Министерство науки и высшего образования Российской Федерации

САНКТ–ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО

Сборник докладов студентов высшей школы программной инженерии 2019

Санкт-Петербург
2019

Оглавление

Улучшение алгоритма устранения артефактов изображений, сжатых алгоритмом JPEG

Konstantin. A. Zateshilov[1]

Peter the Great St.Petersburg Polytechnic University
St. Petersburg, Russia
[1]zateshilov.konstantin@yandex.ru

***Аннотация*. Сжатие изображений с потерями обеспечивает высокие коэффициенты сжатия путём удаления информации, которая не имеет существенного значения для зрительного восприятия изображений. Однако сжатие с потерями по своей природе может приводить к появлению нежелательных артефактов, что значительно ухудшает качество изображений. Сжатие с потерями также отрицательно влияет на различные высокоуровневые (поиск изображений по содержанию, классификация объектов на изображениях) и низкоуровневые задачи обработки изображений, которые принимают сжатые изображения в качестве входных данных. В связи с этим актуальной проблемой на сегодняшний день является эффективное устранение артефактов сжатия. В данной работе рассматриваются свёрточные нейронные сети для решения проблемы улучшения качества изображений, сжатых алгоритмом JPEG. Для анализа качественных характеристик сетей реализуется свёрточная сеть L04 с помощью фреймворка Caffe. Путём выбора оптимального алгоритма оптимизации для обучения сети были улучшены её качественные характеристики.**

Ключевые слова: свёрточные нейронные сети; алгоритм сжатия JPEG; устранение артефактов сжатия; фреймворк Caffe; машинное обучение.

I. ВВЕДЕНИЕ

Сжатие изображений с потерями обеспечивает высокие коэффициенты сжатия путём удаления информации, которая не имеет существенного значения для зрительного восприятия изображений. В наше время, сжатие изображений с потерями является необходимым для крупных компаний (например, Twitter и VK), поскольку экономит пропускную способность сети при передаче изображений и место для их хранения. Однако сжатие с потерями по своей природе может приводить к появлению нежелательных артефактов, что значительно ухудшает качество изображений. Сжатие с потерями также отрицательно влияет на различные высокоуровневые (поиск изображений по содержанию, классификация объектов на изображениях) и низкоуровневые задачи обработки изображений, которые принимают сжатые изображения в качестве входных данных. В связи с этим актуальной проблемой на сегодняшний день является эффективное устранение артефактов сжатия.

Основными методами, используемыми в настоящее время для сжатия изображений с потерями, являются JPEG, WebP и JPEG XR. В этой работе основное внимание уделяется методу сжатия JPEG [17] и устранению артефактов, возникающих при сжатии данным алгоритмом.

Для устранения артефактов сжатия был предложен ряд методов. К ним можно отнести метод, выполняющий фильтрацию вдоль границ блока [12]; метод устранения блочных артефактов с помощью ДКП с адаптивной формой [4], являющийся одним из наиболее популярных и др.

Помимо вышеперечисленных методов устранения артефактов, возникающих при сжатии изображений, на практике в большинстве программ для просмотра изображений и видео применяются простые 9 фильтры. Так, фильтр spp из библиотеки ffmpeg устраняет артфефакты путём повторного использования сжатия JPEG для сдвинутых версий уже сжатого изображения и усреднения результатов. Кроме указанных методов без обучения для устранения артефактов изображения могут применяться методы машинного обучения, в частности, свёрточные нейронные сети. Эти методы начали использоваться на практике относительно недавно, но показывают лучшие результаты по качеству по сравнению с классическими методами. Свёрточные нейронные сети успешно используются во многих задачах восстановления изображений: в задаче суперразрешения, которая заключается в увеличении разрешения изображений, а также в задаче шумоподавления и др. Одни из самых эффективных архитектур сверточных нейронных сетей для устранения артефактов сжатия изображений были предложены Донгом в [2] (обозначим эту архитектуру AR-CNN) и Свободой в работе [16] (обозначим эту архитектуру L04).

Цель данной работы – улучшить существующий метод устранения артефактов JPEG. Для достижения поставленной цели, необходимо решить ряд задач, к которым относятся следующие:

- изучить особенности применения свёрточных нейронных сетей для устранения артефактов при сжатии изображений алгоритмом JPEG;
- реализовать сеть L04;
- получить качественные характеристики применения сетей L04 и AR-CNN;
- улучшить качественные характеристики сверточных нейронных сетей путем изменения

используемого алгоритма оптимизации при обучении сети.

Исследование построено на теоретической базе, основой которой послужили научные исследования следующих авторов: Донга [2], Свободы [19], Кима [8], Ян Цина [7], Кингма [9] и других.

II. ОБЗОР ИСТОЧНИКОВ

Существует большое количество методов, предназначенных для уменьшения артефактов сжатия, начиная от относительно простых и быстрых фильтров, разработанных вручную, до полностью вероятностных методов восстановления изображений [18] и методов, основанных на передовых подходах к машинному обучению [3].

Простые фильтры удаления артефактов включены в большинство программ для просмотра изображений и видео. Например, библиотека FFmpeg включает простой фильтр постпроцессинга (spp), который просто повторно применяет сжатие JPEG к сдвинутым версиям уже сжатого изображения и усредняет результаты. Фильтр spp использует матрицу квантования (качество сжатия) исходного сжатого изображения, поэтому для декомпрессии, матрица должна быть сохранена вместе с изображением.

Самый успешный метод, основанный на устранении блочных артефактов это, пожалуй, метод устранения блочных артефактов с помощью ДКП с адаптивной формой (Shape-Adaptive DCT – SA-DCT) [4]. Однако, как и большинство методов устранения блочных артефактов, данный метод не может восстанавливать резкие перепады яркости и 39 имеет тенденцию к чересчур сильному сглаживанию мелких деталей изображений.

В области сжатия видео, в стандартах H.264 и H.265, применяются встроенные фильтры (фильтры устранения блочных артефактов и Sample Adaptive Offset фильтры). Однако методы деблокирования в видео, деблокирование SA-DCT (только для оценки параметров), используют значние расположение блоков ДКП. В отличие от этих методов, методы, исследованные в данной работе, способны обрабатывать изображения без такого знания.

В этой работе основное внимание уделяется применению сверточных сетей для улучшения изображений, искаженных артефактами сжатия JPEG. Недавние работы в области машинного обучения показывают, что глубокие свёрточные сети могут эффективно решать различные задачи восстановления изображений [11, 10]. Сегодня основанные на свёрточных сетях методы используются во многих областях компьютерного зрения. Модель AR-CNN, качественные характеристики которой будут дальше исследоваться в данной работе, во многом близка к модели, представленной в работе Донга [2] super-resolution convolutional neural network (SRCNN).

Модель SRCNN создана для решения проблемы суперразрешения. Она разработана опираясь на принцип разреженного кодирования и содержит три слоя: слой выделения признаков, слой многомерного отображения и слой восстановления изображения. Модель с большим количеством слоёв, предложенная Кимом [8] опирается на работу Донга [3], и показывает, что глубокие сети можно обучить для решения проблемы суперразрешения при использовании определённых подхо40 дов. Для обучения использовалась специальная стратегия инициализации весов сети, а также использовалось так называемое остаточное обучение (residual learning), при котором сеть прогнозирует изменение входного изображения вместо того, чтобы предсказывать желаемое изображение напрямую. Остаточное обучение используется и в другой рассматриваемой в данной статье модели L04.

III. ОБУЧЕНИЕ НЕЙРОННЫХ СЕТЕЙ

Алгоритмы обучения нейронных сетей часто используют методы оптимизации. Одной из самых сложных проблем оптимизации в машинном обучении является проблема обучения нейронной сети. Довольно часто приходится затрачивать много времени, использовать большие вычислительные мощности чтобы обучить нейронную сеть решать всего одну задачу. Стремление решить данную проблему обусловило разработку множества специализированных методов обучения нейронных сетей в настоящее время.

Обычно задача обучения нейронной сети сводится к нахождению параметров Θ, которые минимизируют значение функции $J(\Theta)$, в которую обычно входят метрики, вычисленные на обучающей выборке, а также дополнительные слагаемые.

Методы оптимизации, использующиеся в машинном обучении, отличаются от классических методов оптимизации. В большинстве задач машинного обучения мы хотим минимизировать определённую метрику P, которая вычисляется на тестовой выборке. Минимизировать метрику P мы можем только косвенно: минимизируя функцию потерь $J(\Theta)$, ожидая, что это поможет нам минимизировать P. В классических же методах целью является минимизация самой функции $J(\Theta)$. Также алгоритмы оптимизации для машинного обучения имеют особенности, отражающие используемую целевую функцию для данного метода.

В целом, алгоритм обучения нейронной сети состоит в следующем. На начальном этапе инициализируются веса и параметры алгоритма обучения, а затем для всей обучающей выборки (до выполнения критерия остановки обучения) вычисляются выход нейронной сети при заданном входе, функция потерь и градиенты; производится обратное распространение ошибки; обновляются веса с учетом полученных градиентов, а также параметры алгоритма обучения.

В целом, алгоритм обучения нейронной сети состоит в следующем. На начальном этапе инициализируются веса и параметры алгоритма обучения, а затем для всей обучающей выборки (до выполнения критерия остановки обучения) вычисляются выход нейронной сети при заданном входе, функция потерь и градиенты; производится обратное распространение ошибки;

обновляются веса с учетом полученных градиентов, а также параметры алгоритма обучения.

Одним из самых используемых алгоритмов оптимизации в машинном обучении в целом, и при обучении нейронных сетей в частности, является метод стохастического градиента.

Далее рассмотрим алгоритм обучение нейронной сети с применением этого метода.

Входы:

Параметр алгоритма: темп обученияϵ_k; ϵ_k

Обучающая выборка x, y;

Начальные значения весов Θ;

Алгоритм:

while не достигнут критерий остановки **do**:

- выбираем подмножество обучающей выборки размером m:$\{x^{(1)}, \dots, x^{(m)}\}$ с соответствующими откликами $\{y^{(1)}, \dots, y^{(m)}\}$;
- вычисляем градиент $g = \frac{1}{m} \Delta_\Theta \sum_i L\left(f\left(x^{(i)}; \Theta\right), y^{(i)}\right)$, , где $L-$ функция потерь, $f\left(x^{(i)}; \Theta\right)$– предсказанный выход на входных данных $x^{(i)}$, $\Delta_\Theta -$ производная по весам;
- обновляем веса $\Theta = \Theta - \epsilon_k g$, а также обновляем темп обучения ϵ_k.

На практике, темп обучения постепенно снижают в процессе обучения сети. Обычно используется следующая формула до итерации τ:

$$\epsilon_k = (1 - \alpha)\epsilon_0 + \alpha\epsilon_\tau,$$

где $\alpha = \frac{k}{\tau}$.

После итерации τ обычно оставляют постоянное значение. Темп обучения выбирают, используя кривые обучения – зависимость значения функции потерь от времени. При использовании линейной формулы, выбираемыми параметрами являются $\epsilon_\tau, \epsilon_0, \tau$.

Обычно τ выбирают равным числу итераций, за которые через сеть порядка 100 раз проходит обучающая выборка, ϵ_τ выбирают порядка $0.01 \times \epsilon_0$. Основной проблемой является выбор ϵ_0. Если выбрано слишком большое значение, то на кривой обучения будут появляться резкие скачки – значение функции потерь будет значительно возрастать. Если выбран слишком маленький темп обучения, то процесс обучения замедляется, и обучение может остановиться при неприемлемо большом значении функции потерь.

Метод стохастического градиента иногда сходится слишком медленно, поэтому появилось его развитие: метод инерции, который используется чтобы ускорить обучение. В методе инерции накапливается среднее значение прошлых градиентов, и оно влияет на дальнейшее движение метода.

Другим методом оптимизации является метод инерции Нестерова. Этот метод был разработан, основываясь на работах Нестерова [13][14].

Параметр темп обучения влияет на скорость обучения и его выбор является очень трудной задачей. Появилась идея динамически менять темп обучения в ходе процесса обучения. Алгоритм AdaGrad индивидуально адаптирует скорости обучения для всех параметров модели, масштабируя их обратно пропорционально квадратному корню из суммы всех их квадратов значений, полученных ранее. Параметры с наибольшей частной производной функции потерь имеют соответственно быстрое снижение скорости обучения, а параметры с небольшими частными производными имеют относительно небольшое снижение скорости обучения.

Алгоритм RMSProp является модификацией AdaGrad для использования в проблемах невыпуклой оптимизации. Накопление градиентов в данном алгоритме заменено на экспоненциальное взвешенное скользящее среднее.

Adam – ещё один алгоритм с адаптивным темпом обучения. Название «Adam» - сокращение от «адаптивные моменты». Его можно рассматривать как комбинацию RMSProp и метода инерции с некоторыми отличиями.

В настоящее время нет единого мнения о том, какой алгоритм является наилучшим. Исследования показывают, что семейство алгоритмов с адаптивной скоростью обучения ведёт себя достаточно устойчиво. Но выбор алгоритма во многом зависит от конкретной решаемой задачи.

IV. ОПИСАНИЕ МЕТОДА L04

Метод L04 основан на методе AR-CNN, но в нём используются некоторые новые подходы: идея остаточного обучения [8], специальная процедура инициализации параметров сети, а также изменена конфигурация свёрточных слоёв. Свёрточная сеть F состоит из слоя входных данных F_0, свёрточных слоёв F_l, где $0 < l \leq L$, и соответствующих им весов W_lи отклонений b_l. Сеть можно описать следующим образом:

$$F_0(Y) = Y$$
$$F_l(Y) = max(0, W_l * F_{l-1}(Y) + b_l)$$
$$F(Y) = W_L * F_{L-1}(y) + b_L$$

где Y- вход сети, в данном случае искажённое изображение; $F(Y)$ – восстановленное изображение.

В качестве функции потерь используется среднеквадратическая ошибка. Для заданного набора исходных изображений $\{X_i\}$ и соответствующих им сжатых изображений $\{Y_i\}$ значение функции потерь вычисляется следующим образом:

$$L(\Theta) = \frac{1}{n}\sum_{i=1}^{n} \|F(Y_i; \Theta) - X_i\|^2$$

где Θ- это набор параметров модели $\Theta = \{W_1, W_2, W_3, W_4, B_1, B_2, B_3, B_4\}$; n – число изображений.

В большинстве методов восстановления изображений, используется прямое отображение входа на выход, как показано на рис. 1

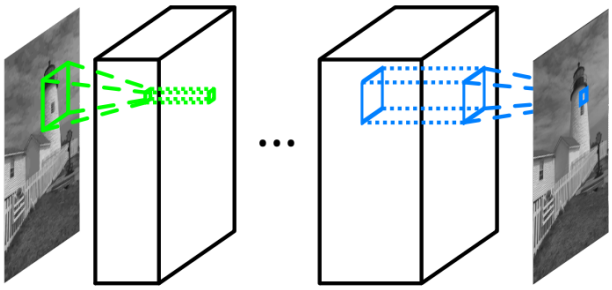


Рис. 1. Прямое отображение в свёрточной нейронной сети.

Такой подход даёт хорошие результаты в некоторых задачах обработки изображений, например, в деконволюции для устранения смазанности текста [6], в задаче суперразрешения [3], а также в задаче устранения артефактов сжатия в работе Донга – AR-CNN. Прямое отображение заставляет сеть передавать всё изображение через все слои. Обучение такой модели, похожей на автоэнкодер, в ситуациях, когда входное и выходное изображения близки, может быть неэффективно, особенно для многослойных сетей. Это может быть одной из основных причин, почему Донг [2] не смог увеличить размер своей сети и требовалось огромное количество итераций для обучения сети.

Остаточное обучение было изначально представлено в работе [8] для решения проблемы суперразрешения, где вход и выход сети достаточно близки. Вместо того, чтобы прогнозировать выходное изображение, сеть с остаточным обучением предсказывает, какие изменения нужно применить ко входному изображению, т.е. она предсказывает $r = y - x$, где y – искажённое изображение, x – выходное изображение с высоким разрешением. На рис. 2 показана работа сети, использующей остаточное обучение.

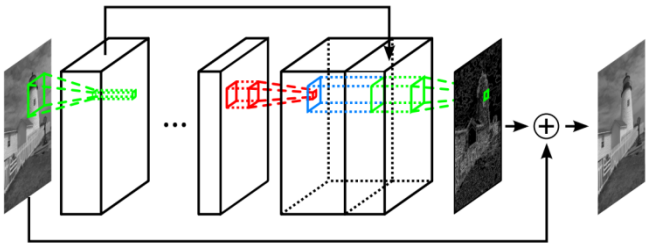


Рис. 2. Остаточное обучение

В качестве метода оптимизации используется метод стохастического градиента.

