IDEA, Emacs, Visual Studio, Eclipse, NetBeans, Xcode, ReSharper, а также возможности настройки раскладок. Шаблоны готового кода, функции генерации конструкторов/деструкторов классов, методов для чтения/записи данных класса. Безопасные рефакторинги, позволяющие автоматически находить и исправлять все использования изменяющегося фрагмента кода: Rename, Change Signature, Extract Function/Variable/Constant/Define/Typedef,

Inline, Pull Members Up, Puch Members Down и другие. За счета анализа кода на лету не только подсвечиваются потенциальные проблемы, но и сразу предлагаются способы их исправить (quick-fixes). Поддержка CMake, включающая редактор CMakeCache, автоматическое добавление новых C/C++ файлов в существующие CMake-таргеты, автоматическую перезагрузку проекта и автоматическое дополнение команд CMake. Встроенный полнофункциональный отладчик, позволяющий выставлять точки остановки (breakpoints), отслеживать значения выделенных переменных (watches), вычислять выражения, отображать структуру STL контейнеров и не только. Помимо поддержки C и C++ (в том числе C++11, libc++ и Boost), CLion также работает с JavaScript, XML, HTML и CSS. Интеграция с системами контроля версий Subversion, Git, GitHub, Mercurial, CVS, Perforce и TFS, а также с баг-трекерами JIRA, Youtrack, Lighthouse, Pivotal Tracker, GitHub и др. Встроенный терминал. Эмуляция Vim-режима (посредством плагина IdeaVim). Большой репозиторий плагинов для расширения имеющейся функциональности.

Eclipse CDT – интегрированная среда разработки C и C++ на базе платформы Eclipse.

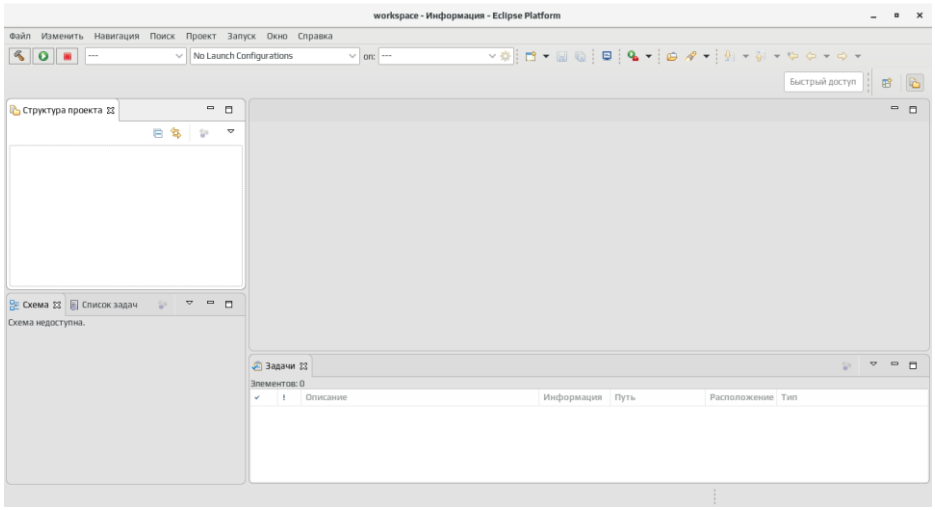


Рис. 3.2. Окно редактора Eclipse CDT при начале работы с программой.

Пакет инструментальных средств разработки на C++ (CDT) Eclipse является расширением платформы Eclipse в форме плагина. Этот плагин

доступен в варианте для любой платформы. Дружественность плагина для пользователя и то, что он имеет открытый исходный код, делает его популярным не только среди разработчиков для Linux, но и среди разработчиков на C++, использующих другие платформы. Плагины CDT и Web Tools являются двумя наиболее распространёнными расширениями для Eclipse. Примерно два из трёх разработчиков, использующих CDT, являются пользователями Windows.

CDT имеет субкомпоненты или плагины, являющиеся независимыми разработками сообщества CDT. Самым важным является основной плагин CDT, обеспечивающий базовые возможности CDT. CDT Debug UI (пользовательский интерфейс отладчика) обеспечивает возможности пользовательского интерфейса для программ редактирования и просмотра при отладке. Плагин CDT UI обеспечивает связанные с пользовательским интерфейсом (UI) функциональные возможности, программы просмотра, редактирования, мастеры и т.д. Отладчик CDT обеспечивает базовые возможности отладки. CDT Feature даёт компонент CDT Feature. Ядро CDT обеспечивает Core Model, CDOM, и другие базовые компоненты. CDT Launch реализует механизм для запуска внешних выполняемых модулей и инструментальных средств. CDT Debug MI (машинный интерфейс) - коннектор приложений для MI-совместимых отладчиков.

Редакторы CDT содержат несколько возможностей, которые делают их популярными. Например, подсветка синтаксиса и помощник по коду (code assist) делают разработку ПО быстрой и простой. Подсветка синтаксиса конфигурируется и может быть настроена индивидуально, в соответствии с личными предпочтениями разработчика. Помощник по коду - это функция дописывания строк кода аналогичная такой же функции в Visual Studio. В плагин можно добавлять задаваемые пользователем шаблоны кода, которые могут затем использоваться помощником по коду.

В итоге выбор был сделан в пользу IDE CLion, так она имеет более расширенные возможности для разработки на C++.

3.2. Реализация системы

Исходя из разработанной модели нейронной сети и поставленной задачи, была разработана следующая архитектура автоматизированной системы:

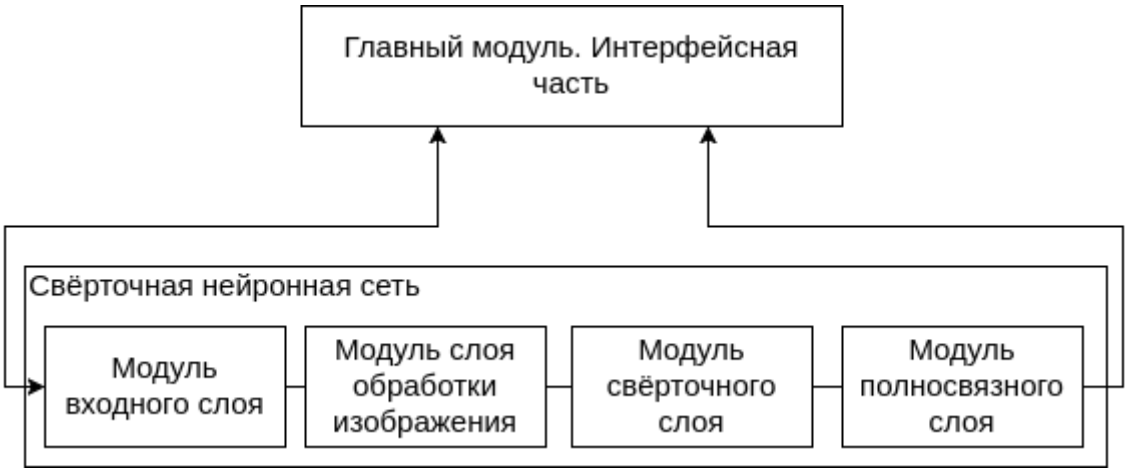


Рис. 3.3. Архитектура системы.

