3. Данные контролируются путем непрерывного сопоставления детекторов с новыми поступлениями. Обнаружения совпадения с детектором рассматривается как изменение в поведении контролируемой системы.

Заключение

В статье даются основные положения системы обнаружения вторжения, основанные на иммунной системе.

Предлагаемый подход имеет явное преимущество по сравнению с другими технологиями по уровню обнаружения вторжений и уровню ложных срабатываний (табл. 1) [3].

Таблица 1. Сравнение с другими технологиями

Система обнаружения вторжений	Способ обнаружения вторжений	Уровень обнаружения вторжений	Уровень ложных срабатываний
mpMAFIA	Обнаружение аномалий	> 96%	< 35%
Snort IDS	Сигнатурный	> 10%	< 50%
Классическая модель иммунной системы	Обнаружение аномалий	> 10%	< 5%
Предлагаемый подход	Обнаружение аномалий	> 97%	< 3%
Предлагаемый подход после адаптации		> 99%	< 1%

Преимущества предлагаемого подхода:

- отсутствие шаблонов известных атак;
- простой алгоритм обучения, в котором необходим только легитимный трафик;
- способность адаптироваться к изменениям характера трафика;
- низкий уровень ложных срабатываний;
- способность детектировать новые типы вторжений;
- простота алгоритмов обнаружения и адаптации.

Обзор выполнен как подготовительный этап разработки тестовой реализации системы обнаружения вторжений, построенной на принципах искусственной иммунной системы, с учетом полученных в процессе обзора результатов.

Литература

1. Гаврилюк С.В., Оныкий Б.Н., Станкевичус А.А. Применение искусственных иммунных систем для защиты сред распределенных вычислений от вредоносного программного обеспечения

2. Дасгупта Д. Искусственные иммунные системы и их применение. М: ФИЗМАТЛИТ, 2006. – 344 с.

3. Котов В.Д. Система обнаружения вторжений на основе технологий искусственных иммунных систем // Материалы докладов Всероссийской научно-технической конференции студентов, аспирантов и молодых ученых ТУСУР-2009, 2009 – 371-381 с.

АЛГОРИТМ СЛОВАРНОГО ПОИСКА ДЛЯ СИСТЕМЫ ОПТИЧЕСКОГО РАСПОЗНАВАНИЯ ТЕКСТОВ НА ОСНОВЕ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

Хаустов П.А., Спицын В.Г.
Томский политехнический университет
634050, Россия, г. Томск, пр–т Ленина, 30
E-mail: eXceibot@sibmail.com

Введение

В системах оптического распознавания текста для улучшения качества распознавания очень часто используют алгоритмы работы со словарями. В качестве входных данных для этих алгоритмов служат результаты оптического распознавания символов в виде некоторой количественной оценки степени принадлежности символа к тому или иному классу. Чаще всего в качестве такой оценки выступают вероятностные оценки.

Предложенный алгоритм словарного поиска

Тривиальным решением является использование взвешенного расстояния Левенштейна, где весами бы являлись вероятностные оценки, а также вероятности корректности распознавания и сегментации, полученные статистическими методами. Фактически, при таком подходе, будет определяться математическое ожидание расстояния Левенштейна для всех возможных вариантов распознавания и сегментации слова. Для определения расстояния Левенштейна используется метод динамического программирования, который, в свою очередь, использует рекуррентное задание функции $F(c_1, c_2)$ – значение функции расстояния Левенштейна для суффиксов двух строк, начинающихся с позиций c_1 и c_2 соответственно.

Существенным недостатком такого подхода является необходимость в нахождении расстояния Левенштейна до каждого из слов в словаре. Стоит учесть, что сложность вычисления расстояния Левенштейна линейно зависит от произведения длины распознаваемого слова и длины словарного слова. Учитывая, что размер словаря может быть достаточно большим, данный алгоритм обладает недостаточным быстродействием – его асимптотическая оценка $O(KL^2)$, где K – количество слов в словаре, L – среднестатистическая длина слова. Однако преимуществом этого алгоритма является то, что он рассматривает каждое слово словаря.