Описание архитектуры L04 представлено в Таблице 1

ТАБЛИЦА I

Архитектура L04

Слой	1	2	3	4
Размер фильтра	11*11	3*3	3*3	5*5
Количество каналов	48	64	64	1

Проанализируем количество параметров в данной модели. Поскольку модель L04 состоит только из свёрточных слоёв, то число параметров может быть вычислено следующим образом:

$$N = \sum_{i=1}^{d} n_{i-1} * n_i * f_i^2,$$

где N- число параметров; i – номер слоя; d – общее число слоёв; f_i – размер фильтра; n_i – число фильтров на -том слое; n_{i-1}- число входных каналов i-ого слоя. Для модели L04: $d = 4, n_0 = 1, n_1 = 48, n_2 = 64, n_3 = 97, n_4 = 1, f_1 = 11, f_2 = 3, f_3 = 3, f_4 = 5.$

Для инициализации весов первых трёх слоёв используется инициализация Ксавье [5]. Она автоматически определяет дисперсию, в зависимости от числа входов нейрона. Дисперсия распределения для весов определяется следующим образом:

$$Var(W) = \frac{1}{n}$$

где $Var(W)$ – это дисперсия распределения (обычно используется равномерное или гауссовское); n - число входов нейрона.

V. РЕАЛИЗАЦИЯ СЕТИ L04

Для реализации сети L04, описанной Свободой в [19], будет использутся фреймворк Caffe. Сравнение фреймворков для свёрточных нейронных сетей представлено в Таблице 2.

ТАБЛИЦА II

Фреймворк	Лицензия	Основной язык	Интерфейсы	CPU	GPU	Открытый исходный код
Caffe	BSD	C++	Python, MATLAB	✓	✓	✓
cuda-convnet	не указана	C++	Python		✓	✓
Decaf	BSD	Python		✓		✓
Theano/ Pylearn2	BSD	Python		✓	✓	✓
Torch7	BSD	Lua		✓	✓	✓

Для хранения обучающей выборки ис-пользутеся формат HDF5 (Иерархический формат данных). Хранение больших коллекций небольших файлов является стандартной проблемой для файловых систем. В работе [11] указано, что использование хранилища ключ-значение позволяет значительно ускорить чтение обучающей выборки по сравнению с хранением выборки в отдельных файлах. HDF5 позволяет разбить обучающую выборку на отдельные наборы данных для обучения (batch), доступ к

которым становится быстрее. В случае сети L04 размер одного набора равен 64.

Caffe предоставляет готовые слои для реализации сети L04:

- входной слой для чтения данных из хранилища HDF5;
- слой свёртки с задаваемым методом инициализации весов (метод Ксавье);
- слой функции потерь (среднеквадратическая ошибка);
- слой суммирования (для суммирования входа сети с выходом последнего свёрточного слоя для реализации остаточного обучения);

После завершения обучения, для вычисления метрик на тестовой выборке используется скрипт на языке MATLAB. Используется интерфейс Caffe, который позволяет выгрузить параметры обученной модели и использовать их для восстановления изображений из тестовой выборки. В целом архитектура системы представлена на рис. 3

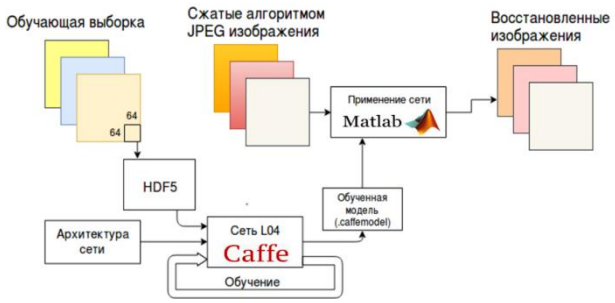


Рис. 3. Архитектура системы

Для ускорения обучения нейронных сетей использовалась платформа Google Cloud Platform. Данная платформа позволяет использовать графические процессоры, такие как NVIDIA Tesla K80, P100 и V100. Использование графического процессора позволяет значительно ускорить обучение, по сравнению с обучением на центральном процессоре. Для развёртывания фреймворка Caffe на сервере используется среда виртуализации NVIDIA-Docker. Docker контейнер содержит в себе все зависимости для данного фреймворка, что позволяет легко его переносить на другие облачные платформы. Надстройка над средой Docker – NVIDIA-Docker позволяет в дополнение к этому распределять вычислительную мощность графического про-цессора между контейнерами. Архитектура системы виртуализации для обучения сетей представлена на рис. 4

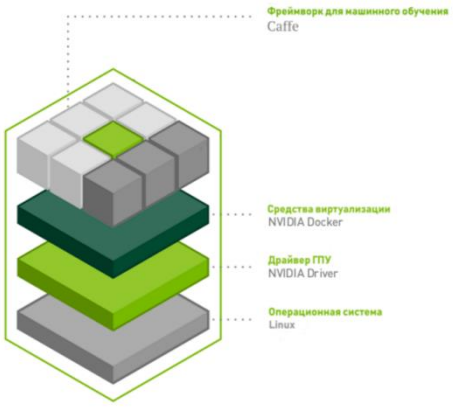


Рис. 4. Архитектура системы виртуализации

VI. ЭКСПЕРИМЕНТАЛЬНЫЕ РЕЗУЛЬТАТЫ

В данной работе используются наборы данных BSDS500 [1] и LIVE1 [15]. Обучение сетей осуществляется на наборе BSDS500, который содержит 400 изображений. Цветные изображения были преобразованы в полутоновые изображения с использованием цветовой моде-ли YCbCr, при этом был сохранён только компонент яркости - Y.

Несмотря на то, что сети могут обрабатывать цветные изображения, оценка сетей происходит по полутоновым изображениям, потому что в данной работе рассматривается подавление артефактов звона и блочных артефактов, а не подавление хроматических искажений. Изображения были сжаты с помощью кодера JPEG MATLAB и разбиты на шесть наборов с разным качеством сжатия. Использовались следующие значения: 10, 20, 30, 40, 50 и 60.

В качестве тестовой выборки используется набор LIVE1, содержащий 29 изображений. Изображения из данного набора также были преобразованы в полутоновые, и сжаты тем же кодером.

Существует несколько показателей для объективной оценки качества изображений. Мы используем пиковое отношение сигнала к шуму (PSNR), пиковое отношение сигнала к шуму с поправкой на блочность (PSNR-B), индекс структурного сходства (SSIM) и визуаль-ное пиковое отношение сигнала к шуму (VPSNR). Достаточно часто метрика PSNR плохо коррелирует с визуальным качеством изображе-ний. Метрика SSIM является развитием метрики PSNR. Она учитывает «восприятие ошибки» благодаря учёту структурного изменения информации. Метрика PSNR-B может дать дополнительную информацию, поскольку в данной работе мы рассматриваем блочные артефакты JPEG.

В некоторых экспериментах указана метрика IPSNR, которая отражает, на сколько увеличилось PSNR по сравнению с PSNR сжатого изображения.

A. *Исследование сетей с использованием разных алгоритмов оптимизации*

Для сетей L04 и AR-CNN найдём наиболее эффективный метод оптимизации. Для всех исследуемых

методов подбирается оптимальный темп обучения с помощью перебора значений по сетке. Результаты обучения сетей представлены в таблицах 3 и 4. Количество итераций при обучении ограничено 150 тысячами для ARCNN и 100 тысячами для L04.

ТАБЛИЦА III РЕЗУЛЬТАТЫ ОБУЧЕНИЯ L04 НА НАБОРЕ LIVE1 С ИСПОЛЬЗОВАНИЕМ РАЗЛИЧНЫХ МЕТОДОВ ОПТИМИЗАЦИИ

Метод оптимизации	L04			
	PSNR	PSNR-B	SSIM	VPSNR
SGD	28.82	28.35	0.817	42.45
RMSProp	28.92	28.46	0.822	42,56
Nesterov	28.42	27.71	0.806	42,04
Adam	**28.96**	**28.57**	**0.821**	**42.58**

ТАБЛИЦА IV РЕЗУЛЬТАТЫ ОБУЧЕНИЯ AR-CNN НА НАБОРЕ LIVE1 С ИСПОЛЬЗОВАНИЕМ РАЗЛИЧНЫХ МЕТОДОВ ОПТИМИЗАЦИИ

Метод оптимизации	AR-CNN			
	PSNR	PSNR-B	SSIM	VPSNR
SGD	28.46	27.94	0.8091	42.07
RMSProp	28.80	28.44	0.8166	42.42
Nesterov	28.66	28.19	0.8132	42.28
Adam	**28.82**	**28.45**	**0.8167**	**42.43**

Для обеих сетей, наилучший результат показывает метод адаптивной инерции (Adam), описанный ранее. Это совпадает с мнением, представленным в статье [9], в которой метод Adam сравнивается с другими методами оптимизации при обучении глубоких свёрточных сетей и показывает хороший результат.

B. Исследование качественных характеристик сетей

Теперь сравним применение простого фильтра постобработки spp из библиотеки ffmpeg, AR-CNN и L04 для восстановления изобра-жений JPEG, сжатых с качеством 10 и 20. Количество итераций при обучении ограничено 250 тысячами для L04 и 500 тысячами для AR-CNN. Для обучения нейронных сетей использовался метод адаптивной инерции (Adam). Результаты представлены в таблицах 5 и 6.

ТАБЛИЦА V РЕЗУЛЬТАТЫ ВОССТАНОВЛЕНИЯ ДЛЯ ИЗОБРАЖЕНИЙ, СЖАТЫХ С КАЧЕСТВОМ 10 НА НАБОРЕ LIVE1

Метод	Q = 10			
	PSNR	PSNR-B	SSIM	VPSNR
Сжатое изображение	27.76	25.33	0.790	41.43
spp	28.29	27.54	0.794	41.90
AR-CNN	28.69	28.30	0.815	42.30
L04	**28.98**	**28.59**	**0.822**	**42.60**

ТАБЛИЦА VI РЕЗУЛЬТАТЫ ВОССТАНОВЛЕНИЯ ДЛЯ ИЗОБРАЖЕНИЙ, СЖАТЫХ С КАЧЕСТВОМ 20 НА НАБОРЕ LIVE1

Метод	Q = 20			
	PSNR	PSNR-B	SSIM	VPSNR
Сжатое изображение	30.07	27.56	0.868	43.73
spp	30.33	28.68	0.867	43.97
AR-CNN	30.43	29.84	0.880	44.07
L04	**31.32**	**30.70**	**0,889**	**44.97**

Сеть L04 превосходит другие методы и при качестве изображений равном 10, и при качестве равном 20. Результаты применения сетей L04 и AR-CNN к изображению показаны на рис. 5.

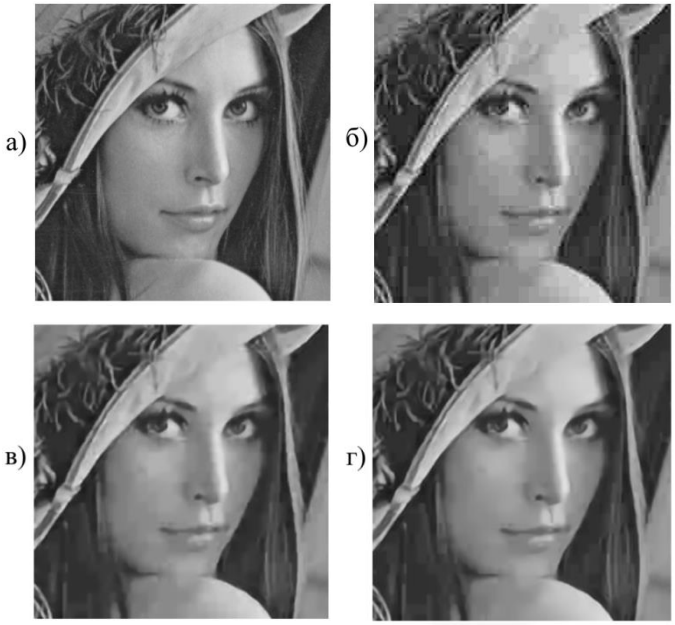


Рис. 5. а) Исходное изображение б) Сжатое JPEG изображение с качеством 10, PSNR = 30.41 в) Изображение, восстановленное с AR-CNN, PSNR = 31.83 г) Изображение, восстановленное с L04, PSNR = 31.92

Исследуем способность сетей "запоминать" изображения, сжатые с разным качеством. Для этого, обучим сеть L04 и AR-CNN на изображениях с качеством сжатия 10, и вычислим метрики качества на изображениях, сжатых при качестве 10, 20, 30, ..., 60, и т.д. чтобы оценить способность одной сети обрабатывать множество различных качеств, мы обучим сети L04 и AR-CNN на обучающей выборке из изображений, сжатых при качестве 10, ..., 60. Обозначим эти сети L04_10-60 и AR-CNN_10-60 соответственно. Результаты представлены на рис. 6 и рис. 7

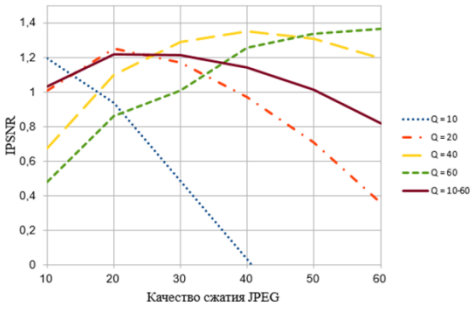


Рис. 6. Зависимость IPSNR от качества сжатия для различных сетей L04

L04_10-60 обеспечивает стабильные результаты для всех качеств сжатия. Однако сети, обученные только на одном качестве, показывают лучший результат для данного качества сжатия. Максимум графиков для моделей

L04_20, L04_40, L04_60 находится при значении качества сжатия, на котором обучались данные модели.

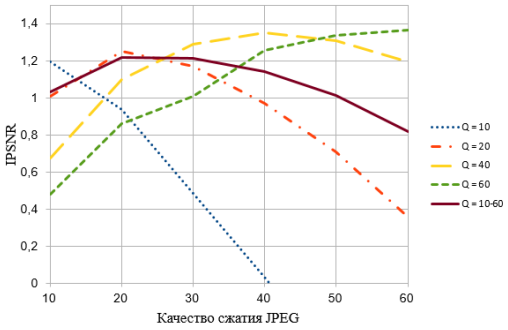


Рис. 7. Зависимость IPSNR от качества сжатия для различных сетей AR-CNN

AR-CNN_10-60 оказалась не способна к эффективному обучению для устранения артефактов сжатия с различным качеством. Это может быть связано с использованием в данной сети прямого отображения из входа сети на выход, а не остаточного обучения, как в сети L04. Сеть с остаточным обучением запоминает только изменения, вызванные сжатием, и обучается эффективнее, чем сеть с прямым отображением.

VII. ЗАКЛЮЧЕНИЕ

В данной статье были рассмотрены специальные архитектуры свёрточных сетей, используемые для эффективного решения проблемы устранения артефактов сжатия изображений алгоритмом JPEG. К ним относится сеть AR-CNN, а также сеть L04, в которой применяется принцип остаточного обучения, позволяющий ускорить обучение сети, и специальная инициализация весов инициализация Ксавье.

Для исследования качественных характеристик сетей в данной работе была реализована сеть L04 с использованием фреймворка Caffe. Создание обучающей выборки на основе изображений осуществляется с помощью скрипта на языке MATLAB, который осуществляет запись в хранилище HDF5 для быстрого чтения примеров из обучающей выборки. Применение обученной сети осуществляется также с помощью скрипта на языке MATLAB. Caffe поддерживает библиотеку CUDA для использования графических процессоров, поэтому для ускорения обучения сети использовалась платформа Google Cloud Platform, которая предоставляет сервера с графическими ускорителями NVIDIA Tesla K80, P100 или V100.

На основе эксперимента можно сделать вывод, что сеть L04 показывает наилучший результат по сравнению с простыми методами и сетью AR-CNN. При исследовании сетей на наличие способности “запоминать” изображения разного качества, было выявлено, что сеть L04, обученная на изображениях с качеством сжатия 10, …, 60, способна показывать стабильные результаты для всех качеств сжатия. Сеть ARCNN оказалась неспособна продемонстрировать такой результат.

В данной работе удалось улучшить качественные характеристики свёрточных нейронных сетей путем изменения используемого алгоритма оптимизации при обучении сети. Наилучший результат показал алгоритм с адаптивным темпом обучения и инерцией Adam.