Пользователь указывает проверяемое изображение-контейнер или массив изображений в окне главного модуля (см. Приложение 1). Далее изображение передаётся модулю входного слоя нейронной сети. Затем передаётся модулю слоя обработки изображения, где подвергается операции фильтрации (см. Приложение 2). Следующим в работу вступает модуль описывающий функции свёрточного слоя. В нём к картам признаков последовательно применяются операции свёртки и гауссовой нелиенйности (см. Приложение 3). После, в модуле классификационного слоя выводится решение о классификации, в виде числового значения в промежутке между 1 и 0. Решение выводится в окно главного модуля или, если это массив избражений в файл, который сохраняется в домашней директории пользователя.

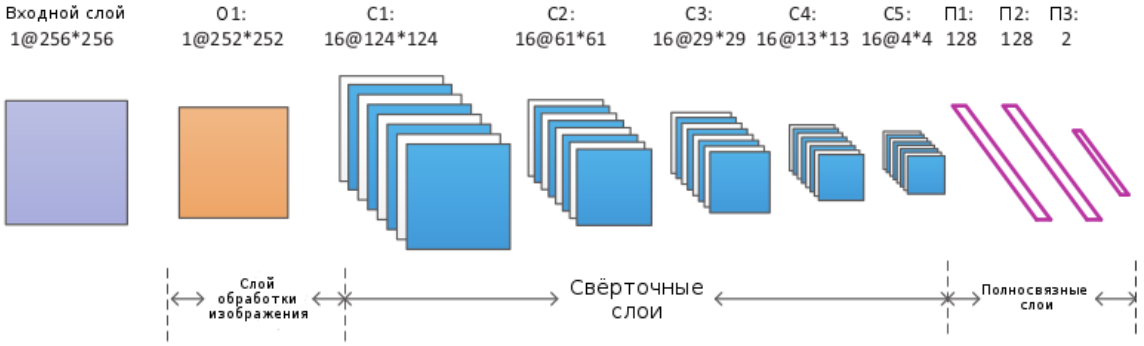


Рис. 3.4. Праметры свёрточной нейронной сети реализованной в данной системе. Форма «a @ b * b» означает количество карт характеристик a и разрешение b * b соответствующего слоя.

гСНС обучалась путем минимизации – log y_t, где $t \in \{1,2\}$ обозначает целевой класс, используя алгоритм обратного распространения. Распространение ошибок и адаптация к весу в свёрточных и полностью связанных слоях следуют стандартной процедуре. Подробности процедуры приведены в [3]. Одним словом, все параметры в уровнях извлечения признаков и уровнях классификации оптимизируются совместно.

Первый свёрточный слой фильтрует входное изображение с ядром размером 5 × 5. Второй свёрточный слой берет выходной сигнал первого уровня как входной и фильтрует его с 16 ядрами размером 5 × 5. Третий, четвертый и пятый свёрточные слои применяют свертки с 16 ядрами размером 3 × 3 соответственно. Гауссова нелинейная функция применяется ко всем выходам второго-пятого свёрточных слоев. Между тем, за каждым из второго-пятого свёрточных уровней следует операция усреднённого объединения с размером окна 3 × 3 и шагом величиной в 2, который работает на каждой из карт признаков в соответствующем свёрточном слое и приводит к такому же количеству карт пространственных объектов с уменьшенным пространственным разрешением. Наконец, извлеченные 256 признаков передаются в модуль классификации, который состоит из трех полносвязных слоев. Первые два полносвязаных слоя имеют по 128 нейронов. Выход каждого нейрона в первых двух полностью связанных

слоях ГСНС активируется функцией Rectified Linear Units (ReLUs) $f(x) = max(0, x)$[56]. Последний полносвязный слой имеет два нейрона, и его выход подается на двухстороннюю нормированную экспоненциальную функцию активации.

Структура проекта:

- img – дирректория с изображениями для обучения.
- src – дирректория с файлами иходного кода.
- logs – дирректория с файлами логированния ошибок и событий в системе.
- build – директория с компилированными исполнительными файлами.

Файлы исходного кода:

- main.h – заголовочный файл с переменными и объявлениями фунций для главного модуля
- main.cpp – главный модуль программы.
- input_layer.h – заголовочный файл для модуля входного слоя.
- input_layer.cpp – модуль входного слоя.
- img_processing_layer.h – заголовочный файл для модуля обработки избражения.
- img_processing_layer.cpp – модуль содердащий функции слоя обработки изображения.
- conv_layer.h – заголовочный файл для модуля свёрточного слоя.
- conv_layer.cpp – модуль содержащий функции свёрточного слоя.
- full_con_layer.h – заголовочный файл для модуля полносвязного слоя.
- full_con_layer.cpp – модуль содержащий функции полносвязного слоя.

Работа с системой происходит в графическом режиме.