Для того чтобы улучшить быстродействие этого алгоритма, можно воспользоваться тем фактом, что достаточно большое количество слов в большинстве языков имеют общий префикс. Так,

например, в английском языке слова «preposition», «predicate» и «present» имеют одинаковый префикс «pre». Очевидно, для подобных префиксов функцию расстояния Левенштейна считать более одного раза не имеет смысла. Следовательно, рационально использовать следующее улучшение алгоритма. Для хранения словаря использовать префиксное дерево. В префиксном дереве каждому ребру соответствует определенный символ. Каждое слово в этом дереве представлено путем от корня к некоторой вершине, которая является терминальной. Терминальными вершинами являются только вершины, в которых заканчивается путь некоторого слова. Теперь при поиске расстояния Левенштейна будет использоваться не позиция символа в конкретном слове, а вершина префиксного дерева. В таком случае, для всех возможных префиксов значение функции расстояния Левенштейна будет посчитано ровно один раз. Фактически, расстояние Левенштейна будет считаться параллельно для всех слов с одинаковым префиксом.

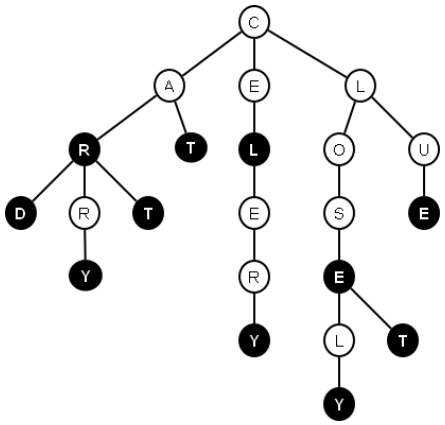

Рис. 1. Пример префиксного дерева

На рисунке 1 можно увидеть префиксное дерево, построенное по словам: «car», «card», «carry», «cart», «cat» , «cel», «celery», «close», «closely», «closet» и «clue». Терминальные вершины, соответствующие этим словам, выделены черным цветом.

При таком подходе функция расстояния Левенштейна принимает следующий вид: F(v, c), где v – вершина в префиксном дереве, а c – текущий символ, для которого будет искаться соответствие при переходе по ребра из вершины v. При переходе по ребрам дерева значение вероятности распознавания соответствующего символа является значением вероятности перехода в вершину, в которую ведет это ребро. Таким образом, наиболее вероятен переход по ребру, соответствующему символу, к которому отнесет соответствующий образ с наибольшей вероятностью. Фактически, функция F(v, c) позволяет найти наиболее вероятный путь в дереве, ведущий к терминальному состоянию.

Очевидно, стоит учесть, что при распознавании могла быть допущена ошибка. Для этого необходимо задать некоторую величину вероятности P_E, с которой система оптического распознавания символов допустила ошибку при распознавании. Значение вероятности P_E можно определить эмпирическим путем. Тогда с вероятностью (P_E / A) необходимо осуществлять переход по каждому из ребер, ведущих из вершины v, где A – количество ребер, выходящих из вершины v.

Операции удаления и добавления символа при поиске расстояния Левенштейна используются для учета ошибок сегментации. Вероятность ошибки сегментации P_S можно так же, как и вероятность ошибки при распознавании P_E определить экспериментальным путем.

Можно дать асимптотическую оценку сложности работы полученного алгоритма. Пусть количество вершин в дереве префиксов равно V, а среднестатистическая длина слова в языке равна L, тогда асимптотическая оценка алгоритма равна O(VL). Для реально существующих языков V намного меньше, чем KL (из-за большого количества слов с одинаковыми префиксами), отсюда можно сделать вывод, что использование префиксного словарного дерева увеличивает быстродействие алгоритма.

При поиске слова в словаре необходимо учитывать некоторые особенности этого слова. Так, например, необходимо посчитать вероятность того, что первая буква этого слова является заглавной. Если эта вероятность более 0.5 следует пересчитать вероятности принадлежности каждому из классов и при поиске искать то же самое слово, но без учета заглавной буквы. Можно также посчитать математическое ожидание количества заглавных букв, чтобы определять слова, являющиеся аббревиатурами. Стоит отметить, что предыдущее правило не применимо для аббревиатур, что тоже необходимо учесть.

Для того чтобы не искать в словаре слова, которые содержат маленькое количество букв можно заранее посчитать математическое ожидание количества букв в слове. Если это значение недостаточно велико, то словарный поиск осуществлять не имеет смысла. Так, например, нет смысла искать в словаре числа или какие-либо численно-буквенные обозначения. Также можно отбрасывать первый и последний символы, если они являются знаками препинания с достаточно большой вероятностью. Такое достаточно часто случается из-за того, что знаки препинания присоединяются к слову в результате сегментации или при наличии небольшого пиксельного шума в строке с этим словом.