СПИСОК ИСТОЧНИКОВ

[1] Arbelaez P. Contour de-tection and hierarchical image segmentation/ P. Arbelaez, M. Maire, C. Fowlkes, J. Malik/ / IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2011. - №33(5). – P 898–916.

[2] Dong C. Compression Artifacts Reduction by a Deep Convolutional Network/ C. Dong, Y. Deng, C. Change Loy, X.Tang// Department of Information Engineering, The Chinese University of Hong Kong – 2015.

[3] Dong C. Learning a deep convolutional network for image superresolution/ C. Dong, C. C. Loy, K. He, X. Tang// Lecture Notes in Computer Science (LNCS) – 2014. – P. 184–199.

[4] Foi A. Pointwise shape-adaptive DCT for high-quality denoising and deblocking of gray-scale and color images/ A. Foi, V. Katkovnik, and K. Egiazarian// TIP. – 2007. - №16(5) – P. 1395–1411.

[5] Glorot X. Understanding the difficulty of training deep feedforward neural networks/X. Clorot, Y. Bengio// DIRO, Universite de Montreal – 2016.

[6] Hradis M. Convolutional Neural Networks for Direct Text Deblurring/ M. Hradis, J. Kotera, P. Zemcik, F. Sroubek// HRADIŠ et al.:CNN FOR DIRECT TEXT DEBLURRING – 2015.

[7] Jia Y. Caffe: Convolutional Architecture for Fast Feature Embedding / Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell// UC Berkeley EECS, Berkeley – 2014. – P.4

[8] Kim J. Accurate image super-resolution using very deep convolutional networks/ J. Kim, J. K. Lee, K. M. Lee.// CoRR. – 2015.

[9] Kingma D.P. Adam: a method for stochastic optimization// D.P. Kingma, Ba J.L.// Conference paper at ICLR. – 2015.

[10] Lecun Y. Gradient-Based learning applied to document recognition/ LeCun Y., Bottou L., Benigo Y., Haffner P// Proc/ of the IEEE –1998.

[11] Lim S.-H. An analysis of image storage systems for scalable training of deep neural networks /S.-H. Lim, Young S.R., Patton R.M.// Conference Paper - 2016.

[12] List P. Adaptive deblocking filter/ P. List, A. Joch, J. Lainema, G. Bjontegaard, and M. Karcze-wicz.// IEEE Transactions on Circuits and Systems for Video Technology – 2003. -№ 13(7)/ P. :614–619.

[13] Nesterov, Y. A method of solving a convex programming problem with convergence rate O (1/k^2) / Y. Nesterov// Soviet Mathematics Doklady – 1983. – P. 372–376.

[14] Nesterov, Y. Introductory lectures on convex optimization: a basic course. Applied optimization /Y. Nesterov// Kluwer Academic Publ. – 2004.

[15] Sheikh H.R. LIVE image quality assessment database release /.H.R. Sheikh, Z. Wang, L. Cormack, and A. C. Bovik// - 2015.

[16] Svoboda P. Compression Artifacts Removal Using Convolutional Neural Networks/ P. Svoboda, M. Hradis, D. Barina, P. Zemcik// Brno University of Technology – 2016.

[17] Wallace G.K. The JPEG picture compression standard/ G.K. Wallace// IEEE Transactions on Consumer Electronics. – 1992. - Vol. 38. - No. 1 – P. 18-34.

[18] Wong T.S. A document image model and estimation algorithm for optimized JPEG decompression/ T. S. Wong, C. A. Bouman, I. Pollak, and Z. Fan// IEEE Transactions on Image Processing. – 2009. – No.18 (11). - P. 2518–2535

A new approach to feature generation by complex-valued econometrics and sentiment analysis for stock market prediction[1]

Dmitry Baryev
Peter the Great St.Petersburg Polytechnic University
St. Petersburg, Russia
baryev@yahoo.com

Igor Konovalov
Peter the Great St.Petersburg Polytechnic University
St. Petersburg, Russia
rogee.nok@gmail.com

Nikita Voinov
Peter the Great St.Petersburg Polytechnic University
St. Petersburg, Russia
voinov@ics2.ecd.spbstu.ru

The study was supported by the Russian Foundation for Basic Research, Grant No. 19-010-00610 \ 19 «Theory, Methods and Techniques for Forecasting Economic Development by Autoregressive Models of Complex Variables»

Abstract. The theory of complex-valued econometrics makes it possible to generate qualitatively new features that can be used in machine learning algorithms. Our study reveals the task of determining the long-term dependence of future companies' stock prices from a time-generated feature, i.e., a calculated tonality coefficient gained by methods of semantic analysis of texts from social networks. Data was gathered from the Twitter platform with the use of Big Data ETL-scenarios. The resulting data sets were used to train machine learning algorithms designed to work with Big Data technologies. A semantic coefficient was calculated on the basis of aggregated estimates for each day, with the further application of the methods of complex-valued econometrics. To demonstrate the new approach of feature generation, a complex-valued linear regression model based on the semantic coefficients and stock markets data was constructed. The outcome obtained by the new approach was compared with existing solutions in terms of accuracy. Finally, we demonstrate a possible route for impacting improvements of the existing algorithms for trading strategies using the complex-valued regression.

Keywords: machine learning, sentiment analysis, complex-valued modeling, NLP, Big Data, ETL, Spark, Python, PySpark, Mongo DB, Twitter, stock markets

I. INTRODUCTION

Machine learning is actively used to predict stock prices [1]. Millions of players attempt to maximize profits by selling and buying assets on stock exchanges every day [2]. According to the reports of the U.S. Securities and Exchange Commission [3], around 4,000 brokers are registered in the USA alone. Most of their clients prefer to use automated systems that allow them to buy and sell securities in seconds based on their own predictions. The share of such systems in the market in 2004 was already ¾ of all participants [4]. As a result, the fulfilment of predictions based only on time series becomes an increasingly difficult task. Millions of bidders using various algorithms (trend following strategies [5], weighted average pricing [6], determining local shortages [7], etc.) gradually reduce the role and, accordingly, the possible profit from the dependencies they found. Thus, it becomes especially important to analyse the external information, not directly related to the process of trading on stock exchanges. One of the sources of such information can be the Internet. A text on the Internet has its own tonality – an emotional indicator expressing the author's attitude towards an object of the utterance. The main purpose of this work is to describe a new approach of feature generation that can be used in order to predict future values based on the regression model. This model uses the complex-valued coefficient, characterizing the possible connection between statements in social networks in a natural language.

II. DATA ACQUISITION AND CONVERSION

The three possible sources of data were identified in this research: dedicated forums of brokers trading on financial exchanges, news aggregators, and social networks. The social network Twitter, which has more than 300 million active users [8] and an extensively documented API [9], were identified as a preferable source of information. For the subsequent analysis, a dataset of 1.8 million messages was collected from February 24 to April 20, 2016. It contains the references to "Brexit" – the process of the UK leaving the EU that resulted in a series of economic shocks [10]. For the pre-processing data sources, the open-source tool Hydrator was used to download full texts [11] by identifiers. For establishing an ETL module, the Bonobo framework [12] was used for receiving raw JSON-files. The processed data is loaded into the MongoDB database. The final format of the data is presented in Table 1. In order to support continuous data processing, the Digital Ocean cloud platform [13] was used. The ETL scenarios, which were performing the data processing continuously, were run after the preliminary settings.

Table 1. Data format

A. Record data

id	Record identifier
user	User
date	Publication date
text	Message text
hashtags	Hashtags
retweets	Number of retweets of message

B. Composite field «user»

name	Account name
location	User's location
verified	Verified account
desc	Account description
followers	Number of followers

III. SEMANTIC ANALYSIS MODEL

Big Data is a term which describes data sets with high rates of volume, diversity and speed of appearance. Gigabytes of Twitter messages are an example of Big Data. Such large amounts of information are difficult to be handled with traditional methods of data processing. The Apache Spark [14] open-source framework was chosen as the main data processing tool; it is included in the Apache Hadoop ecosystem [15]. Spark provides multiple access to the data stored in memory for processing information in RAM [16]. Programming language Python was chosen for Spark.

A. Data Preprocessing

The whole process of handling data by machine learning methods must be divided into two parts in order to obtain the resulting set of values for the tonality of messages from Twitter:

- Construction of the model and its training on the ready marked up data set.
- Applying the trained model to the target message set.

Nowadays, several data sets exist on the Internet–Twitter messages classified into groups: positive, negative and neutral. The most extensive collection is one of 1.6 million classified messages made by Stanford University [17]. For further work, data in the form of a special format - DataFrame, was loaded into Spark. Spark DataFrame is a distributed two-dimensional collection organized as a set of named columns. In the subsequent processing of this set by machine learning methods using regular expressions, links to users and other sites were removed from all messages and hashtags were turned into plain text. The pre-processing of the received data was enclosed in a pipeline – a set of the following text processing algorithms.

- Tokenizer performs a process of analyzing the input sequence of characters into recognized groups—lexemes, in order to get identified sequences—tokens;
- StopWordsRemover word filter clears the input from frequently used meaningless words.
- Hashing TF-IDF hashing vectorizer of the TF-IDF statistical measure is used to assess the importance of a word in the context of a document.
- String Indexer is applied to the resulting word frequency vectors.

The entire data set was divided into two subsamples: training and test sets with the ratio of 95% to 5%. The procedure described above was applied to both samples.

B. Training of machine learning model

Logistic regression (LR) [18] was chosen as the main algorithm for the implementation of the machine learning model. The LR-model was first applied in a training set containing both the pre-processing pipeline results and classification of the corresponding messages. The model trained on that sample was then applied to the test set in order to measure its accuracy. The resulting classification, based on the obtained posterior probabilities of the belonging of objects to two classes of tonality, was compared with test indicators. The F-measure (3) – an aggregate criterion that represents the harmonic mean for the precision (1) and recall (2) metrics [19] – was used as the main metric for the model.

$$precision = \frac{TP}{TP + FP} = 0.7774 \quad (1)$$

$$recall = \frac{TP}{TP + FN} = 0.7925 \quad (2)$$

$$F - measure = 2 * \frac{(precision * recall)}{(precision + recall)} = 0.7849 \quad (3)$$

C. Applying the Model

Brexit affects a huge number of people and is closely connected with the sphere of finance – the results of the referendum have already influenced the European market structure.

In order to extend the functionality of Spark, SQLContext was created. It is an entry point for working with Spark [20]. Spark SQL allows performance of data processing. A dedicated data structure DataFrame is used.

Filtered data from the MongoDB database is fed to the input of the trained model. In order to improve the accuracy by using resulting probability, it is possible to define the boundaries of belonging to classes (negative, neutral and positive): 0.2 and 0.8, which is shown in Fig. 1. Negative (0–0.2), neutral (0.2–0.8) and positive (0.8–1).

Figure 1. Boundaries of selection of classes of tonality.

The results of the partitioning are then aggregated using the sum of values of each group for each day. PySpark DataFrame is converted there to Pandas DataFrame for further work within the traditional data size.

IV. Complex-Valued Features Generation

The results of tonality analysis and stock trading were visualized. The dependence of the amount of each key on time is presented in Fig. 2 in the form of a stacked bar graph. It is shown that the amount of data is unstable and has constant outliers. This can be explained by a sharp social agenda of the event. For instance, there was an act of terrorism in Brussels [21] on March 22, 2016, which resulted in a flurry of messages.

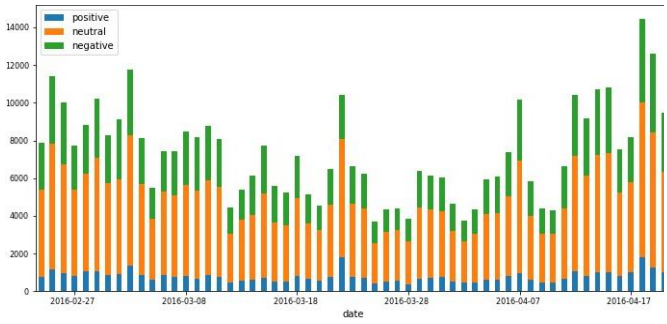


Figure 2. Stacked bar graph of positive, neutral and negative semantics.

Three tonality indicators were aggregated into one coefficient (4), which is proposed by authors:

$$k = \frac{pos - neg}{pos + neut + neg} \qquad (4)$$

As stated above, there is a huge variety of stock trading algorithms. The main purpose of this study is to demonstrate a new approach of feature generation that can be used in order to predict more accurate future values in trading strategies. The main distinguishing feature of this proposed approach is using the complex-valued econometric theory for feature generation.

The complex-valued econometrics theory is a vast field for researches. The main provisions of this theory were described several years ago [22], but this area is still under development.

The received semantic coefficients and relevant indicators of the shares (adjusted closing price and volume per day) were compared for the purposes of the feature generation. The FTSE (Financial Times Stock Exchange 100 Index) shared stock index was taken. This index displays the British stock markets status.

In order to carry out the complex-valued analysis, the volume and the closing price were brought to dimensionless form and centered. The target variable in our analysis is the closing price. The remaining two quantities, the volume and the semantics, were converted into a complex variable, with the former being its real part, and the latter being the imaginary one.

To check the correctness of the choice of values, it is necessary to perform a correlation analysis of the obtained values. However, as shown in the theory of complex-valued econometrics [22], the generally accepted methods for calculating pair correlation are not applicable if there are interrelations between the real and imaginary parts. In this regard, the complex correlation coefficient was calculated (5):

$$r_{cXY} = \frac{\sum_t (y_{rt} + iy_{it})(x_{rt} + ix_{it})}{\sqrt{\sum_t (y_{rt} + iy_{it})^2 \sum_t (x_{rt} + ix_{it})^2}} \qquad (5)$$

In order to obtain a complete picture of the pair dependence, iteratively for the offset from one quantity to another, both coefficients were found: the complex and standard Pearson coefficients, implemented in the NumPy package [24]. The results are presented in Fig. 3.

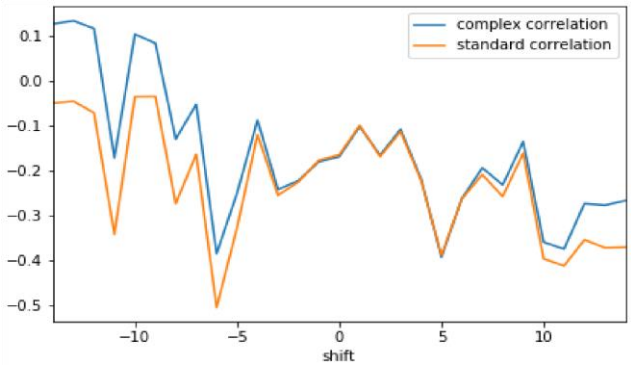


Figure 3. Dependence of complex correlation and Pearson coefficient on time shift.

As can be seen in Fig. 3, the difference between the two coefficients is small, with a slight shift of the complex one. This fact might be explained by the appearance of hidden relationships between stock prices and the volume and semantics manifesting over the time shifts.

In this case, the highest negative correlation is achieved with a slight forward and backward shifts. Thus, according to the data obtained, we can speak about some possible influence of the complex-valued components on the trading process several days before the fact of trading and vice versa – several days after.

Therefore, some relationship between the generated complex value and the target index – the closing price – was proved. In that regard, the generated feature can be estimated in terms of possible practical impact in improving the existing algorithms for trading strategies. To this end, two types of construction were carried out by one of the most common regression analysis models – linear regression.

As a standard LR-model, the object LinearRegression without fit interception from the Scikit-Learn tools library [25] was used. The input parameters for this function are the same two columns, which are the basis of the generated complex-valued feature – the volume and semantic coefficient.

According to the complex-valued econometrics theory [22], in case of a relation between the real and imaginary parts of a complex value, the standard linear regression equation, as well as the standard correlation coefficient, does not work appropriately. In that regard, the following formula (6) was used to build a complex linear regression model. Since all values are centered, the formula for the parameter of the model is reduced to:

$$(b_0 + ib_1) = \frac{\sum_t (y_{rt} + iy_{it})(x_{rt} + ix_{it})}{\sum_t (x_{rt} + ix_{it})^2}$$

Generally, in stock market price predicting analysis, forecast for the next day is the most significant, because longer-term forecasts, in one way or another, will have to be adjusted by incoming values in the future. Therefore, in order to evaluate the generated features, both models will be trained on identical iterative forward moving samples with different lengths.

RMSE (Root-mean-square error) was chosen as the comparison metric for two linear regression models. In this case, RMSE is implemented as the root results of the mean_squared_error [26] function from Scikit-Learn tools library.