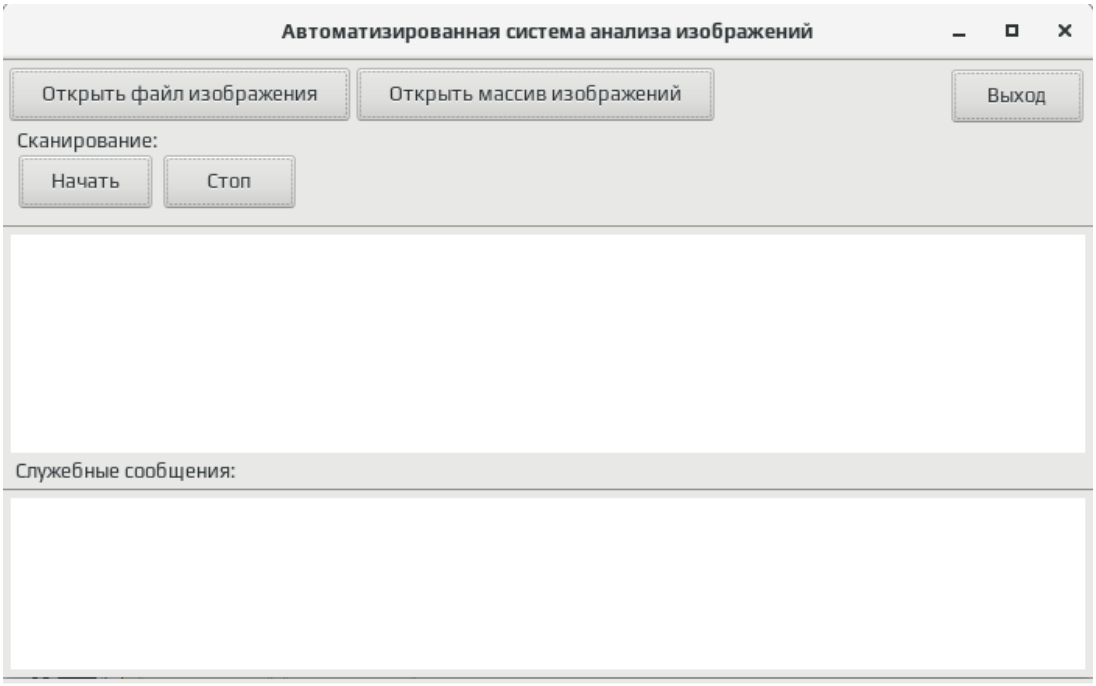


Рис. 3.5. Главное окно системы

В зависимости от поставленной задачи, пользователь может передать системе на сканирование один предполагаемый контейнер и массив контейнеров.

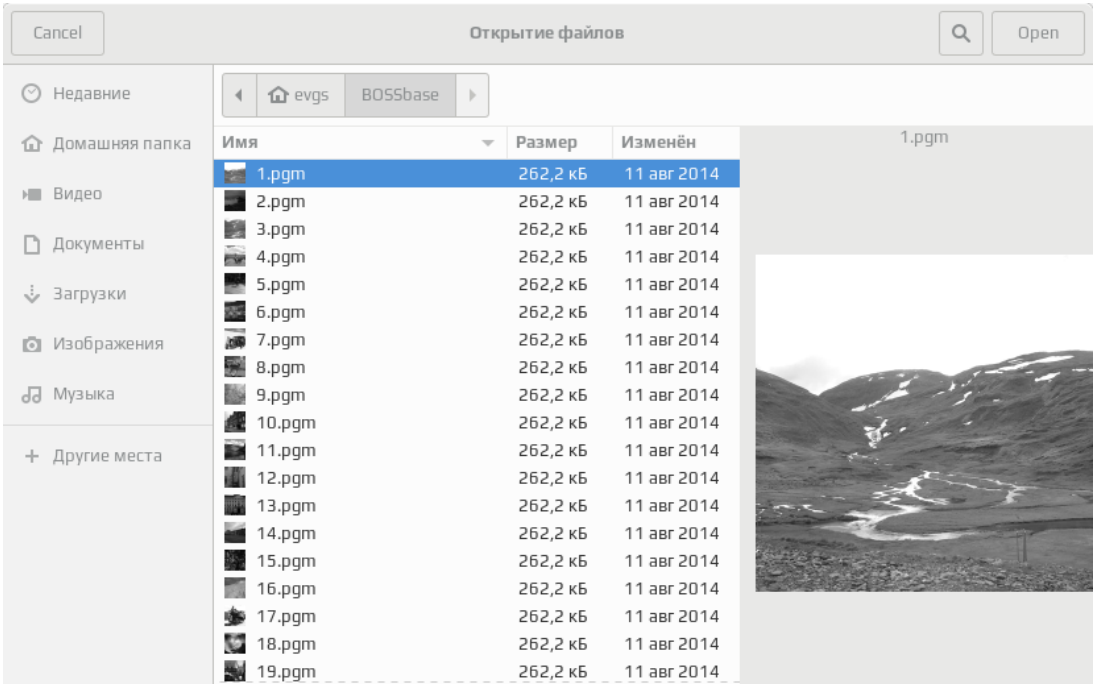


Рис. 3.6. Диалоговое окно выбоа файла/файлов.

В случае если это один конетйнер будет выведено следующее сообщение об итогах анализа.

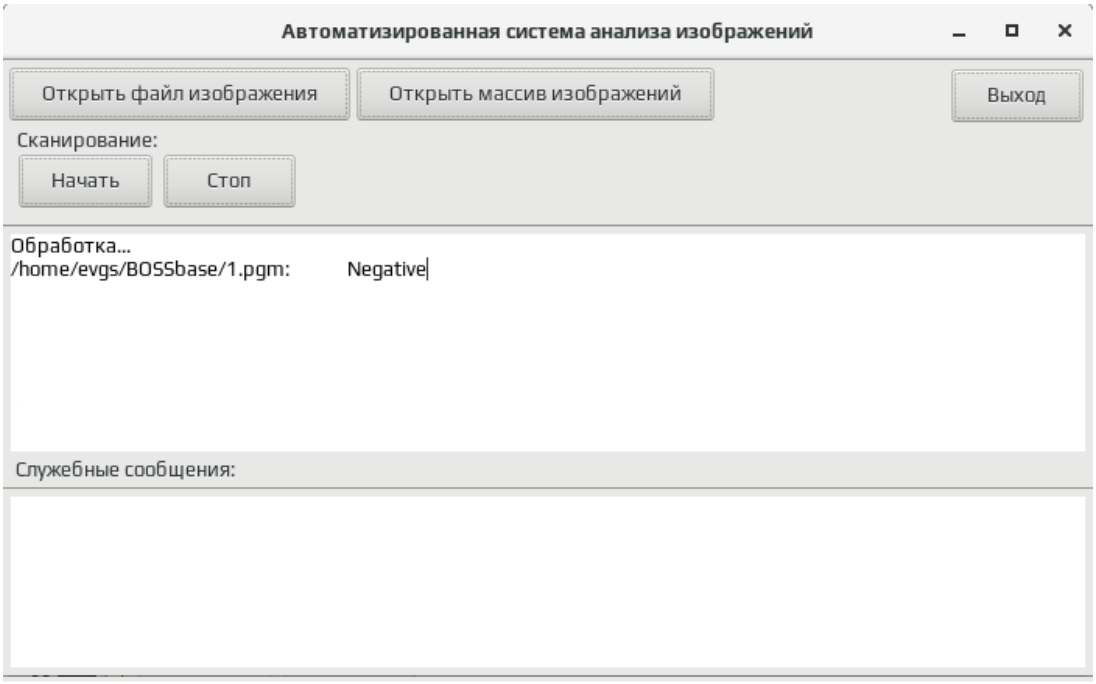


Рис. 3.7. Результат сканирования файла.

В случае если система классифицирует изображение как стегокантейнер. Будет выведено следующее.