Зачастую из-за ошибок сегментации некоторые слова склеиваются с использованием знаков препинания. Подобное возможно из-за пиксельного шума, который ошибочно воспринимается как точка, запятая, апостроф или двоеточие. В таких случаях, казалось бы, поиск по словарю не имеет смысла, ведь даже информация о реальном коли-

честве символов в слове является утерянной. В таком случае возникает необходимость в алгоритме, способном находить наиболее вероятное разбиение полученной лексемы на словарные слова.

Идея алгоритма предложенного для решения такого рода задачи так же основывается на принципе динамического программирования. Будем считать некоторую последовательность символов словарно-представимой, если она состоит только из знаков препинания или образует слово, которое содержится в словаре. Тогда требуется найти наиболее вероятное разбиение имеющейся лексемы на словарно-представимые последовательности символов. Если применить идею динамического программирования, то для каждого суффикса полученной лексемы можно находить наиболее вероятное разбиение на словарно-представимые последовательности. Для того чтобы найти наиболее вероятное разбиение некоторого суффикса достаточно перебрать все последовательности символов начинающиеся с этой позиции. Для каждой из них найти вероятность P_{dict} того, что эта последовательность является словарно-представимой, умножить ее на вероятность наиболее вероятного разбиения оставшейся суффиксной части лексемы P_{suf} и выбрать из всех этих значений максимум. Таким образом, искомое разбиение для суффикса наибольшей длины и будет являться наиболее вероятным разбиением на словарно-представимые последовательности всей лексемы.

Заключение

В результате проделанной работы был предложен собственный алгоритм словарного поиска для системы оптического распознавания текстов. Для хранения словаря в предложенном алгоритме используется префиксное дерево. Для вычисления наиболее вероятного совпадения в словаре и нахождения наиболее вероятных ошибок сегментации используется метод динамического программирования.

Предложенный алгоритм реализован в системе оптического распознавания текста и протестирован на массиве сильно зашумленных текстов. В результате тестирования было установлено, что за счет данного алгоритма словарного поиска было скорректировано 85,96 % ошибок оптического распознавания символов.

Литература

1. Левенштейн В.И. Двоичные коды с исправлением выпадений, вставок и замещений символов: Доклады Академий Наук СССР / В.И. Левенштейн – М.: Проспект, 2009. - 861 с.

2. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ / Т.Кормен – М.: Издательский дом «Вильямс», 2011. – 1293 с.

3. Learning on the Fly: Font-Free Approaches to Difficult OCR Problems [Электронный ресурс]. Режим доступа: http://vis-www.cs.umass.edu/papers/kae_miller_ICDAR_09.pdf, свободный (15.10.2013)

АЛГОРИТМ РАСПОЗНАВАНИЯ МАРКЕРА ДОПОЛНЕННОЙ РЕАЛЬНОСТИ

Коровкин В.А.
Томский политехнический университет
634050, Россия, г. Томск, пр–т Ленина, 30
E-mail: vitaliy.korovkin@gmail.com

Введение

Дополненная реальность – это особый вид проектов, которые направлены на дополнения реальности любыми виртуальными объектами. Рональд Азума в 1997 году выделил основные положения для систем с дополненной реальностью:

- совмещает виртуальное и реальное;
- работает в режиме реального времени;
- работает только в трехмерном пространстве (3D).

Применения данных систем обширно: от военных технологий до электронной коммерции, издательских технологий и игр. Разработка приложений с использованием технологий дополненной реальности является комплексной задачей. В процессе разработки приходится решать различные проблемы из областей анализа видеопотока в реальном времени, компьютерного зрения, распознавания различных изображений, передача больших объемов данных, способы хранения и обработки таких данных и т.д.

В настоящее время для создания приложений дополненной реальности используются два принципиально-различных подхода: с использованием маркера и безмаркерный. Оба варианта используют алгоритмы «компьютерного зрения» для определения объектов в кадре и их дополнения. В данной статье рассматриваются алгоритмы распознавания для приложений дополненной реальности с маркерами.

Методика распознавания маркера

Маркером может быть любая фигура (объект). Но на данном развитии техники часто накладываются ограничения: разрешения камеры (например, веб-камеры), особенности цветопередачи, освещения. Кроме того, весь процесс работы приложения происходит в реальном времени, то необходимы достаточно мощные вычислительные средства. Для того, чтобы снизить нагрузку выбирается черно-белый маркер простой (например, квадрат-