Figure 4 shows an example of the comparison of RMSE between the complex and the standard linear regression models. As can be seen from the figure, the complex model error line usually lies below the standard regression model, i.e., the complex model RMSE is smaller. The average RMSE of the complex model is 0.007869, and that of the standard model is 0.009752. Thus, in this case, the complex linear regression model, most likely, will give a better result than the standard regression.

In order to verify the stability of the obtained results, an iterative resizing of the training set should be carried out and the mean RMSE for a current training sample size calculated. A dependency graph of the resulting mean RMSE on the size of the training sample is presented in Fig. 5.

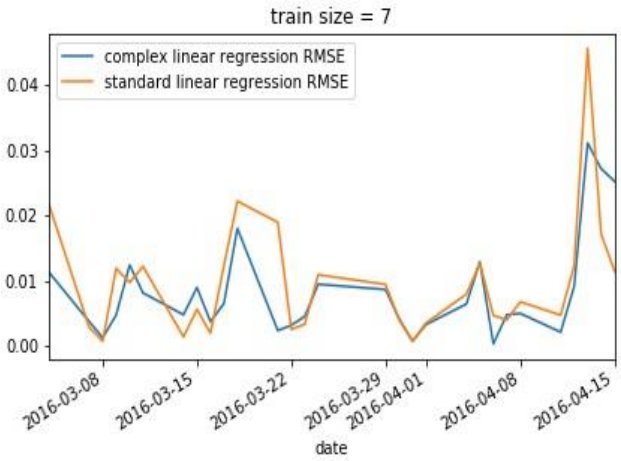


Figure 4. RMSE of complex and standard regression.

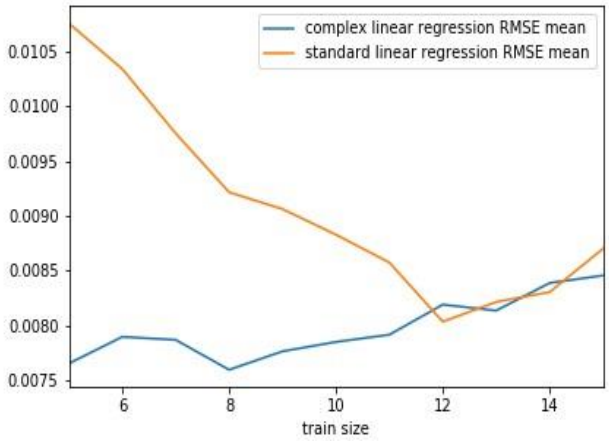


Figure 5. Dependence of mean RMSE of complex and standard regression on train sample size.

Figure 5 shows how RMSE of the complex and standard linear regression models changes depending on the sample size. It is easy to see that a complex regression model based on the generated complex feature appears to be better than the standard linear regression. The difference between RMSE reaches more than 40% with a small size of the training sample (5 elements). With a gradual increase of the training sample size, RMSE of the standard model decreases, while the complex one slowly increases. After that, their RMSEs becomes approximately equally ascending.

In addition to the RMSE-case, all described earlier actions were performed with MSE (mean square error) and MAE (mean absolute error) metrics. All obtained results were similar to the RMSE-case.

CONCLUSION

The paper presents new approaches of generation features that can be used in regression tasks to improve

existing or create new trading strategies. The implementation of the new approaches of feature generation was carried out using the example of determining the long-term dependence of future companies' stocks prices on a timegenerated feature – a calculated tonality coefficient gained by methods of semantic analysis of texts from social networks.

For semantic analysis, Big Data ETL scenarios were used to aggregate and preprocess Twitter messages associated with Brexit, from a micro-blogging platform.

The methods of the complex-valued econometrics theory were applied to the generated complex feature. The complex correlation coefficient between the generated feature and the closing price was calculated. It appears that negative correlations between the feature and the closing price are the strongest several days before and after the trading. We have compared the complex correlation coefficient with the real-valued one and found that the discrepancy in the results of the complex and standard correlation increases with the shift of the feature over time.

The accuracy of the complex and standard linear regression models with the proposed feature was compared on the test data set. We found that the complex-valued regression model based on the generated complex feature performs better in terms of RMSE than the standard linear regression.

REFERENCES

1. Chan, E.: Algorithmic Trading: Winning Strategies and Their Rationale, 656 p. Wiley, Hoboken (2013). ISBN 978-1118460146
2. Harris, L.: Trading and Exchanges: Market Microstructure for Practitioners, 1st edn., 304p. Oxford University Press, Oxford (2002). ISBN 978-0195144703
3. Company Information about Active Broker-Dealers. U.S. Securities and Exchange Commission. https://www.sec.gov/help/foiadocsbdfoiahtm.html
4. MacGregor, A.: As automated trading takes over markets, rational human investors matter even more. http://www.abmac.com/industry-insight/as-automated-trading-takes-overmarkets-rational-human-investors-matter-even-more
5. Merello, S., Ratto, A.P., Oneto, L., Cambria, E.: Predicting future market trends: which is the optimal window? In: Oneto, L., Navarin, N., Sperduti, A., Anguita, D. (eds.) Recent Advances in Big Data and Deep Learning. INNSBDDL 2019. Proceedings of the International Neural Networks Society, vol. 1. Springer, Cham (2020)
6. Yang, R., He, J., Xu, M., Ni, H., Jones, P., Samatova, N.: An intelligent and hybrid weighted fuzzy time series model based on empirical mode decomposition for financial markets forecasting. In: Perner, P. (ed.) Advances in Data Mining. Applications and Theoretical Aspects, ICDM 2018. Lecture Notes in Computer Science, vol. 10933. Springer, Cham (2018)
7. Galimberti, J.K., Suhadolnik, N., Da Silva, S.: Comput. Econ. 50, 393 (2017). https://doi.org/10.1007/s10614-016-9591-2
8. Twitter's Q3 earnings by the numbers. Fast Company. https://www.fastcompany.com/90256723/twitters-q3-earnings-by-the-numbers
9. Makice, K.: Twitter API: Up and Running. Learn How to Build Applications with the Twitter API, 416 p. O'Reilly, Sebastopol (2009). ISBN 978-0596154615
10. 1Brexit and the UK's Public Finances. Institute for Fiscal Studies (IFS Report 116), May 2016. https://www.ifs.org.uk/uploads/publications/comms/r116.pdf
11. Hydrator by samdark. https://github.com/samdark/hydrator
12. Bonobo. Data-processing for humans. https://www.bonobo-project.org
13. Tagliaferri, L.: DigitalOcean eBook: How to Code in Python. DigitalOcean, New York City. ISBN 978-0-9997730-1-7
14. Karau, H., Konwinski, A., Wendell, P., Zaharia, M.: Learning Spark: Lightning-Fast Big Data Analytics, 304 p. O'Reilly, Sebastopol (2015). ISBN 978-5-97060-323-9
15. Frampton, M.: Mastering Apache Spark, 318 p. Packt Publishing Ltd., Birmingham (2015). ISBN 978-1783987146
16. Karau, H.: High-Performance Spark: Best Practices for Scaling and Optimizing Apache Spark, 358 p. O'Reilly, Sebastopol (2017). ISBN 978-1491943205
17. Go, A., Bhayani, R., Huang, L.: Twitter Sentiment Classification using Distant Supervision. https://cs.stanford.edu/people/alecmgo/papers/TwitterDistantSupervision09.pdf
18. Lyman Ott, R., Longnecker, M.T.: An Introduction to Statistical Methods and Data Analysis, 1296 p. Cengage Learning (2015). ISBN 978-1305269477
19. Hackeling, G.: Mastering Machine Learning with Scikit-Learn Paperback, 238 p. Packt Publishing Ltd., Birmingham (2014). ISBN 978-1783988365
20. Gulati, S., Kumar, S.: Apache Spark 2.x for Java Developers: Explore Big Data at Scale Using Java APIs, 350 p. Packt Publishing Ltd., Birmingham (2017). ISBN 978-1787126497
21. Brussels explosions: What we know about airport and metro attacks. BBC News. https:// www.bbc.com/news/world-europe-35869985
22. Sergey, S.: Complex-Valued Modeling in Economics and Finance, 318 p. Springer, New York (2012)
23. FTSE UK Index Series. https://www.ftse.com/products/indices/uk
24. numpy.corrcoef – NumPy v1.16 Manual. https://docs.scipy.org/doc/numpy/reference/generated/numpy.corrcoef.html
25. sklearn.linear_model.LinearRegression – Scikit-Learn 0.20.3 Documentation. https://scikitlearn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html
26. sklearn.metrics.mean_squared_error – Scikit-Learn 0.20.3 Documentation. https://scikitlearn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html

Распределенное инструментирование программного приложения с микросервисной архитектурой по стандарту OpenTracing

Nikita A. Manakov
Peter the Great St.Petersburg Polytechnic University
St. Petersburg, Russia
manakovnikita95@ya.ru, molod@ics2.ecd.spbstu.ru

***Аннотация* — При возникновении проблем, связанных с производительностью программных приложений, достаточно много времени тратится на мониторинг и разбор записей логов. При логировании времени отдельных операций в файл, сложно понять, что привело к вызову этих операций, отследить последовательность действий или смещение во времени одной операции относительно другой в разных сервисах. Инструменты для трассировки позволяют минимизировать ручной труд по поиску узких мест программы. Основной целью работы является рассмотрение спецификации OpenTracing для инструментирования программного комплекса с микросервисной архитектурой. Объектом исследования является программный сервис с микросервисной архитектурой. Основными с точки зрения практической значимости результатами представляются разработанный способ, подход к инструментированию приложений с микросервисной архитектурой.**

Ключевые слова—инструментирование, программный, метод, система, микросервис, приложение, opentracing, распределенная, взаимодействие, трассировка, kafka.

Введение

В области разработки программного обеспечения под инструментированием понимают возможность отслеживания или установления количественных параметров уровня производительности программного продукта, а также возможность диагностировать ошибки и записывать информацию для отслеживания причин их возникновения [1]. Измерения в виде инструкций кода обычно используются для отслеживания работы определенных компонент системы. Когда приложение содержит инструментированный код, им можно управлять при помощи специальных инструментов-утилит., систем анализа, необходимых для оценки производительности приложения.

Благодаря внутреннему параллелизму и асинхронности современных программных приложений распределенная трассировка стала важной частью эффективного мониторинга. Тем не менее, программный инструментарий системы для отслеживания, по крайней мере исторически, был трудоемкой, сложной задачей. Трассировка приносит преимущества отслеживаемости, прозрачности обмена сообщениями приложения по мере его роста до нескольких модулей, когда прослеживается повышенный параллелизм или нетривиальные взаимодействия между клиентами и серверами.

С помощью трассировки можно решить следующие задачи [17]:

- найти узкие места в производительности как внутри одного сервиса, так и во всем дереве выполнения между всеми участвующими сервисами;
- наглядно понять в какой последовательности что вызывается и что происходит когда выполняется операция;
- сбор информации о дереве исполнения для последующего отложенного анализа. На каждом этапе выполнения в трейс можно добавить информацию, которая доступна на данном этапе и дальше разобраться какие входные данные привели к подобному сценарию;
- превращение трейсов в подмножество метрик и дальнейший анализ уже в виде метрик.

Настройка инструментария и решение о том, какой трассировщик использовать, может привести к усложнению проекта. Стандарт OpenTracing меняет это, делая возможным трассировки приложения с минимальными усилиями. OpenTracing состоит из спецификации API, фреймворков и библиотек, реализующие спецификацию, и документации для проекта. OpenTracing позволяет разработчикам добавлять инструментирование в код своего приложения с помощью API, который не привязан к какому-то конкретному поставщику или вендору [3].

Целью данной работы является исследование возможности, способов инструментирования программного комплекса с микросервисной архитектурой. Передача информации между компонентами проекта происходит посредством *HTTP-запросов* и через брокер сообщений *Apache Kafka*. В частности необходимо выявить возможность профилировать:

- все методы, функции в исходном коде приложения с минимальным вмешательством разработчика;
- запросы по протоколу HTTP от одного микросервиса к другому;
- события получения и отправки сообщений в брокер сообщений, в том числе и асинхронных.

Обзор литературы

Анализ исходного кода и его инструментирование являются фундаментальными аспектами процесса разработки программного обеспечения, необходимыми для понимания поведения программы во время исполнения. Понимание структуры кода на высоком уровне детализации (глобальная область программы), а также на более низком уровне (уровень процедур, функций) очень важно для отладки. Высокий уровень абстракции позволяет изучить то, как процедуры обмениваются сообщениями между собой или какие процедуры требуют выделения памяти, что позволяет проводить сложные преобразования исходного кода, ориентированные на улучшение производительности приложения.

Среди всех авторов выделим двух, затрагивающих тему инструментирования программных приложения наиболее полно и разносторонне.

Cardoso, J. M. P в своей работе «Source code analysis and instrumentation» классифицирует анализ кода на *статический* и *динамический* [1]. *Статический анализ* выполняется без запуска приложения, фокусируется на структуре программного кода, нацелен на понимание исходного кода разработчиком, человеком. *Динамический анализ*, с другой стороны, концентрируется на метриках производительности программы. Автор показывает, что существует большое количество метрик, которые представляют интерес, а также то, что профилирование кода, поддерживается как на уровне исходного кода, так и на уровне бинарного представления исходных файлов. Часто используемая и важная метрика — время выполнения. Чтобы найти «узкое место» программы, обычно в код встраивают функции, примитивы, измеряющие время выполнения конкретной области кода. Автор показывает, что с помощью динамического анализа можно получить циклический граф с инструкциями программы, с фактическими вызовами процедур. Польза графов динамического анализа не только в идентификации конкретных вызовов, но также и в динамической информации, такой как размер данных, передаваемых между процедурами.

В своем исследовании Rabiser, R. показывает необходимость мониторинга сложных, неоднородных программных комплексов в силу того, что полное поведение таких программ обнаруживается только во время исполнения. Также автор предлагает несколько подходов к мониторингу, среди которых выделяется подход связанных с инструментированием, выступающий как способ для достижения задач, связанных с мониторингом, динамическим анализом. Мониторинг ресурсов направлен на сбор информации о потреблении вычислительных ресурсов контролируемой системой. К некоторой информации о потреблении ресурсов можно получить прямой доступ из операционной системы или среды выполнения, например из памяти использование виртуальной машины Java (JVM) [9]. Но, как уже было сказано выше, такая информация, как время исполнения процедур, функций, может быть получения только во время выполнения программы с помощью генерации дополнительного кода для измерения времени выполнения определенных участков. Но так добавление такого кода не представляется возможным в современных программных комплексах, то измерение времени достигается за счет инструментирования. В исследовании говорится, что инструменты для инструментирования добавляют избыточное время выполнения в программу, поэтому нужно тщательно подходить к выбору последних. Это одна из главных причин, почему для настоящего исследования был сделан выбор в сторону стандарта OpenTracing.

Обзор спецификации OpenTracing

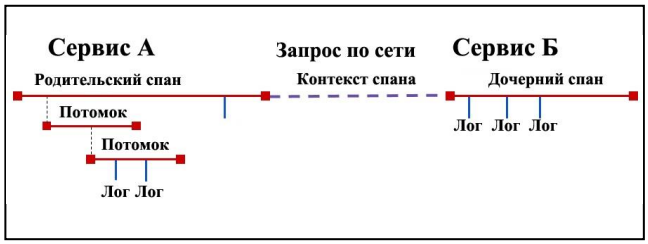
Микросервисы предоставляют мощную архитектуру, но не без проблем, особенно в том, что касается отладки и наблюдения за распределенными транзакциями в сложных системах.

Распределенная трассировка обеспечивает решение для описания и анализа межпроцессных транзакций. Некоторые из случаев использования распределенной трассировки, включают обнаружение аномалий, диагностику проблем состояния системы, распределенное профилирование, распределение ресурсов и моделирование рабочей нагрузки микросервисов.

Терминология спецификации OpenTracing включает в себя следующие понятия:

- Trace — описание транзакции, проходящей через распределенную систему;
- Span — именованная, синхронизированная операция, представляющая часть рабочего процесса. Содержит теги *«ключ: значение»*, а также детализированные структурированные логи с метками времени, прикрепленные к конкретному экземпляру спана;
- Span Context — информация трассировки, которая сопровождает распределенную транзакцию, в том числе когда она передается из сервиса в сервис по сети или через шину сообщений. Контекст спана содержит

идентификатор трассировки, идентификатор спана и любые другие данные, которые система трассировки должна передать в нисходящий сервис.



С точки зрения инструментирования современная программная система выглядит следующим образом:

Рисунок 2: Программная система со стороны распределенного

Дизайн системы содержит в себе четыре важных элемента:

- Программный интерфейс для инструментирования;
- Протокол, использующийся для отправки трассировочной информации вместе с данными приложения в запросах RPC;
- Протокол данных: протокол, по которому передаются данные в систему анализа;
- Система анализа: база данных и интерактивный интерфейс для работы с данными трассировки.