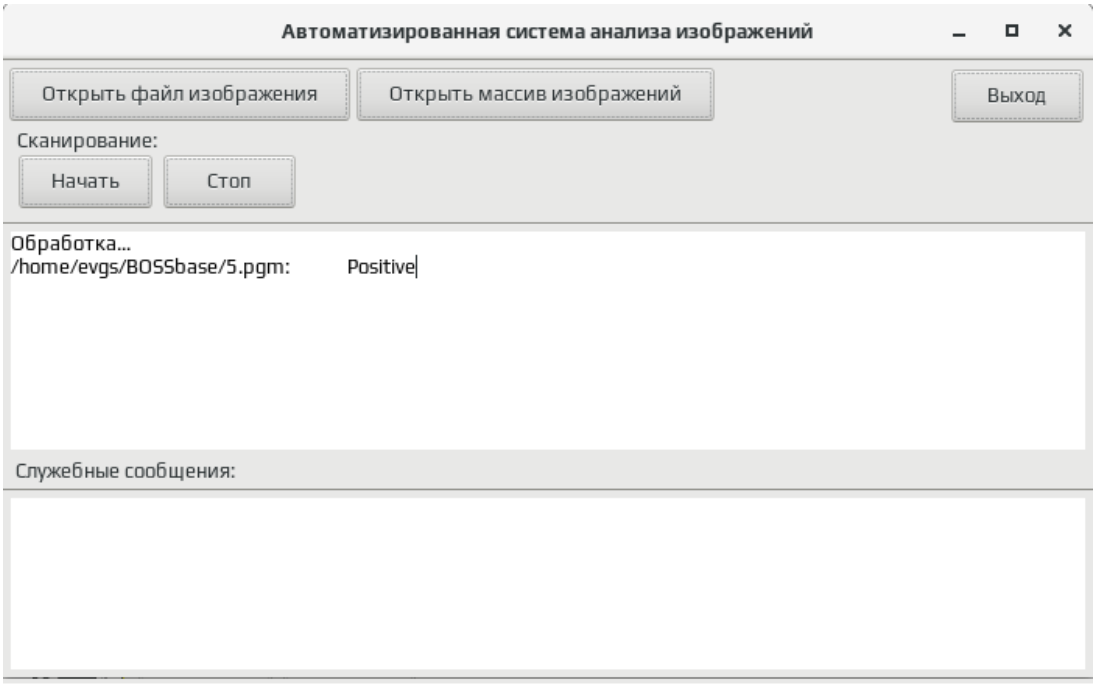


Рис. 3.8. Обнаружен стегоконтейнер.

В случае когда необходимо проверить множество файлов за раз, пользователь выделяет необходимые для проверки файлы и система

последовательно их анализирует. Результаты сохраняются в файл в домашней дирректоррии пользователя в /home.

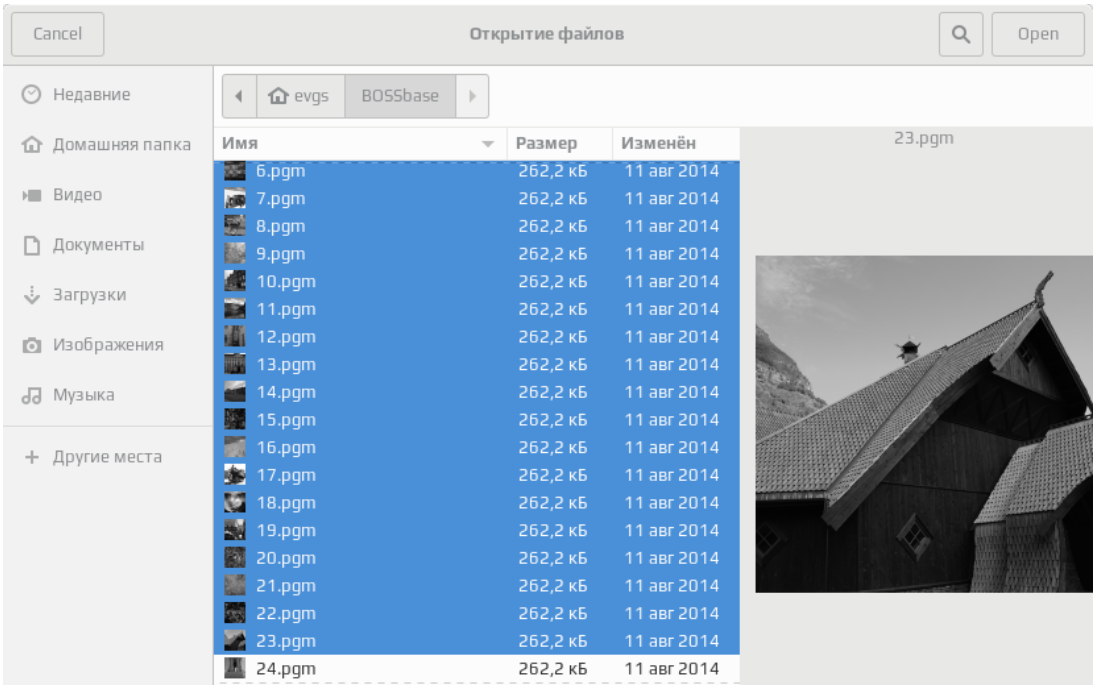


Рис. 3.9. Выбор нескольких изображений.

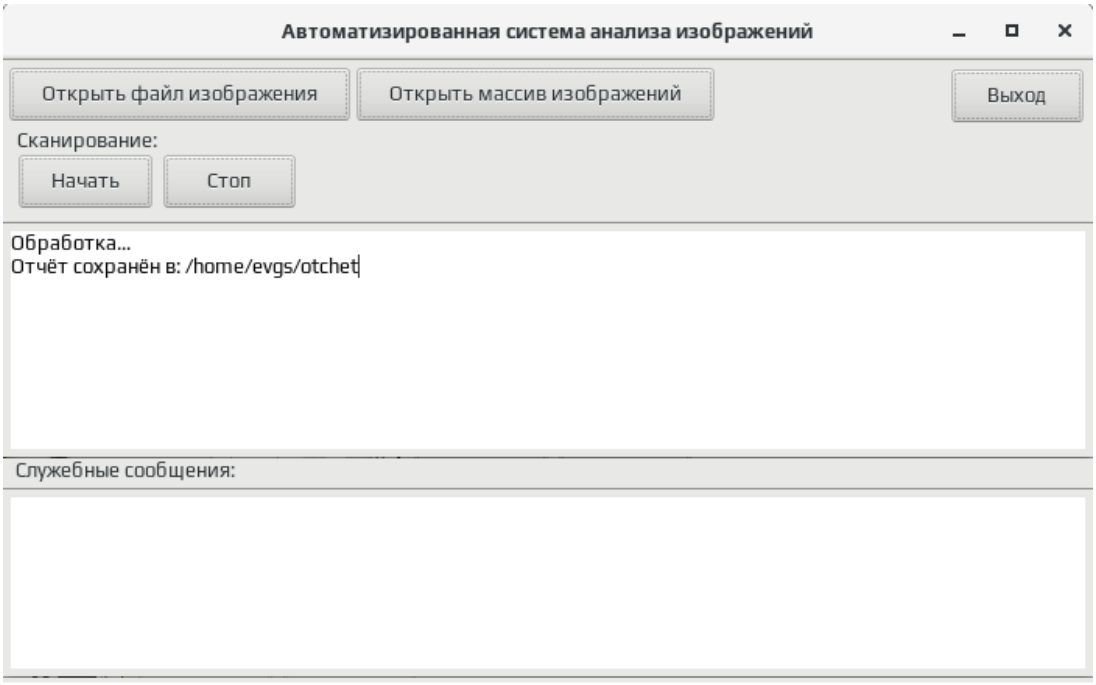


Рис 3.10. Проверка массива изображений.