Программный интерфейс OpenTracing предоставляет стандартную независимую от производителя среду для инструментирования. Это означает, что если разработчик хочет опробовать другую систему распределенной трассировки, то вместо того, чтобы повторять весь процесс инструментария для новой системы, можно просто изменить конфигурацию реализации этого программного интерфейса.

Стоит отметить, что инструментирование не является синонимом профилирования, отображающего скорость выполнения работы **каждого** метода в программе [4]. Это правда, что инструментирование может быть приведено к профилированию, но идеология OpenTracing`a говорит о том, что инструментированием занимается разработчик только в тех местах программного кода, которые являются значимыми. Например, имеется точках входа в программу в виде API-идентификатора. Метод, которому делегируется вызов, внутри себя по цепочке вызывает другие методы из других сервисов, и те в свою очередь делают тоже самое. Чтобы найти узкое место программы, инструментируются интересующие места исходного кода. После отображения цепочки вызовов в системе анализа становится понятно, что необходима дополнительная инструментация запроса к базе данных. После система анализа оказывает, что запрос выполняется сравнительно длительное время по сравнению с остальными инструментированными методами.

Также отличие инструментирования от профилирования заключается в информации, которая передается по цепочке вызовов, набор которой также определяется разработчиком [4]. Например, при исключительной ситуации спан может содержать сообщение об ошибке, при журналировании успешной операции сохранения сущности в базу данных сообщение может также быть записано в спан.

Инструментирование программной системы

Необходимо выявить возможность профилирования Java методов в исходном коде приложения без изменения исходного кода методов, возможности инструментирования взаимодействий между сервисами посредством HTTP-запросов, а также получения и отправки событий через брокер сообщений, в том числе и асинхронного получения событий.

A. Профилирование программных методов

Так как было поставлено условие, что инструментация методов должна происходить без вмешательства в исходный код инструментируемого метода, то был выбран подход аспектно-ориентированного программирования — это методика программирования в рамках классовой парадигмы, основанная на понятии аспекта — блока кода, инкапсулирующего сквозное поведение в составе классов и повторно используемых модулей.

Основные понятия АОП:

- Аспект (англ. aspect) — модуль или класс, реализующий сквозную функциональность. Аспект изменяет поведение остального кода, применяя «совет» в точках соединения, определённых некоторым срезом;
- Совет (англ. advice) — средство оформления кода, которое должно быть вызвано из точки соединения. Совет может быть выполнен до, после или вместо точки соединения;
- Точка соединения (англ. join point) — точка в выполняемой программе, где следует применить совет;
- Срез (англ. pointcut) — набор точек соединения. Срез определяет, подходит ли данная точка соединения к данному совету.

Для внедрения сквозной функциональности трассирования был создан аспект, заворачивающий вызов метода в спан. Для того, чтобы аспект мог обработать метод, необходимо получить к нему доступ. Это было сделано через обработку Java аннотации, ставящийся поверх желаемого метода.

Таким образом, для добавления инструментации для желаемых методов необходимо добавить только аннотацию к этим методам.

B. Трассирование HTTP-запросов

OpenTracing получил достаточно широкое распространение, за счет чего имеется огромное сообщество разработчиков и единомышленников, поддерживающих и развивающих данный проект. Имеется множество готовых решений для многих популярных библиотек и технологий. Например, имеется поддержка инструментирования запросов в базу данных *MongoDB*, трассирование запросов клиента *okhttp*, поддержка брокера сообщений *RabbitMQ* и т. д.

Так как в исходном проекте используется фреймворк *Spring*, имеющий внутри себя клиент *RestTemplate* для отправки http-запросов, то существует библиотека *java-spring-web*, позволяющая инструментировать все входящие и исходящие *http-запросы* «из коробки». Для автоматической настройки достаточно лишь добавить зависимость в Java classpath. Происходит это из-за наличия фильтров-интерсепторов — классов, добавляющих обработку запроса перед его отправкой. Внутри фильтра происходит создание спана и контекста, передающегося вместе с полезной нагрузкой на указанный URI.

Так как фреймворк *Spring* позволяет задавать цепочку интерсепторов без непосредственного участия разработчика, то данный фильтр добавляется внутри вспомогательной библиотеки обособленно, что позволяет инструментировать *RestTemplate* без дополнительных настроек.

C. Инструментация взаимодействия через брокер сообщений

Прежде чем перейти к описанию способов инструментирования взаимодействий через брокер сообщений, необходимо описать, произвести введение в архитектуру и терминологию *Apache Kafka*. *Kafka* запускается как кластер на одном или нескольких серверах, которые могут охватывать несколько узлов обработки данных. Кластер *Kafka* хранит записи в категориях, называемых темами (англ. topic). Каждая запись состоит из ключа (англ. key), значения (англ. value) и отметки времени (англ. timestamp).

Кафка имеет четыре основных программных интерфейса, из которых в этой работе понадобятся только два:

- API производителя (англ. producer) позволяет приложению публиковать поток записей в одной или нескольких темах;
- API потребителя (англ. Producer) позволяет приложению подписываться на одну или несколько тем и обрабатывать поток созданных для них записей.

Связь между клиентами и серверами осуществляется с помощью простого, высокопроизводительного, независимого от языка протокола TCP. Имеется Java-клиент для *Kafka*, но клиенты доступны на многих языках высокого уровня.

Тема — это категория или название канала, в который публикуются записи. Темы в *Kafka* всегда многопользовательские, то есть тема может иметь ноль, одного или нескольких потребителей, которые подписываются на нее.

1) Синхронное взаимодействие

Как и для *RestTemplate*`а для *Apache Kafka* существует набор библиотек *java-kafka-client*, позволяющих задать инструментирование всех запросов производителя и потребителя. Для настройки поведения *Kafka* в *Spring*-проектов используется шаблон проектирования декоратор — структурный шаблон проектирования, предназначенный для динамического подключения дополнительного поведения к объекту. За счет данной настройки стандартному потребителю и производителю задаются поведение для инструментирования, оборачивающее все вызовы соответствующих методов в создание спанов и трассировочного контекста.

Информация о трассировочном контексте передается вместе со структурой типа *ключ-значение*. В этой структуре содержаться заголовки, которые способен распознать Jaeger, реализация OpenTracing`a, на сервере, куда поступает вся информация о профилировании со всех сервисов.

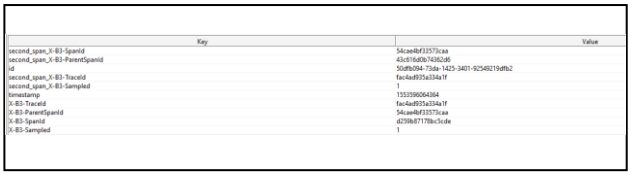
Необходимо пояснить, что каждая реализация спецификации OpenTracing, определяет свой набор заголовков, используемых для передачи и получения трассировочного контекста между сервисами, модулями. Jaeger может быть запущен в режиме совместимости с Zipkin`ом, распределенной системой отслеживания, являющийся одновременно спецификацией и реализацией. Данный фреймворк появился раньше OpenTracing`a и Jaeger`a, то есть могут существовать системы, которые уже были инструментированы им. Чтобы не переделывать функциональность инструментирования для этих сервисов, возможно задать формат заголовков для Jaeger`a, которые будут приниматься Zipkin`ом. Формат следующий:

- X-B3-TraceId — идентификатора трассировки, является единым для всей цепочки вызовов для одной точки входа;
- X-B3-SpanId — идентификатора спана;
- X-B3-ParentSpanId — идентификатор родительского спана, используется для создания связей родитель — потомок;

- X-B3-Sampled — идентификатор выборки инструментируемых запросов, бывает четырех видов:
  - постоянная выборка — обработка или не обработка всех запросов;
  - вероятностная выборка — выбор запроса на основе вероятностной характеристики;
  - выборка на основе алгоритма текущего ведра, задающий количество принимаемых запросов в секунду.

Стратегии выборки используются для высоконагруженных сервисов, для уменьшения не полезной нагрузки на сервис. Например, имеется сервис, обрабатывающий 100 запросов с секунду. Если инструментировать все запросы, поступающие на этот сервис, то быстро произойдет переполнения базы данных, куда загружается информация о трассировке, также сервис начнет работать медленней из-за накладных расходов. Чтобы избежать данных проблем, можно использовать выборку одной десятой всех запросов, что позволит работать сервису в прежнем режиме, а системы анализа будет иметь достаточное количество информации для просмотра.

Каждая запись в *Kafka* содержит заголовки,



представленные на рисунке Рисунок.

Таким образом, потребитель при принятии сообщения распознает заголовки записи, и сможет продолжить цепочку вызовов дальше, не прерывая контекст на обособленные части.

2) Асинхронное взаимодействие

Некоторые микросервисы содержат в себе асинхронную подпись на темы через специальную аннотацию *KafkaListener*. Метод, который работает с данной аннотацией, считывает сообщения из темы в тот момент, когда они там появляются. Так как происходит непрямое получение записей из темы, а также сервис может находится в неактивном состоянии в момент появления записей в теме, то необходима дополнительная настройка для асинхронного взаимодействия. Данная функциональность не работает «из коробки».

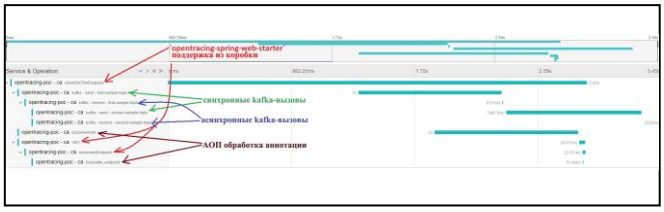
Для того, чтобы была возможность продолжить цепочку спанов при асинхронном считывании из темы брокера сообщений, необходимо выполнить следующие шаги:

- сконфигурировать *KafkaTemplate* клиент, отвечающий за отправку сообщений в темы брокера, чтобы заголовки отправлялись вместе с телом сообщения. Поведение по-умолчанию не записывает заголовки пришедшего сообщения при его дальнейшей отправке;
- активировать функцию принятия записей в режиме «batch», то есть порциями, пакетами;
- в методе с аннотацией *KafkaListener* вместо входного параметра в виде *DTO-объекта* принимать лист записей типа *ConsumerRecord*;
- записать заголовки пришедшей записи в локальное *ThreadLocal* хранилище;
- отправить сообщение через метод *org.springframework.kafka.core.KafkaTemplate#send(org.springframework.messaging.Message<?>),* при этом дополнив сообщение заголовками из локального *ThreadLocal* хранилища.

Таким образом, любой сервис может асинхронно принимать сообщения и трассировочный контекст из темы брокера сообщений, при этом сохраняя консистентность, единообразность цепочки вызовов в системе анализа спанов. При этом у в заголовочной информации появляются заголовки с префиксом *second_span_*, указывающим на наличие родительского контекста в записи.

Заключение

В результате, на выходе данной работы имеются изученная спецификация, стандарт инструментирования OpenTracing, являющийся программным интерфейсом, рекомендацией к реализации, рассмотрены определенные реализации этого интерфейса (Jaeger, Zipkin). Изучены способы передачи контекстной информации между сервисами как в плане типов носителей контекста, так и в плане протоколов



взаимодействия.

Исследованы подходы к инструментации приложений с микросервисной архитектурой, способы добавления функциональности трассировки как для обычных программных методов, единиц функциональности, так и для способов межпроцессного взаимодействия, таких как HTTP-запросы и брокеры сообщений в роли системы с принципом «издатель-подписчик».

На выходе работы имеется инструментированный микросервисный проект. На рисунке ниже показаны спаны, неделимые сущности выполнения, для различных взаимодействий между единицами выполнения исходного кода.

В ходе исследовательской работы были поставлены и выполнены следующие задачи:

- изучена спецификация OpenTracing, рассмотрены принципы работы и идеологии проектирования по данному стандарту;
- рассмотрены способы инструментирования всех единиц выполнения с точки зрения работы компонент микросервиса:
  - методы исходного кода;
  - http-запросы;
  - взаимодействие через брокер сообщений;
- получен работающего программного сервиса с функциональностью трассировки запросов во всей цепочке вызовов.

Источники

1. Cardoso, J. M. P., Coutinho, J. G. F., & Diniz, P. C. (2017). Source code analysis and instrumentation. Embedded Computing for High Performance, 99–135
2. Rabiser, R., Schmid, K., Eichelberger, H., Vierhauser, M., Guinea, S., & Grünbacher, P. (2019). A Domain Analysis of Resource and Requirements Monitoring: Towards a Comprehensive Model of the Software Monitoring Domain. Information and Software Technology
3. Opentracing. [Электронный ресурс]. What is Distributed Tracing?. URI: https://opentracing.io/docs/overview/what-is-tracing/ (дата обращения: 01.10.19)
4. Cinque, M., Cotroneo, D., Corte, R. D., & Pecchia, A. (2018). A framework for on-line timing error detection in software systems. Future Generation Computer Systems
5. Zvara, Z., Szabó, P. G. N., Balázs, B., & Benczúr, A. (2018). Optimizing distributed data stream processing by tracing. Future Generation Computer Systems
6. Dnyaneshwar Panchal (2017). Program Analysis Using Code Instrumentation Techniques
7. Bruno Cabral, Paulo Marques (2019). RAIL: code instrumentation for .NET
8. Garbervetsky, D., Nakhli, C., Yovine, S., & Zorgati, H. (2015). Program Instrumentation and Run-Time Analysis of Scoped Memory in Java. Electronic Notes in Theoretical Computer Science
9. Bechini, A., & Prete, C. A. (2015). Behavior investigation of concurrent Java programs: an approach based on source-code instrumentation. Future Generation Computer Systems
10. Muhammad, T., Halim, Z., & Khan, M. A. (2017). Visualizing trace of Java collection APIs by dynamic bytecode instrumentation. Journal of Visual Languages & Computing
11. Nadeau, D., Ezzati-Jivan, N., & Dagenais, M. R. (2019). Efficient large-scale heterogeneous debugging using dynamic tracing. Journal of Systems Architecture.
12. Angius, E., & Witte, R. (2016). OpenTrace: An Open Source Workbench for Automatic Software Traceability Link Recovery. 2016 19th Working Conference on Reverse Engineering
13. Russell S. Krajec, Ying Li (2018). Application tracing by distributed objectives
14. Zhang, T., Zheng, X., Zhang, Y., Zhao, J., & Li, X. (2016). A declarative approach for Java code instrumentation. Software Quality Journal
15. Zou, B., Yang, M., Benjamin, E.-R., & Yoshikawa, H. (2017). Reliability analysis of Digital Instrumentation and Control software system. Progress in Nuclear Energy
16. Jacob Dormuth, Ben Gelman, Jessica Moore, David Slater (2016). Logical Segmentation of Source Code
17. Jaeger. [Электронный ресурс]. Getting started. URI: https://www.jaegertracing.io/docs/1.12/getting-started/ (дата обращения: 08.10.19)

Methodology of service development with a single Application Programming Interface

Vitaly Monastyrev [1][0000-0001-6770-4481], Pavel Drobintsev [1][0000-0003-1116-7765]

[1] Institute of computer science and technology Peter the Great St. Petersburg Polytechnic University, Saint -Petersburg, Russia,
vit34-95@mail.ru

Abstract. Users of the services can interact with the application using a browser or using mobile devices. The most popular mobile platforms today are iOS and Android. Development of any service includes backend (application logic, database) and frontend (interface) part. Development of frontend part for web, iOS and Android parts is carried out separately, but you can use a single API, which is implemented in the backend part, instead of implementing different backend parts for each platform. In this article we will consider the architecture of a single API and describe the methodology of its development, which allows you to save resources when creating a service.

Keywords: mobile development, web development, Application Programming Interface, backend architecture.

1 Introduction

When developing any service, it is necessary to try to reach the largest possible audience of users. If 10 years ago the service was usually implemented only in the form of a web application, today it is also necessary to develop for mobile platforms. The most popular are Android and iOS. Development for several platforms requires significant resources, so you need to try to optimize this process as much as possible [1]. One way is to implement a backend part that will support a single Application Programming Interface (API) for all platforms.

We will consider creating a single API based on the Representational State Transfer (REST) API architecture. By a single API, we will mean a system built on the REST API architecture, which does not contain duplicate methods for different platforms, but implements only 1 method used by all platforms. The REST API architecture implies that server-side methods are implemented that perform certain actions, for example, return values from the database, change values in the database, load the sent file on the server, and so on. The main problem here is that the implemented methods should work equally well for different clients (Android, iOS, web). For example, different devices have different speed of connection to the Internet, and if for one device ten photos in high resolution from the server are downloaded quickly, then for another device 10 photos from the server need to be shipped in lower quality. This problem could be solved by implementing two different methods, but it is not effective. In the following chapters, we'll look at the methodology of the process of creating a single API that allows you to not implement different methods for different devices.

At the moment, there are no standards for how a single API should be implemented, although there are various studies in the field [20, 21, 22]. Existing studies consider either REST API capabilities, or only a single item - documentation, or a specific tool. In this article we will consider the methodology for using REST API regardless of the framework used. Usually, each company decides for itself how it will implement its API. Also, the team chooses in what language it will do it. This may depend on the goals of the service (C++ is better for high-load systems, NodeJS is better for fast development, etc.), the preferences of the team, the experience of creating previous solutions, the existing code base, and so on. As examples of a single API it is possible to consider Twitter, Vkontakte, Yandex, Google. API of these companies can be used for authorization on other platforms, obtaining information from the account, etc. It is also a source of income for some companies. For example, Yandex [2] and Google [3] provide access to the API of their maps for a certain price.

However, many companies often do not develop a single API that would be optimized for all platforms and release their applications only on one of the platforms. For example, Prisma [4] and Face App [5] are available for iOS and Android, but are not available in the web version, which could attract new users.

When developing a single API, you need to consider many factors to avoid rewriting it in the future, adding duplicate queries, system failures [6] and so on. Since at the moment there is no single methodology for developing single APIs and different authors offer different approaches [7, 8], in this article we have tried to build a methodology on the example of our own development. The main purpose of this article is a General methodology for developing a single API, taking into account the problems that may arise. We offer the following methodology points that will help to solve most of the problems that will arise in the development:

1. Application architecture. The most important part, as it defines the architecture of the entire project.
2. User registration and authorization. Usually, to be able to use all the functions of the service, the user needs to register in it, so it is necessary to provide a mechanism for registration and authorization of the user.
3. Using the same GET/POST requests for different clients. You need to understand what methods are needed for each platform and how they can be combined so as not to implement duplicate or redundant code.
4. Documenting the implemented API. You need to make sure that the developers of the frontend part have documentation and understand why one or the other method is implemented.

5. Backward compatible versions. When updating the API, in addition to taking care of backward compatibility of older versions of the application, you also need to take care of compatibility of all platforms.

Thus, our development methodology is represented by 5 points. It is worth noting that we recommend that the development follow the above sequence of actions. The first step is to determine the structure of the project, then it is convenient to do authorization and registration, as it affects how the 3 point will be implemented. After implementing step 3, you need to document the API. You can then consider backward compatibility for future releases.

2 Application architecture

Consider a single system architecture that uses a single API (figure 1). This architecture is General and allows you to abstract from specific programming languages and frameworks. The architecture of the application discussed in this article is shown in figure 1. It is worth noting that this architecture can be used in the development of almost any service, regardless of its purpose.

Let's consider the presented architecture in more detail. The interaction of the client parts of the application with the server takes place via the REST Protocol using GET/POST requests. The information is transmitted in JSON [9] format. All GET/POST requests implemented on the backend side are the API of our service.

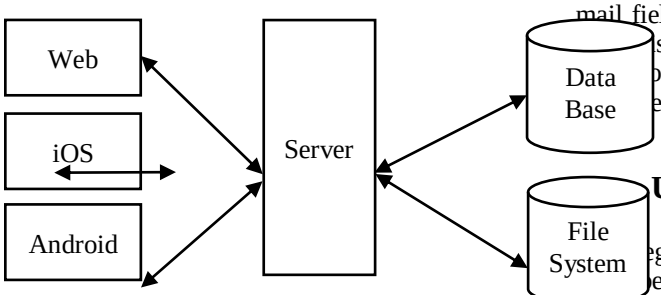


Fig. 1. The architecture of the service.

2.1 Frontend

The first 3 modules (web, iOS and Android) are the frontend part of the app. Web – to view and interact with the service through any browser, iOS – for Apple smartphones and Android for smartphones based on Android.

1. We propose to implement the web part using two submodules-frontend and backend. The Frontend part is responsible for UI rendering and user interaction. The Backend part will work with the single API of the service. This separation simplifies the development process. In addition, in this case it will be easier to correct requests to the service if the API changes. As an example of the framework - for the backend part you can use Laravel [10], and for the frontend part Angular [11].
2. IOS – a client that allows you to work with the service using a device running IOS. The main development languages for iOS are objective-c and swift. XCode was used as the development environment. To send GET/POST requests to the server it is very convenient to use the free libraries SwiftyJSON [12] and Alamofire [13]. It is worth noting that all requests must be executed asynchronously so as not to block the main thread.
3. Android – a client that allows you to work with the service using a device running Android. The most popular development languages are Kotlin and JSON. It is convenient to use gradle [14] as a build system. Requests are sent to the server via REST-Protocol, data format-JSON.

2.2 Backend

As mentioned earlier, the team decides in what language backend will be implemented. It can be C++, Python, NodeJS, Java etc. Later in the article we will give examples implemented in Java + Spring, but similar technologies are available for other programming languages. Any database can also be used as a database - Oracle, Postgresql, MySQL, mongoDB. This depends on the amount and type of data the service is working with. Our team used for their purposes and for the MySQL [15] server connection was used SpringJPA [16]. To store media files, you should use the file system, and store only links to these resources in the database.

Because any service has users, you must create a user table in the database. This table has an id, user name, and mail field that is unique. It also stores the user's password sh for greater security. It is worth noting that the es not store an access token, which is described in tail in the next section.

User registration and authorization

egistration must be supported on any of the 3 clients ed above. Also, users must be able to log in through any of the 3 clients and access the functionality of the application. To solve this problem, you can use JSON Web Token (JWT) [17].

Also, in our case we used Spring filters to filter requests with and without tokens, but almost all other languages have similar frameworks. Filters in the Spring framework filter out all requests that come to the server and do not contain a token. Without a token, only 2 method is available – registration of a new user and getting token. Moreover, (new user registration and token acquisition) are the same for all frontend clients.

Filters in Spring are implemented using Spring Security. In our case, we created a special class WebSecurity extends WebSecurityConfigurerAdapter and added an annotation @EnableWebSecurity. After that, we redefined the method configure as follows:

The authentication filter is used to register a new user, and authorization to access the service of an already registered user. For the new user registration method, this is achieved by sending JSON to a controller whose content looks like this:

```
{
```

```
    "username": "someUsername",
    "password": "somePassword",
    "email":"someEmail@email.com"
}
```

The JSON body is sent from the client to the server via https, which allows traffic to be encrypted. Library to create POST request and use JSON in web applications, iOS and Android. Therefore, we use only 1 method in the server-side controller that allows you to register a new user using a template defined by the JSON body. Upon successful registration, an access token is returned to the user in the response, which may or may not have a lifetime. This token is generated on the server side and encodes inside the user name, which can be decrypted only using a special key stored on the server. This allows you to further accept requests containing an access token and, on its basis, to determine from which user the request came. The token encryption algorithm is chosen by the developer. In this case we used HMAC512.

Typically, the token is stored after it is received and used when sending requests. For example, to the web client a token is stored in the web session, and for the iOS client token is stored using SwiftKeychainWrapper.

If the user is already registered, he can request his token by sending it to the JSON server with the following content:

```
{
    "username": "someUsername",
    "password": "somePassword"
}
```

This information is also transmitted via https. The ability to obtain a token is necessary, since the user can register through one client and save the token on one device, and then try to log in from another device.

When sending any other request needed in the request header, add the following key/value pair – Authorization /"bearer someAccessToken". Thanks to the token, it is possible to accurately determine which user sent the request, regardless of the client used by him. As a result, the problem of registration and authorization of users from different clients is solved by only 2 methods on the server side and the use of JWT tokens for authorization.

It is worth noting that we do not store tokens in the database, as this is not necessary. The user can enter the application from any number of devices and all of them will be issued a token. The most important thing is that we create a token ourselves using a secret word that is stored on the server and the user name is encrypted inside the token, which allows us not to store it in the database.

4 Using the same GET/POST requests for different clients

The next problem that occurs is the use of the same GET/POST requests by different clients. Thus, before developing it is necessary to determine in advance what functionality the service should perform and strive to ensure that the web, Android application and iOS application work according to the same logic and provide the same functionality. In addition, it should be noted that it is also worth striving to ensure that all clients have the most similar interface with the necessary adaptations for specific platforms. This allows users to seamlessly switch from one client to another. Below are examples of the interface of our application. Figure 2 shows the interface on web and the interface on the iOS.

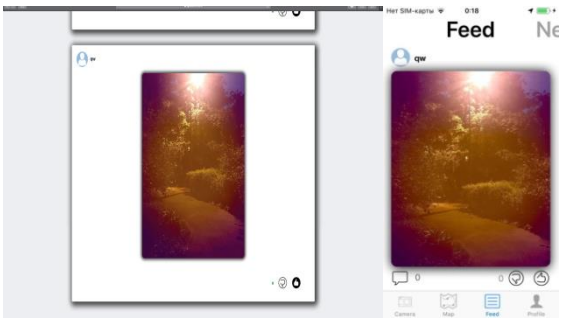


Fig. 2. Web and iOS interface.

When working out the GET/POST requests necessary for the functionality of the application, the following factors should be taken into account: as a rule, the speed of the station Internet is higher than the speed of the mobile Internet, so it can request more information from the server for 1 request. Another factor is the amount of RAM on different mobile devices that is available to the app. On older devices, there may be a situation that when requesting information from the server there is not enough memory to store the result in the cache (for example, if the server requests photos, videos or other files).

One of the most single problems in development is loading information by pages. Pagination should work equally well on iOS, Android, and the web. As part of our methodology, we offer the following universal approach. The information with the request comes to the server in the form of JSON and has the following form:

```
{
    "page": "1",
    "id": "23",
    "pageSize":"10"
}
```

"Page" is the page number that you want to return. "Id" is the number of the record from which the page is counted. This is done to ensure that when the user has reached the end of the list and the server needs to upload the next batch of data does not load those that have already been uploaded, because during this period, new records may appear and then the page will shift. This is the number of records that are contained on 1 page. "PageSize" is set depending on the client used. For example, both iOS and Android allow you to get the size of the available RAM. Based on the average amount of memory occupied by one record, the developer can set the desired value.

Thus, the creation of the single requests to upload information using pagination solves the problem of app usage across different customers and different mobile device. We have only 1 upload method on the server side instead of a bunch of different methods.

5 Documenting the implemented API

Documenting the API is one of the most important parts in development. Clear documentation gives developers an understanding of how to use the provided API. Documentation can reduce or even completely eliminate the time that the development team spends discussing the frontend part and the backend part. In addition, documentation is necessary if you develop only the backend part of the service, and the client part can be developed by third-party developers. This practice is used for example in Telegram [18], Vkontakte [19], etc.

For the possibility of further API support it is recommended to implement methods responsible for different entities in different controllers. So, it will be easier to Orient backend developers and will not cause confusion in the project.

Each team chooses its own method and place for documentation. In our case, the documentation for the single API is based on the following points:

1. Request address. Typically, this is the data transfer Protocol (http or https), server ip address, port, and request name. For example:

   https://127.0.0.1:8080/getBestPhotoArray

2. Request type. For example, it can be GET, POST, PUT, PATCH, etc.
3. Header fields. For example, it can be Authorization, Content-Type, Accept, Cookie, etc.
4. Request JSON fields. These are the fields and the field type of the outgoing request that the backend command has identified as required. It is also worth noting which fields are required and which are not.
5. Response JSON fields. These are the fields and type that will be returned from the server to the client. Mandatory fields are also specified.
6. A description of the purpose of the method. Describes what the query does, what it was created for, and when to use it.

The API storage location is selected depending on who will use the API. If this is an internal command, it is recommended to use some internal portal, for example, in this project we used confluence. If you expect the API to be used by third-party developers, it makes sense to place it in a separate section of your website or put the documentation on git.

6 Backward compatible versions

When developing a single API service, you need to be very careful when modifying old methods. Any renaming of JSON fields can result in users with older versions of clients simply not being able to connect to the service. There are also several possible solutions to the problem.

First you need to understand who uses the service. If these are firm-specific services, you can modify existing methods to ensure that users are delivered a timely version and that all frontend development teams are ready to incorporate the changes into their client application implementations by the due date. But most often we are dealing with services that work with third-party users who can both update their application and not update.

If we work with users who do not always update their application, it is necessary to provide the possibility of backward compatibility with older versions. For example, if in the new version we need to return an additional field to render it to the UI, then we can just add another JSON field to the existing JSON method. This will also allow the front-end development teams to start using the new field when needed.

If the structure of the entire query changes completely, you should implement the new API method and mark the old method as obsolete. You can finally remove the deprecated method after the front-end development teams replace the old method with the new one, and most users update their application. For users who have not updated their application (there should be a small percentage), you can display a message on the download screen that their client version is outdated and no longer supported, they need to update their application.

7 Conclusion

As a result of the work, a methodology for developing a single API was proposed, which provides for 5 points: development of the application architecture, the use of JWT tokens for authorization and registration, optimization of GET/POST requests for different clients by adding additional parameters to requests, maintaining structured documentation and working out the possibilities of backward compatibility of the API

The development methodology was used by our development team and allowed us to reduce the cost of implementing the server part. Instead of 3 backend API developers for different platforms, we only had 1 backend developer. The project size is about 3500 lines of code. It took us about 4 months to write the backend. Thus, we get a development speed of about 30 lines of code per day. Even if we assume that optimization for different platforms would take a thousand lines of code, and the speed would be, for example, 50 lines of code per day, since we do not need to negotiate with all the teams about the architecture, we will get that 3 separate parts would be implemented in 50 days, i.e. a total of 5 months instead of 4 (if 1 programmer). Or should we keep 3 developers instead of 1.

It is worth noting that this metric is not entirely objective. To write a method that will work equally well on all platforms, a person needs more time than a simple method. But in any case, the implementation of one method, instead of for example 3 is more profitable in terms of further support of the code. For example, if in the future we want the method that returns photos to the client to return more and text records, then it will be enough for us to edit and test one method, not three.

Future plans – continue to develop the service with a single API, search for new methods to improve and optimize the development process. Collect feedback from the front-end and back-end development teams.

References

1. Volkova, V. N., Loginova, A. V., Shirokova, S. V., & Kozlovskay, E. A. (2016). Development of the innovative IT-project and managing project human resources. Paper presented at the *Proceedings of the 19th International Conference on Soft Computing and Measurements, SCM 2016*, 470-473. doi:10.1109/SCM.2016.7519816.
2. API Yandex Maps, URL: https://tech.yandex.ru/maps%20/.
3. Google Maps Platform Documentation, URL: https://developers.google.com/maps/documentation/.
4. Prisma, URL: https://prisma-ai.com.
5. Face App, URL: https://www.faceapp.com.
6. Ziniakov, V. Y., Gorodetskiy, A. E., & Tarasova, I. L. (2016). *System failure probability modelling* doi:10.1007/978-3-319-27547-5_19.
7. Raja CSP Raman, Ludovic Dewailly: Building RESTful Web Services with Spring 5 – 2018. – p. 228. ISBN 1788475895.
8. Martin Kalin: Java Web Services: Up and Running – 2009. – p. 320. ISBN 1449365116.
9. Marrs: JSON at Work: Practical Data Integration for the Web – 2017. – p. 320. ISBN 1449358322.
10. Matt Stauffer: Laravel: Up & Running, 2nd Edition – 2019. – p. 544. ISBN 1492041211.
11. Jeremy Wilken: Angular in Action – 2019. – p. 320. ISBN 1617293318.
12. SwiftyJSON URL: https://github.com/SwiftyJSON/SwiftyJSON.
13. Alamofire, URL: https://github.com/Alamofire/Alamofire.
14. Ian F. Darwin: Android Cookbook: Problems and Solutions for Android Developers Android – 2017. – p. 774. ISBN 1449374433.
15. Joel Murach: Murach's MySQL – 2012. – p. 612. ISBN 1890774685.
16. Mike Keith, Merrick Schincariol: Pro JPA 2 (Expert's Voice in Java) – 2013. – p. 508. ISBN 1430249269.
17. JSON Web Tokens, URL: https://jwt.io.
18. Telegram APIs, URL: https://core.telegram.org/.
19. VK Learning API, URL: https://vk.com/dev/first_guide.
20. Paul Adamczyk. REST and Web Services: In Theory and in Practice – 2011. – REST: From Research to Practice pp 35-57. ISBN 978-1-4419-8302-2.
21. Ruben Verborgh. Survey of Semantic Description of REST APIs – 2013. – REST: Advanced Research Topics and Practical Applications pp 69-89. ISBN 978-1-4614-9298-6.
22. Sergio Inzunza. API Documentation – 2018. – WorldCIST'18 2018: Trends and Advances in Information Systems and Technologies pp 229-239. ISBN 978-3-319-77711-5.