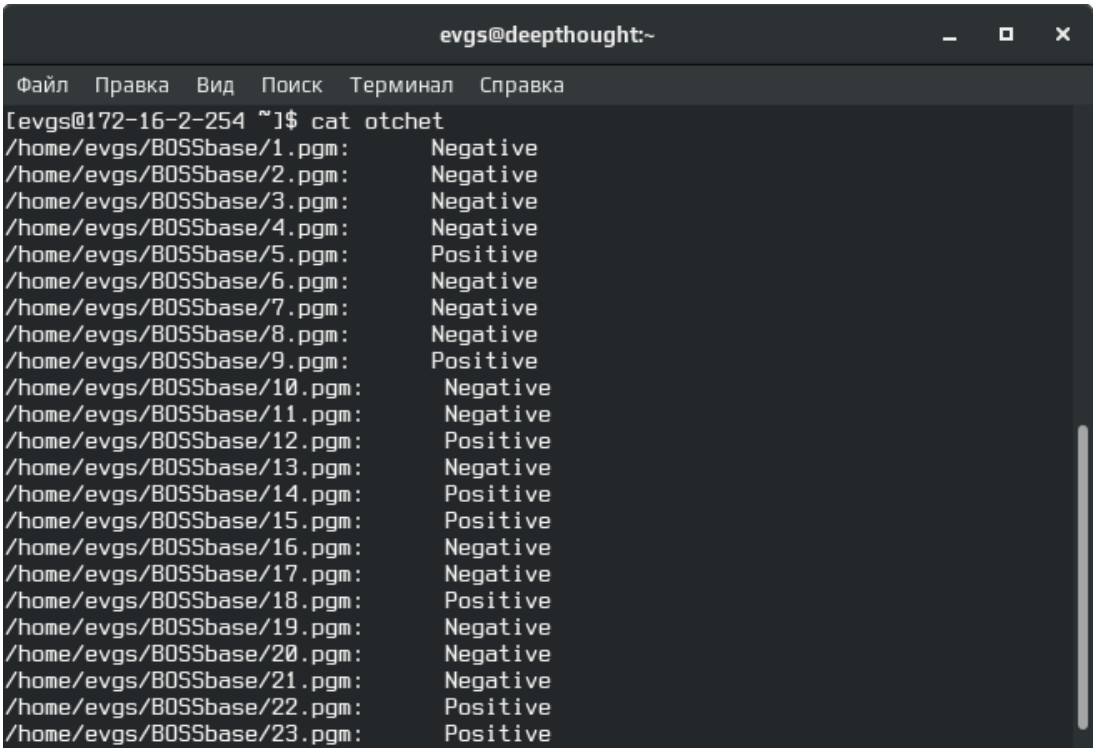


Рис 3.11. Содержание отчёта о проверке массива изображений.

Сеть проходила обучение на сервере, внутри виртуальной машины, на гипервизоре QUEMU/KVM[57] с прямым пробросом процессора Intel Xeon E5-2650 2.0 ГГц и графического ускорителя Tesla K40 12G драйверами virtio[58]. .Среднее время тренировки составило около 2 часов. При использовании более современных графических ускорителей время обучения может значительно сократиться.

3.3. Анализ результатов экспериментов

Эксперименты проводились на стандартизованной базе данных BOSSbase 1.01[59]. Эта база данных содержит 10 000 изображений в оттенках серого, полученные семью цифровыми камерами в необработанном формате и затем ужаты до размера 512 × 512 пикселей. Все изображения храняться в формате pgm. В экспериментах дополнительно был изменён размер изображений до 256 × 256 пикселей. Это изменение размера было необходимо только из-за ограничения вычислительных возможностей. Чтобы

оценить эффективность разработанной системы для стеганализа, эксперименты были проведены на трех современных стеганографических алгоритмах: HUGO, WOW, и S-UNIWARD. Производительность всех методов оценивалась на трех полезных нагрузках: 0,3, 0,4 и 0,5 бит на пиксель (bpp). Ошибки обнаружения представлены в таблице 3.1.

Таблица 3.1

Сравнительная таблица результатов экспериментов.

bpp	HUGO			WOW			S-UNIWARD		
	ГСНС	SRM	SPAM	ГСНС	SRM	SPAM	ГСНС	SRM	SPAM
0.3	33.8%	29.6%	42.9%	34.3%	31.2%	42.2%	35.9%	31.5%	40.0%
0.4	28.9%	25.2%	39.1%	29.3%	25.7%	38.2%	30.9%	26.3%	35.1%
0.5	25.7%	21.4%	35.7%	24.8%	22.1%	34.9%	26.3%	21.4%	30.6%

Результаты сравнивались с двумя вручную подобранными наборами признаков: низкоразмерным набором SPAM[60], реализованным с помощью SVM на основе гауссовой радиальной базисной функции, и высокоразрешающим набором SRM[61], одним из современных наборов признаков, реализованным с помощью ансамбля классификаторов.

На BOSSbase, по всем трем алгоритмам внедрения и полезным нагрузкам. разработанный метод достигает гораздо более низкой ошибки обнаружения, чем набор SPAM. По сравнению с набором SRM, реализованным с помощью ансамбля классификаторов, ошибка обнаружения примерно на 2-5% выше в зависимости от полезной нагрузки. Все результаты этих экспериментов подтверждают, что автоматическое вырабатывание признаков эффективно в разработанной модели. И оно лучше позволяет противодействовать сложным методам втраивания.

Заключение

При выполнении данной магистерской диссертации были выполнены следующие задачи:

1) Исследованы и проанализированны существующие методы стеганографии и обнаружение скрытых сообщений в рамках статистического и нейросетевого подходов в стегоанализе.

2) Разработана архитектуру свёрточной нейронной сети для нужд стегоанализа и архитектура системы для анализа изображений.

3) На основании разработанной архитектуры создан программный продукт.

4) Проведён анализ эффективности системы по сравнению с сущетвующими методами.

В ходе работы над диссртацией была разработана архитектура свёрточной нейронной сети для стегоанализа. Данная архитектура позволяет решить проблему выработки признаков при противодействии сложным алгоритмам встраивания. Также был разработанный программный продукт реализующий преимущества архитектуры при решении задачи классификации контейнера. В дальнейшем данная система может быть расширена добавлением функционала проведения атак на выделение секретного сообщения.

Список использованных источников

1. Hinton, G. E. and Salakhutdinov, R. R., “Reducing the dimensionality of data with neural networks,” Science 313(5786), 504–507 (2006).
2. Salakhutdinov, R. and Hinton, G. E., “Deep boltzmann machines,” in [International Conference on Artificial Intelligence and Statistics], 448–455 (2009).
3. Larochelle, H., Bengio, Y., Louradour, J., and Lamblin, P., “Exploring strategies for training deep neural networks,” The Journal of Machine Learning Research 10, 1–40 (2009).
4. LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P., “Gradient-based learning applied to document recognition,” Proceedings of the IEEE 86(11), 2278–2324 (1998).
5. Simmons G.J. The prisoner`s problem and the subliminal channel, Proc. Workshop on Communications Security (Crypto`83), 1984, 51-67.
6. Pfitzmann B. Information Hiding Terminology, in Information Hiding, Springer Lecture Notes in Computer Science, v.1174, 1996, 347-350.
7. T. Pevny, Steganalysis by subtractive pixel adjacency matrix / T. Pevny, P. Bas, J. Fridrich // IEEE Transactions on Information Forensics and Security, June 2010. – 5(2). – P. 215 – 224.
8. T. Pevny, Using high - dimensional image models to perform highly undetectable steganography // T. Pevny, T. Filler, P. Bas // Information Hiding, 12th International Workshop. – 2010. – LNCS 6387. – P. 161 – 177.
9. M. Barni, Watermarking systems engineering: enabling digital assets security and other applications / M. Barni, F. Bartolin i // Signal processing and communications. – CRC Press, 2004. – 500 p.

10. Fridrich, J. Writing on Wet Paper / J. Fridrich, M. Goljan, P.Lisonek, D. Soukal // IEEE Transactions on Signal Processing. – 2005. –vol. 53, No 10. – P. 3923–3935.

11. Pevny, T. Using high-dimensional image models to perform highly undetectable steganography // T. Pevny, T. Filler, P. Bas // Information Hiding, 12th International Workshop. – 2010. – LNCS 6387. – P. 161–177.

12. Небаева, К.А. Стегосистемы на основе каналов с шумом при использовании слепого декодера / К.А. Небаева // В мире научных открытий. – 2013. –No10.1(46). – С. 118–132.

13. Korzhik, V. Stegosystems based on noisy channels / V. Korzhik, G. Morales-Luna, K. Loban // International Journal of Computer Science and Applications. – 2011.– Vol. 8 No. 1. – P. 1–13.

14. Korzhik, V. The capacity of a stegosystem for the noisy attack channel / V. Korzhik, G. Morales-Luna, K. Nebaeva // Journal of Information Hiding and Multimedia Signal Processing. – 2012. – vol. 3, No 2. – P. 204–210.

15. J. Fridrich and J. Kodovský. Rich models for steganaly- sis of digital images. IEEE TIFS, 7(3):868–882, 2011.

16. J. Fridrich, J. Kodovský, M. Goljan, and V. Holub. Steganalysis of content-adaptive steganography in spatial domain. In Information Hiding, 13th Int. Conf., volume 6958 of Springer LNCS, pages 102–117, 2011.

17. G. Gül and F. Kurugollu. A new methodology in steganalysis : Breaking highly undetactable steganograpy (HUGO). In Information Hiding, 13th Int. Conf., volume 6958 of Springer LNCS, pages 71–84, 2011.

18. R. Böhme. Improved Statistical Steganalysis Using Models of Heterogeneous Cover Signals. PhD thesis, Faculty of Comp. Sci., TU Dresden, Germany, 2008.

19. J. Fridrich and R. Du. Secure steganographic methods for palette images. In A. Pfitzmann, editor, Information Hiding, 3rd International Workshop, volume 1768 of Lecture Notes in Computer Science, pages 47–60, Dresden, Germany, September 29–October 1, 1999. Springer-Verlag, New York.

20. J. Fridrich, T. Pevný, and J. Kodovský. Statistically undetectable JPEG steganography: Dead ends, challenges, and opportunities. In J. Dittmann and J. Fridrich, editors, Proceedings of the 9th ACM Multimedia & Security Workshop, pages 3–14, Dallas, TX, September 20–21, 2007.

21. L. Guo, J. Ni, and Y.-Q. Shi. An efficient JPEG steganographic scheme using uniform embedding. InFourth IEEE International Workshop on Information Forensics and Security, Tenerife, Spain, December 2–5, 2012.

22. V. Holub and J. Fridrich. Designing steganographic distortion using directional filters. In Fourth IEEE International Workshop on Information Forensics and Security, Tenerife, Spain, December 2–5, 2012.

23. F. Huang, J. Huang, and Y.-Q. Shi. New channel selection rule for JPEG steganography. IEEE Transactions on Information Forensics and Security,7(4):1181–1191, August 2012.

A. D. Ker. A capacity result for batch steganography. IEEE Signal Processing Letters, 14(8):525–528, 2007.

A. D. Ker. A fusion of maximal likelihood and structural steganalysis. In T. Furon, F. Cayre,G. Doërr, and P. Bas, editors, Information Hiding, 9th International Workshop, volume 4567 of Lecture Notes in Computer Science, pages 204–219, Saint Malo, France, June 11–13, 2007. Springer-Verlag, Berlin.

24. Westfeld A. Attacks on Steganographic Systems / A. Westfeld, A. Pfitzmann // Information Hiding. Third International Workshop, LNCS 1768, Springer-Verlag Berlin Heidelberg 2000. P. 61–76.

25. Елтышева Е.Ю. Построение стегосистемы на базе растровых изображений с учётом статистики младших бит / Е.Ю. Елтышева, А.Н. Фионов // Вестник СибГУТИ. No1. 2009. С. 67-84.

26. Lee K. Generalised Category Attack - Improving Histogram-Based Attack on JPEG LSB Embedding / K. Lee, A. Westfeld, S. Lee // Information Hiding. 9th International Workshop. Lecture Notes in Computer Science. Vol. 4567. Springer-Verlag, 2007. P. 378-391.

27. Wilfrid J. Introduction to Statistical Analysis / J. Wilfrid, Dixon, F.J. Massey // McGrawHill Book Company, Inc., New York 1957.

28. Fridrich J. Practical steganalysis of digital images – state of the art [Электронный ресурс] / J. Fridrich, M. Goljan. – Режим доступа: http://faculty.ksu.edu.sa/ghazy/Steg/References/ref28.pdf.

29. Fridrich J. Reliable detection of LSB steganography in color and grayscale images [Text] / J. Fridrich, M. Goljan, R. Du. // Proc. of the ACM Workshop on Multimedia and Security. Ottawa, Canada. 2001. October. P. 27-30.