Deep learning for analysis cancer dataset for use in computer-aided detection and diagnosis research

Katterine Rodrigue Garzon
Institute of Computer Science and Technology
Peter The Great St. Petersburg Polytechnic University
St. Petersburg, Russian
katterin13@gmail.com

***Abstract*— Computer-aided diagnosis (CAD) is rapidly entering the routine clinical, the use of systems based on artificial intelligence provides tools that support and facilitate the diagnosis of different types of diseases.**

One of the main diseases with the major cause of morbidity and mortality it is Cancer. Cancer is an abnormal growth of cells. There are more than 100 types of cancer, including breast cancer, skin cancer, lung cancer, colon cancer, prostate cancer, and others. The objective of this work is to expose different projects based on Convolutional Neural Network in the diagnosis of breast cancer and make a comparison between the results.

Keywords—breast cancer, image classification, learning (Artificial Intelligence), tumors, medical image processing, cancer classification methods, computer-aided diagnosis system, mass detection method, deep convolutional neural network, neural nets.

V. INTRODUCTION

Cancer is not a simple disease, on the contrary, it is a complete set of different factors that begin when cells grow out of control and form tumors in different organs, which prevents the proper functioning of the human body. There are many types of cancer, we can find similar characteristics between different types, but they are different in the way they grow and spread.

Cancer is a major cause of morbidity and mortality, with approximately 14 million new cases and 8 million cancer-related deaths in 2012, affecting populations in all countries and all regions. These estimates correspond to age-standardized incidence and mortality rates of 182 and 102 per 100 000, respectively. Among men, the five most common sites of cancer diagnosed in 2012 were the lung (16.7% of the total), prostate (15.0%), colorectum (10.0%), stomach (8.5%), and liver (7.5%). Among women, the five most common incident sites of cancer were the breast (25.2% of the total), colorectum (9.2%), lung (8.7%), cervix (7.9%), and stomach (4.8%). More than 60% of the world's cancer cases occur in Africa, Asia, and Central and South America, and these regions account for about 70% of the cancer deaths [1].

Doctors in most cases use two techniques to relieve the diagnosis. A first study is based on the analysis of images, the procedures with images generate graphic representations of the internal regions of the body, the images can be obtained for example by means of X-rays, nuclear exploration, magnetic resonance, among others. The second study is called a biopsy, this is a procedure in which the doctor removes a sample of tissue. Then, a pathologist examines the tissue under a microscope to see if there are cancer cells.

Different specialty clinics, government entities and entities dedicated to the study of cancer have published large banks of images and medical histories. Many of these databases have been classified and analyzed by highly qualified doctors. These databases are the inputs for all applications and developments based on machine learning.

VI. STUDY CASES : BREAST CANER

A breast is constituted by multiple lobes and lobules in which milk is produced. The lobules and lobules are joined by a series of tubes called ducts or milk ducts that lead the milk to the nipple. From birth to adulthood, the breasts undergo more changes than any other organ. Under the influence of female hormones (estrogen and progesterone), the breasts grow during puberty and are influenced in reproductive age by menstrual cycles.

Our organism is constituted by a set of organs, which in turn are formed by cells, which divide in a regular way in order to replace the ages; and thus, maintain the integrity and proper functioning of the various bodies. This process is regulated by a series of mechanisms that indicate a cell to begin to divide and to remain stable.

When these mechanisms are altered in a cell, this and its descendants initiate an uncontrolled division, with time, they give rise to the tumor or nodule. If these cells besides growing without control, acquire the ability to invade surrounding tissues and organs (infiltration) and to move and proliferate in other parts of the body (metastasis) are called malignant tumors, which is what we call cancer [2].

There will be an estimated 18.1 million new cancer cases (17.0 million excluding nonmelanoma skin cancer) and 9.6 million cancer deaths (9.5 million excluding nonmelanoma skin cancer) in 2018. In both sexes combined, lung cancer is the most commonly diagnosed cancer (11.6% of the total cases) and the leading cause of cancer death (18.4% of the total cancer deaths), followed closely by female breast cancer (11.6%).

Among females, breast cancer is the most commonly diagnosed cancer and the leading cause of cancer death [3].

Breast cancer is typically detected either during a screening examination or after a woman notices a lump. When there is a suspicion of breast cancer, the doctor will perform a mammography exam. A mammogram is an

image of the breast taken with X-rays. The image is currently the most popular diagnostic tool, but it is not a simple image to interpret, to make an effective diagnosis is necessary for the knowledge and experience of a highly qualified doctor.

now Computer-aided detection (CADe) and diagnosis (CADx) systems are designed to assist radiologists for mammography interpretation. CADe is used to discover abnormal structures within the mammogram while CADx is used to determine the significance of the discovered abnormality. Despite promising results, current CADe systems are limited by high false-positive rates1, and CADx systems for mammography are not yet approved for clinical use.[4].

VII. Convolutional Neural Network

A Convolutional Neural Network (CNN) is comprised of one or more convolutional layers (often with a subsampling step) and then followed by one or more fully connected layers as in a standard multilayer neural network. The architecture of a CNN is designed to take advantage of the 2D structure of an input image (or other 2D input such as a speech signal). This is achieved with local connections and tied weights followed by some form of pooling which results in translation invariant features. Another benefit of CNNs is that they are easier to train and have many fewer parameters than fully connected networks with the same number of hidden units [10].

CNN had known its great advancement in 2012 through the study conducted by Krizhevsky et al. [11]. They developed a CNN model namely AlexNet which won the ImageNet Large Scale Visual Recognition Competition (ILSVRC) reducing the classification error record with a margin greater than 10%. Thereafter, new CNN models with more layers have been put forward including VGG-Net [12], ResNet [13], [14], GoogLeNet [15], SENet [16], etc[17].

In its most general form, convolution is an operation on two functions of a real-valued argument [18]. The convolution operation is typically denoted with an asterisk:

$$s(t) = (x * w)(t)$$

In convolutional network terminology, the first argument (the function x) to the convolution is often referred to as the input and the second argument (the function w) as the kernel. The output is sometimes referred to as the feature map.

A CNN consists of a number of convolutional and subsampling layers optionally followed by fully connected layers. The input to a convolutional layer is a m x m x r image where m is the height and width of the image and r is the number of channels, e.g. an RGB image has r=3. The convolutional layer will have k filters (or kernels) of size n x n x q where n is smaller than the dimension of the image and q can either be the same as the number of channels r or smaller and may vary for each kernel. The size of the filters gives rise to the locally connected structure which are each convolved with the image to produce k feature maps of size $m-n+1$. Each map is then subsampled typically with mean or max pooling over p x p contiguous regions where p ranges between 2 for small images (e.g. MNIST) and is usually not more than 5 for larger inputs. Either before or after the subsampling layer an additive bias and sigmoidal nonlinearity is applied to each feature map. See figure 1.

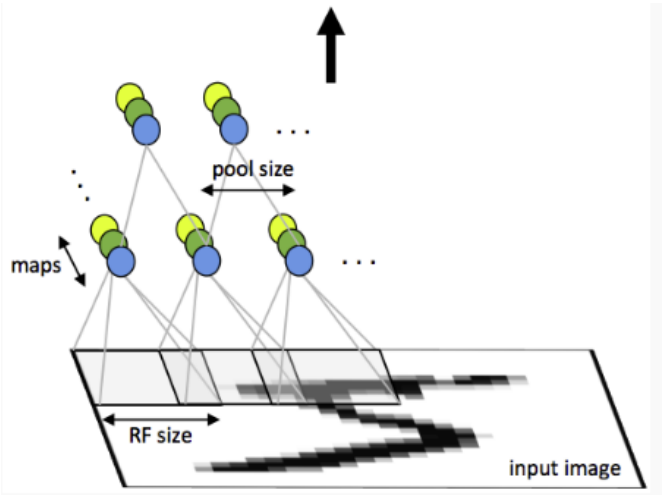


Fig. 1. First layer of a convolutional neural network with pooling. Units of the same color have tied weights and units of different color represent different filter maps.[10]

VIII. CNN Applications in Medical Imaging

BCC breast cancer analysis project

DREAM Breast Cancer Prognosis Challenge (BCC) is a crowdsourced research study for breast cancer prognostic modeling using genome-scale data. BCC promoted to a community of data analysts a data set "MATEBRIC" that contains nearly 2000 breast cancer samples. Within the clinical information of the patient are data such as, for example, age, tumor size and histological grade (see Table 1). For the first phase of training the community participants took 1000 examples from the database, and the remaining samples confirm the set of test data. Additionally, in the final stage of evaluating the models, another set of data called "OsloVal" containing 184 additional samples was taken. The final objective of the project is to measure overall survival (OS).

TABLE I. Clinical Characteristics

Table 1		
Categories	***METABRIC***	***OsloVal***
Cohort size	1981	
Age, years (%)		184
<= 50	21,4	
50-60	22,5	33.1
>=60	56.1	18.5
Tumor size, cm (%)		48.4
<=2	43.3	
2-5	48.2	38.0
>=5	7.5	42.4
NA	1.0	7.1
Node Status (%)		12.5
Node negative	52.3	
1-3 nodes	31.4	49.5
4-9 nodes	11.4	21.2
>= 10 nodes	4.6	9.8
NA	0.3	8.2
ER status (%)		11.3
ER+	76.3	
FR-	23.7	60.9
PR status (%)		39.1
PR+	52.7	
PR-	47.3	21.2
HER2 Copy status (%)		78.8
HER2 amplification	22.1	
HRE2 neutral	72.6	13.6

HRE2 loss	5.0	86.4
NA	0.3	0.0
Tumor grade (%)		0.0
1	8.6	
2	39.1	6.5
3	48.1	37.0
NA	4.2	30.4
		26.1

Table 1. Clinical characteristics of METABRIC and OsloVal data sets.[5]

The predictive value of each model was scored by calculating the concordance index (CI) of predicted death risk compared to OS in a held-out data set, and the CIs were posted on a real-time leaderboard (http://leaderboards.bcc.sagebase.org). The CI is a standard performance measure in survival analysis that quantifies the quality of ranking risk predictors with respect to survival (25). In essence, given two randomly drawn patients, the CI represents the probability that a model will correctly predict which of the two patients will experience an event before the other (for example, a CI of 0.75 for a model means that if two patients are randomly drawn, the model will order their survival correctly three of four times). [5]

As a result of the work in the community as a competition, the participants created more than 1400 models. The best model used a random survival forest trained on the clinical feature data in addition to a genomic instability index derived from the copy number data. The top-scoring model achieved a CI of 0.7408 when trained on the full METABRIC data set and evaluated in the OsloVal data. For comparison, a research version of a 70-gene risk signature was also evaluated in the OsloVal cohort and achieved a CI of 0.60. In addition, two more test models were developed that included only the clinical covariates available for the two data sets. The first model was based on boosted regression [19] and achieved a CI of 0.7001 on the validation data set, whereas the second model used random forest regression [18] and achieved a score of 0.6964. The winning Challenge model achieved a score of 0.7562, significantly higher than the two clinical-only models (Wilcoxon paired test, $P = 6.1 \times 10{-}32$ for both). [5]

Image analysis of biopsies in breast cancer

The following project focuses on the analysis of biopsy images for the diagnosis of breast cancer. A biopsy is a control test that indicates the possibility of malignant tissue growth. Breast tissue biopsies allow the pathologists to histologically assess the microscopic structure and elements of the tissue. The histology allows to distinguish between normal tissue, non-malignant (benign) and malignant lesions and to perform a prognostic evaluation.

The tissue collected during the biopsy is commonly stained with hematoxylin and eosin (H&E) prior to the visual analysis performed by the specialists. During the analysis of the stained tissue, pathologists analyze overall tissue architecture, along with nuclei organization, density, and variability. for example, see figure 2.

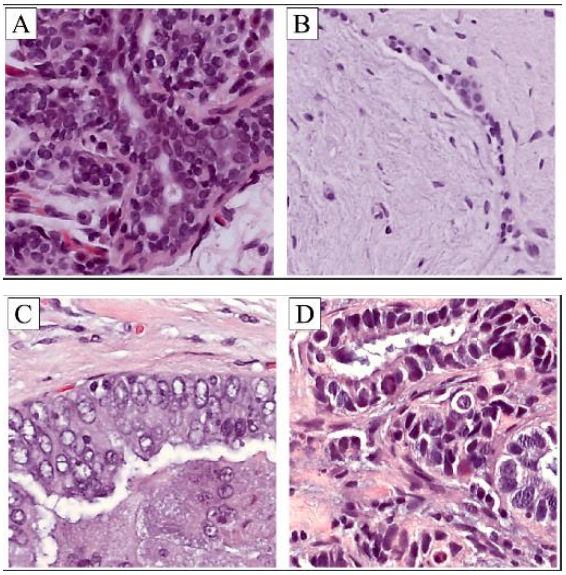


Fig. 2. Examples of microscopy image patches from the used dataset. Nuclei and cytoplasm appear purple and pinkish, respectively, due to the hematoxylin and eosin staining. A normal tissue; B benign abnormality; C malignant carcinoma in situ; D malignant invasive carcinoma.[6]

The database consists of images with a resolution of 2040 × 1536 pixels, uncompressed and with H&E staining. This data set is published at http://www.bioimaging2015.ineb.up.pt/dataset.html. For the design of the project, the data is divided into three sets, a set with 249 images for the training stage, 20 images for the test stage and a set of 16 images as an extended data set.

Before performing the analysis, the images go through a normalization process. First, the colors of the images are converted to optical density (OD) using a logarithmic transformation. Then, singular value decomposition (SVD) is applied to the OD tuples to find the 2D projections with higher variance. The resulting color space transform is then applied to the original image. Finally, the image histogram is stretched so that the dynamic range covers the lower 90% of the data. Figure 3 shows two images before and after normalization. [6]

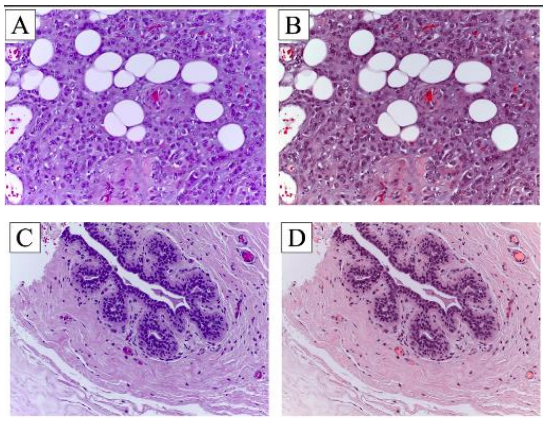


Fig. 3. Histology image normalization. A and C original images; B and D images after normalization.[6]

The procedure to classify one image is as follows. First the original image is divided into twelve contiguous non-overlapping patches. The patch class probability is computed using the patch-wise trained CNN and CNN+SVM classifiers. Then, the image-wise classification

is obtained using one of three different patch probability fusion methods: i) majority voting, where the image label is selected as the most common patch label, ii) maximum probability, where the patch with higher class probability decides the image label and iii) sum of probabilities, where the patch class probabilities are summed and the class with the largest value is assigned. Draws are solved by prioritizing malignant classes using the following order: i) invasive, ii) in situ, iii) benign and iv) normal. [6]

The architecture of the CNN model has in its first layer the entrance of patches, a patch corresponds to a section of the original image, after the normalization of the images these are divided into patches of 512 x 512 pixels. Then in its intermediate layers you explore the characteristics and structures of the nuclei, he final fully-connected network performs the integration of the information for the whole image patch, and provides the final classification. The Output layer is composed of four neurons, corresponding to each of the four classes, that are normalized with a Softmax activation function. See, Figure 4.