30. Солодуха Р.А. Усовершенствование метода RS-стеганоанализа применением его к группам пикселов различного размера / Р.А. Солодуха, И.В. Атласов // Вестник Воронежского института МВД России. No2. 2012. С. 53-59.

31. Узун И.А. Стеганоанализ цифровых изображений, хранящихся в произвольных форматах / И.А. Узун // Информатика и математические методы в моделировании. Т.2. No3. 2013. С. 179-189.

32. Lyu S. Detecting messages using higher-order statistics and support vector machines [Электронный ресурс] / S. Lyu, H. Farid // Режим доступа: http://hackerzvoice.net/madchat/crypto/stegano/ih02.pdf.

33. Lyu S. Steganalysis using color wavelet statistics and one-class support vector machines [Электронный ресурс] / S. Lyu, H. Farid // Режим доступа: http://www.ists.dartmouth.edu/library/34.pdf.

34. Pevny T. Steganalysis by subtractive pixel adjacency matrix [Текст] / T. Pevny, P. Bas, J. Fridrich // ACM Multimedia and Security Workshop 2009. – Sep, 2009. P. 75-84.

35. Kodovsky J. Modern steganography can detect YASS [Электронный ресурс] / J. Kodovsky, T. Pevny, J. Fridrich // Режим доступа: http://dde.binghamton.edu/kodovsky/pdf/Kod10yass.pdf.

36. Kerckhoffs A. La Cryptographie Militaire [Текст] / A. Kerckhoffs // Journal des Sciences Militaires. 1883. Vol. 9. Jan. P. 5-38.

37. Kodovsky J. Ensemble classifiers for steganalysis of digital media [Электронный ресурс] / J. Kodovsky, J. Fridrich, V. Holub // Режим доступа: http://dde.binghamton.edu/download/ensemble/TIFS-2011-ensemble.pdf.

38. Ranzato, M., Huang, F. J., Boureau, Y.-L., and LeCun, Y., “Unsupervised learning of invariant feature hierarchies with applications to object recognition,” in [IEEE Conference on Computer Vision and Pattern Recognition], 1–8, IEEE (2007).

39. Krizhevsky, A., Sutskever, I., and Hinton, G. E., “Imagenet classification with deep convolutional neural networks,” in [Advances in neural information processing systems], 1097–1105 (2012).

40. Ciresan, D. C., Meier, U., Masci, J., Maria Gambardella, L., and Schmidhuber, J., “Flexible, high performance convolutional neural networks for image classification,” in [IJCAI Proceedings-International Joint Conference on Artificial Intelligence], 22(1), 1237 (2011).

41. Browne, M. and Ghidary, S. S., “Convolutional neural networks for image processing: an application in robot vision,” in [AI 2003: Advances in Artificial Intelligence], 641–652, Springer (2003).

42. Boureau, Y.-L., Ponce, J., and LeCun, Y., “A theoretical analysis of feature pooling in visual recognition,” in [Proceedings of the 27th International Conference on Machine Learning (ICML-10)], 111–118 (2010).

43. Lee, H., Grosse, R., Ranganath, R., and Ng, A. Y., “Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations,” in [Proceedings of the 26th Annual International Conference on Machine Learning], 609–616, ACM (2009).

44. Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R., “Improving neural networks by preventing co-adaptation of feature detectors,” arXiv preprint arXiv:1207.0580 (2012).

45. Описание библиотеки opennn / [Электронный ресурс] / Режим доступа: http://www.opennn.net/

46. Описание библиотеки fann / [Электронный ресурс] / Режим доступа: http://leenissen.dk/fann/wp/

47. Описание библиотеки convnet / [Электронный ресурс] / Режим доступа: http://conv-net.sourceforge.net/doc/

48. Описание библиотеки alglib / [Электронный ресурс] / Режим доступа: http://www.alglib.net/

49. Описание библиотеки cuda-convnet2 / [Электронный ресурс] / Режим доступа: https://github.com/akrizhevsky/cuda-convnet2

50. Описание библиотеки pybrain / [Электронный ресурс] / Режим доступа: http://pybrain.org/

51. Описание библиотеки theano / [Электронный ресурс] / Режим доступа: http://deeplearning.net/software/theano/

52. Описание библиотеки pylearn2 / [Электронный ресурс] / Режим доступа: http://deeplearning.net/software/pylearn2/

53. Описание библиотеки caffe / [Электронный ресурс] / Режим доступа: http://caffe.berkeleyvision.org/

54. Описание библиотеки treading / [Электронный ресурс] / Режим доступа: https://docs.python.org/3/library/threading.html

55. Описание библиотеки gtk+ / [Электронный ресурс] / Режим доступа: https://www.gtk.org/

56. Nair, V. and Hinton, G. E., “Rectified linear units improve restricted boltzmann machines,” in [Proceedings of the 27th International Conference on Machine Learning (ICML-10)], 807–814 (2010).

57. Описание гипервизора KVM / [Электронный ресурс] / Режим доступа: https://www.linux-kvm.org/page/Main_Page

58. Описание драйверов virtio/ [Электронный ресурс] / Режим доступа: http://www.linux-kvm.org/page/Virtio

59. Описание базы BOSSBase / [Электронный ресурс] / Режим доступа: http://agents.fel.cvut.cz/boss/index.php?mode=VIEW&tmpl=materials

60. Pevny, T., Bas, P., and Fridrich, J., “Steganalysis by subtractive pixel adjacency matrix,” IEEE Transactions on information Forensics and Security 5(2), 215–224 (2010).

61. Fridrich, J. and Kodovsky, J., “Rich models for steganalysis of digital images,” IEEE Transactions on Information Forensics and Security 7(3), 868–882 (2012).

Приложение 1. Листинг главного модуля

```
#include <gtk/gtk.h>

static void
activate (GtkApplication* app,
        gpointer        user_data)
{
//Объявление виджетов
  GtkWidget *window;
  GtkWidget *view;
  GtkWidget *button;
  GtkWidget *button_box;
  GtkWidget *gtk_text_view_new (void);
  GtkWidget *dialog;
  GtkWidget *grid;
  GtkWidget *text_view;
  GtkWidget *scrolled_window;

  GtkTextBuffer *buffer;
  GtkFileChooserAction action = GTK_FILE_CHOOSER_ACTION_OPEN;
  gint res;
//Парамеры окна
  window = gtk_application_window_new (app);
  gtk_window_set_title (GTK_WINDOW (window), "Автоматизированная
система анализа изображений");
  gtk_window_set_default_size (GTK_WINDOW (window), 700, 400);
//Диалог открытия файла
  dialog = gtk_file_chooser_dialog_new ("Open File",
                                        parent_window,
                                        action,
```

```
                                _("_Cancel"),
                                GTK_RESPONSE_CANCEL,
                                _("_Open"),
                                GTK_RESPONSE_ACCEPT,
                                NULL);

        res = gtk_dialog_run (GTK_DIALOG (dialog));
        if (res == GTK_RESPONSE_ACCEPT)
          {
            char *filename;
            GtkFileChooser *chooser = GTK_FILE_CHOOSER (dialog);
            filename = gtk_file_chooser_get_filename (chooser);
            open_file (filename);
            g_free (filename);
          }

        gtk_widget_destroy (dialog);
        gtk_label_set_text(GTK_LABEL(label), "Сканирование: ");
        gtk_widget_show (label);

        button_box = gtk_button_box_new
(GTK_ORIENTATION_HORIZONTAL);
        gtk_container_add (GTK_CONTAINER (window), button_box);

        button = gtk_button_new_with_label ("Стоп");
        gtk_container_add (GTK_CONTAINER (button_box), button);

        gtk_widget_show_all (window);

      }
```

```
    grid = gtk_grid_new ();
     gtk_container_add (GTK_CONTAINER (window), grid);

     label = gtk_label_new ("Служебные сообщения:");
     gtk_grid_attach (GTK_GRID (grid), label, 0, 0, 1, 1);

     gtk_widget_set_vexpand (label, TRUE);
     gtk_widget_set_hexpand (label, TRUE);

   buffer = gtk_text_buffer_new (NULL);


    /* Text view is a widget in which can display the text buffer.
     * The line wrapping is set to break lines in between words.
     */
    text_view = gtk_text_view_new_with_buffer (buffer);
    gtk_text_view_set_wrap_mode (GTK_TEXT_VIEW (text_view),
GTK_WRAP_WORD);


    /* Create the scrolled window. Usually NULL is passed for both
parameters so
     * that it creates the horizontal/vertical adjustments automatically. Setting
     * the scrollbar policy to automatic allows the scrollbars to only show up
     * when needed.
     */
    scrolled_window = gtk_scrolled_window_new (NULL, NULL);
    gtk_scrolled_window_set_policy (GTK_SCROLLED_WINDOW
(scrolled_window),
```

```
                                GTK_POLICY_AUTOMATIC,
                                GTK_POLICY_AUTOMATIC);
        /* The function directly below is used to add children to the scrolled
window
         * with scrolling capabilities (e.g text_view), otherwise,
         * gtk_scrolled_window_add_with_viewport() would have been used
         */
        gtk_container_add (GTK_CONTAINER (scrolled_window),
                                        text_view);
        gtk_container_set_border_width (GTK_CONTAINER (scrolled_window),
5);

      int
      main (int    argc,
            char **argv)
      {
        GtkApplication *app;
        int status;

        app = gtk_application_new ("org.gtk.example",
G_APPLICATION_FLAGS_NONE);
        g_signal_connect (app, "activate", G_CALLBACK (activate), NULL);
        status = g_application_run (G_APPLICATION (app), argc, argv);
        g_object_unref (app);

        return status;
      }
```

Приложение 2. Листинг фильтра верхних частот

```
#include<iostream>
#include<opencv2/imgproc/imgproc.hpp>
#include<opencv2/highgui/highgui.hpp>

using namespace std;
using namespace cv;

int reflect(int M, int x)
{
   if(x < 0)
   {
      return -x - 1;
   }
   if(x >= M)
   {
      return 2*M - x - 1;
   }
  return x;
}

int circular(int M, int x)
{
   if (x<0)
      return x+M;
   if(x >= M)
      return x-M;
  return x;
}
```

```
void noBorderProcessing(Mat src, Mat dst, float Kernel[][3])
{

   float sum;
   for(int y = 1; y < src.rows - 1; y++){
      for(int x = 1; x < src.cols - 1; x++){
         sum = 0.0;
         for(int k = -1; k <= 1;k++){
            for(int j = -1; j <=1; j++){
               sum = sum + Kernel[j+1][k+1]*src.at<uchar>(y - j, x - k);
            }
         }
         dst.at<uchar>(y,x) = sum;
      }
   }
}

void refletedIndexing(Mat src, Mat dst, float Kernel[][3])
{
   float sum, x1, y1;
   for(int y = 0; y < src.rows; y++){
      for(int x = 0; x < src.cols; x++){
         sum = 0.0;
         for(int k = -1;k <= 1; k++){
            for(int j = -1;j <= 1; j++ ){
               x1 = reflect(src.cols, x - j);
               y1 = reflect(src.rows, y - k);
               sum = sum + Kernel[j+1][k+1]*src.at<uchar>(y1,x1);
            }
```

```
            }
            dst.at<uchar>(y,x) = sum;
        }
    }
}

void circularIndexing(Mat src, Mat dst, float Kernel[][3])
{
    float sum, x1, y1;
    for(int y = 0; y < src.rows; y++){
        for(int x = 0; x < src.cols; x++){
            sum = 0.0;
            for(int k = -1;k <= 1; k++){
                for(int j = -1;j <= 1; j++ ){
                    x1 = circular(src.cols, x - j);
                    y1 = circular(src.rows, y - k);
                    sum = sum + Kernel[j+1][k+1]*src.at<uchar>(y1,x1);
                }
            }
            dst.at<uchar>(y,x) = sum;
        }
    }
}

int main()
{

     Mat src, dst;
```

```
src = imread("salt.jpg", CV_LOAD_IMAGE_GRAYSCALE);

if( !src.data )
{ return -1; }


float Kernel[3][3] = {
                      {1/9.0, 1/9.0, 1/9.0},
                      {1/9.0, 1/9.0, 1/9.0},
                      {1/9.0, 1/9.0, 1/9.0}
                     };

  dst = src.clone();
  for(int y = 0; y < src.rows; y++)
     for(int x = 0; x < src.cols; x++)
         dst.at<uchar>(y,x) = 0.0;

  circularIndexing(src, dst, Kernel);



  namedWindow("final");
  imshow("final", dst);

  namedWindow("initial");
  imshow("initial", src);

waitKey();
```

```
    return 0;
}
```

Приложение 3. Листинг гауссовой функции активации

```
void GaussianProcess::compute()
 {
  if (!cf->loghyper_changed) return;
  cf->loghyper_changed = false;
  int n = sampleset->size();
  if (n > L.rows()) L.resize(n + initial_L_size, n + initial_L_size);
  for(size_t i = 0; i < sampleset->size(); ++i) {
   for(size_t j = 0; j <= i; ++j) {
    L(i, j) = cf->get(sampleset->x(i), sampleset->x(j));
   }
  }
  solver.compute(K.selfadjointView<Eigen::Lower>());
  L.topLeftCorner(n, n) = L.topLeftCorner(n,
n).selfadjointView<Eigen::Lower>().llt().matrixL();
  alpha_needs_update = true;
 }
```