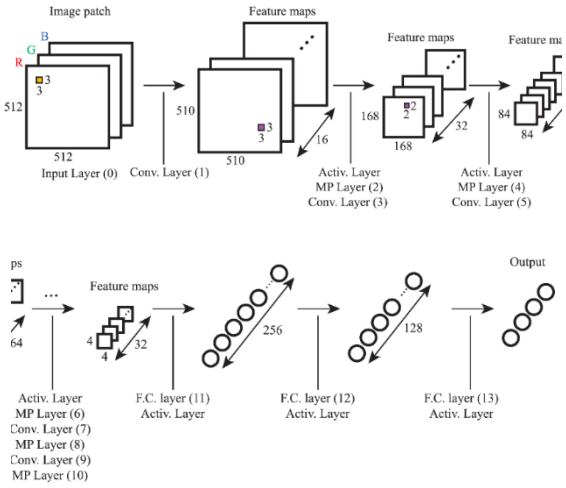


Fig. 4. Convolutional Neural Network architeture [6]

a.

patch ranking results show the overall accuracy (initial plus extended datasets) is 66.7% for the CNN and 65.0% for the CNN+SVM classifier. The performance of our system is lower for the extended dataset due to its increased complexity. The overall accuracy increases when only two classes (non-carcinoma and carcinoma) are considered (77.6% for the CNN and 76.9% for the CNN+SVM). This indicates that the normal/benign and in situ/invasive classes share similar features between them. Furthermore, the proposed system achieves an overall sensitivity of approximately 81% for carcinoma patch-wise classification. Image-wise classification results are shown majority voting shows the best results, achieving an overall accuracy of 77.8% for four classes. These results are constant regardless of using CNN or CNN + SVM for patch-wise classification. In both methods maximum probability is the worst performing method suggesting it is not a suitable strategy for this problem. Regarding binary classification, the overall accuracy increases for both classifiers when compared to the four-class problem. Furthermore, CNN + SVM seems to outperform the CNN model, achieving a total accuracy of 83.3% for the best voting methods. In comparison, CNN's performance is only better for the extended set using majority voting. The lower accuracy of patch-wise classification is explained by the fact that patch labels are obtained from the image labels without any information about the location of the abnormalities. This approach is sub-optimal as, regardless of the image class, normal tissue regions may also be present. As a result, noise is introduced in the training set, contributing to the lower patch-wise accuracy. [6]

TABLE II. RESULTS

Patch-wise accuracy				
Classifier	***No classes***	***Initial***	***Extended***	***Overall***
CNN	4	72.5	59.4	66.7
	2	80.4	74.0	77.6
CNN + SVM	4	72.9	55.2	65.0
	2	82.9	69.3	76.9

Table 2. Patch wise accuracy.[6]

Patch-wise sensitivity					
		Non-carcinoma		***carcinoma***	
Dataset	***Classifier***	***Normal***	***Bening***	***In-situ***	***invasive***
Initial	CNN	69.2		91.7	
		61.7	56.7	83.3	88.3
	CNN + SVM	76.7		89.2	
		65.0	61.7	76.7	88.3
Extended	CNN	81.3		66.7	
		50.0	72.9	58.3	56.3
	CNN + SVM	82.3		56.3	
		54.2	66.7	43.8	56.3
Overall	CNN	74.5		80.6	
		56.4	63.9	72.2	74.1
	CNN + SVM	79.2		74.5	
		60.2	63.9	62.0	74.1

Table 3. Patch-wise sensitivity.[6]

Image-wise accuracy							
Classifier	***Vote***	***4 classes***			***2 classes***		
		Initial	***Exten.***	***Overall***	***Initial***	***Exten.***	***Overall***
CNN	Maj.						
	Max.	80.0	75.0	77.8	80.0	81.3	80.6
	Sum.	80.0	62.0	72.2	80.0	75.0	77.8
		80.0	68.8	75.0	80.0	75.0	77.8
CNN + SVM	Maj.	85.0	68.8	77.8	90.0	75.0	83.3
	Max.	80.0	62.5	72.2	80.0	75.0	77.8
	Sum.	85.0	68.8	77.8	90.0	75.0	83.3

Table 4. Image wise accuracy.[6]

Image-wise sensitivity					
		Non-carcinoma		***carcinoma***	
Dataset	***Classifier***	***Normal***	***Bening***	***In-situ***	***invasive***
Initial	CNN	70		90	
		80	40	100	100
	CNN +	80		100	

	SVM	80	60	100	100
Extended	CNN	50		100	
		75	75	75	75
	CNN + SVM	50		90	
		75	75	50	75
Overall	CNN	61.1		94.4	
		77.8	55.6	88.9	88.9
	CNN + SVM	66.7		95.6	
		77.8	66.7	77.8	88.9

Table 5. Image wise sensitivity.[6]

Image analysis of mammograms in breast cancer

This project works with a database of 400 mammography images, which contains 200 images of malignant masses and 200 images of benign masses for patients between 32 and 74 years of age. As in the other projects, these images have been evaluated by medical specialists in this area. To achieve the objective, the project proposes five steps to follow: breast image preprocessing, mass detection, feature extraction, training data generation, and classifier training. The figure. 5 presents the flowchart of the entire diagnosis process.

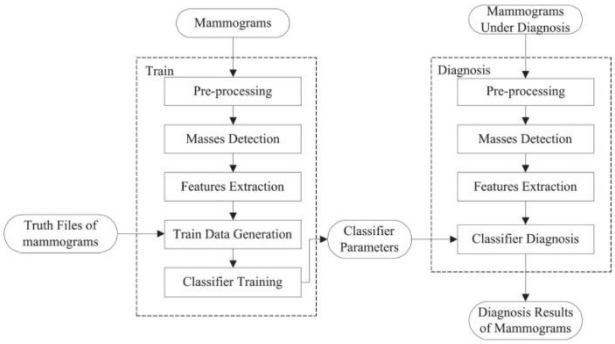
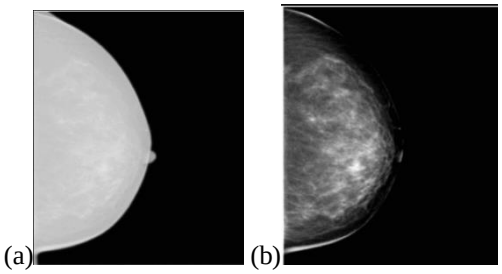


Fig. 5 Flowchart of mass detection process [7]

- in the breast image preprocessing phase, the images are preprocessed by a contrast enhancement algorithm with the objective of increase the contrast between the suspected masses and surrounding tissues. See, Figure 6.



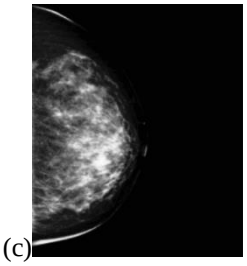


Fig. 6 Images after denoising and enhancement. (a) Initial mammogram. (b) Denoising after. (c) Enhancement after(b) [7]

- In the detection phase, we propose a method that utilizes CNN and US-ELM for feature extraction and clustering, respectively. First, a mammogram will be segmented into several sub-regions. Then, CNN is used to extract features based on each sub-region, followed by utilizing US-ELM to cluster features of sub-regions, which eventually locate the region of the breast tumors.
- In the phase of feature extraction, we design an 8-layer CNN architecture and obtain 20 deep features. In addition, we integrate extra 5 shape features, 5 texture features and 7 density features of the tumor with those deep features to form a fusion deep feature set.
- In the diagnosis phase, we use the fusion deep feature set of each mammogram as an input of ELM for classification. The output directly indicates whether the patient has either a benign or a malignant breast tumor.
- Finally, the experimental results demonstrate that our proposed methods, the sub-regional US-ELM clustering and the ELM classification with fusion deep feature sets, achieve the best performance in the diagnosis of breast cancer.

The conclusions proposed by the authors based on the results obtained are the following:

When the classifier is ELM, the accuracy, sensitivity and specificity of the CNN feature model are the best in the single deep feature model, which shows that the CNN model chosen in this paper is the most suitable method. In the double feature model, the CNN deep feature model combined with texture feature has the best performance in the diagnosis accuracy, sensitivity and specificity.

According to the deep fusion model, comparing the analysis of each evaluation metrics obtained by using ELM and SVM classifier for benign and malignant tumor classification, it can be seen that ELM classifier gives better diagnostic accuracy, sensitivity, and specificity.[7]

Image analysis of mammograms in breast cancer

This project worked with a set of 44,090 mammographic views given by the screening program of the Netherlands (bevolkingsonderzoek midden-west). The system architecture is CNN, but before patch extraction in the CNN system, we segmented all lesions in the training set in order to get the largest possible lesion and choose the patch size with an extra margin resulting in patches of size 250 × 250 (5 × 5 cm). The pixel values in the patches were

scaled using simple min-max scaling, with values calculated over the whole training set. After augmentation, the train set consisted of 334, 752 positive patches and 853, 800 negatives. When combining the train and validation set, this amounts to 379, 632 positive and 931, 640 negative patch. See, figure 7.

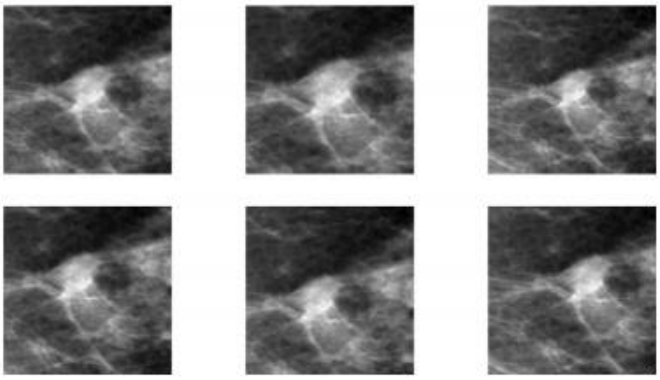


Fig. 7 Examples of scaling and translation of the patches. The top left image is the original patch, the second and third image of the top row examples of the smallest and largest scaling employed. The bottom row indicates the extrema in the range of translation used. [7]

For the second stage classification, we have experimented with several classifiers (SVMs with several different kernels, Gradient Boosted Trees, MLPs) on a validation set, but found in nearly all circumstances the random forest performed similar or better than others.

They used OxfordNet-like architectures (Simonyan and Zisserman, 2014) with 6 convolutional layers of {16, 32, 64, 128, 128} with 3 × 3 kernels and 2 × 2 max-pooling on all but the fourth convolutional layer. Astride of 1 was used in all convolutions. Two fully connected layers of 300 each were added. (See figure 8, 9.)

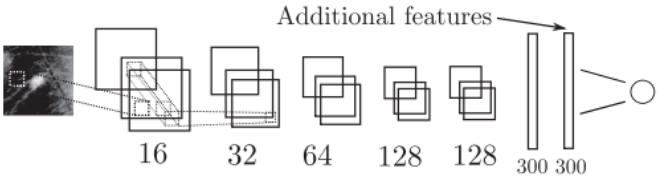


Fig. 8 Illustration of the network architecture, The numbers indicate the amount of kernels used. We employ a scaled-down version of the VGG model. To see the extent to which conventional features can still help, the network is trained fully supervised and the learned features are subsequently extracted from the final layer and concatenated with the manual features and retrained using a second classifier [8]

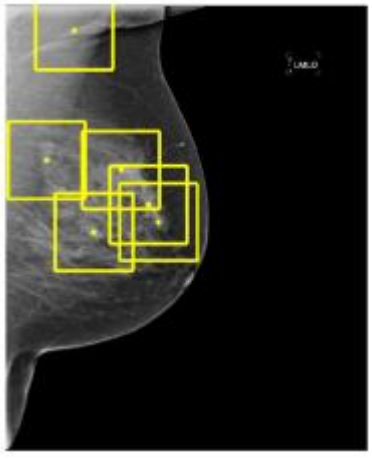


Fig. 9 Illustration of extracted patches for CNN [8]

To first get an understanding of how well each feature set performs individually, we trained different RFs for each feature set and applied them separately to the test set. In all cases, the training procedure as described above was used. AUC values along with a 95% confidence interval, acquired using bootstrapping [9] with 5000 bootstrap samples are shown in Table 2.[8]

TABLE III. OVERVIEW OF RESULT OF INDIVIDUAL FEATURE SETS

Table 2		
Feature group	***Area Under Curve***	***Confidence Interval***
Candidate detect	0.858	[0.827, 0.887]
Contrast	0.787	[0.752, 0.817]
Texture	0.718	[0.681, 0.753]
Geometry	0.753	[0.721, 0.784]
Location	0.686	[0.651, 0.719]
Context	0.816	[0.781, 0.850]
Patient	0.651	[0.612, 0.688]
Equal information	0.892	[0.864, 0.918]
all	0.906	[0.881, 0.929]

Table 6. Overview od result of individual feature sets along the 95% confidence interval (CI) obtained using 50000 bootstraps.[8]

To combine the CNN with other descriptors, we extracted the features from the last fully connected layer and appended the other set. For each augmented patch, the additional features were simply duplicated. Table 3 shows results of the CNN combined with different feature sets, again with confidence interval acquired by bootstrapping with 5000 samples.[8]

TABLE IV. OVERVIEW OF RESULT OF THE CNN COMBINED WITH INDIVIDUAL FEATURE SETS

Table 3		
Feature group	***Area Under Curve***	***Confidence Interval***
CNN only	0.929	[0.897, 0.938]
Candidate detect	0.938	[0.919, 0.955]
Contrast	0.931	[0.91, 0.949]
Texture	0.933	[0.912, 0.950]
Geometry	0.928	[0.907, 0.946]
Location	0.933	[0.913, 0.950]
Context	0.934	[0.914, 0.952]
Patient	0.929	[0.908, 0.947]
all	0.941	[0.922, 0.958]

Table 7. Overview od result of the CNN combined with individual feature sets[8]

From the results in Tables 6 and 7 we can see that individually, apart from the candidate detector, contrast and context are useful features. Although age and screening round are some of the most important risk factors, we do not see clear improvements when added as features, which is slightly disappointing. To get training data, we took negative patches only from normal images, but not only from normal exams, to get as many data points as possible. A possible explanation for the disappointing performance may be that the relation between age and cancer is more difficult to learn in the setting, since it is a relation that exist on an exam level. [8]

CONCLUSION

CADe and CADx in cancer imaging are the current tools for breast cancer detection. The detection of cancer by radiologists is limited by the presence of noise in the structure, incomplete visual search patterns, fatigue, distractions, the evaluation of disease states, and the physical quality of the image. For this reason and with the objective of minimizing the mortality of women due to dinner cancellation, many scientists have applied the use of CNN in the analysis of clinical images considering that CNN has shown good results in other applications.

The projects presented here state that the use of CNN for breast cancer image analysis generates good results and these results are better compared CADe and CADx. In addition, the error rate is lower.

Performing a retrospective, we can see that all the projects in their first step perform a preprocessing of the image in order to improve the contrast of the image by implementing different algorithms, then make a division of the image into patches to have a much more data set large and through CNN to deduce more characteristic in the image. We can also see that the CI in all projects does not exceed the value of 1.

The curation of comprehensive data sets and outcomes that incorporate both disease- related and unrelated elements also will help train and expand AI systems to identify cancer. The greater incorporation of these technologies will reduce costs, improve service quality and support the work of doctors and radiologists.

REFERENCES

[1] World Health Organization. “World Cancer Report 2014”. France, 69372 Lyon Cedex 08, ISBN 978-92-832-0443-5, World Health Organization. 2014.

[2] Spanish association against cancer. 2018. [online] Available. from:https://www.aecc.es/es/todo-sobre-cancer/tipos-cancer/cancer-mama/que-es-cancer-mama.

[3] Global cancer statistics 2018: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries. American Cancer Society. Published: 12 September 2018. [online] Available from: https://onlinelibrary.wiley.com/doi/full/10.3322/caac.21492.

[4] A curated mammography data set for use in computer-aided detection and diagnosis research. Published: 19 December 2017. [online] Available from:https://www.nature.com/articles/sdata2017177.

[5] A. Margolin, E. Bilal, E. Huang, T. Norman, L. Ottestad, B. H. Mecham, ... "Systematic Analysis of Challenge-Driven Improvements in Molecular Prognostic Models for Breast Cancer". Science Translational Medicine 17 Apr 2013: Vol. 5, Issue 181, pp. 181re1. [online] Available from: https://stm.sciencemag.org/content/5/181/181re1.full

[6] T. Araújo, G. Aresta, E. Castro, J. Rouco, P. Aguiar, C. Eloy, A. Polónia, A. Campilho."Classification of breast cancer histology images using Convolutional Neural Networks". June 1, 2017. [online] Available from: HYPERLINK "https://journal.pone.0177544"https://doi.org/10.1371/journal.pone.0177544

[7] Z. Wang, M. Li, H Wang, H. Jiang, Y. Yao. ..."Breast Cancer Detection Using Extreme Learning Machine Based on Feature Fusion With CNN Deep Features". IEEE Access, Vol 7. Electronic ISNN:2169-3536. January, 2019. [online] Available from: https://ieeexplore.ieee.org/document/8613773/authors#authors

[8] T. Kooi, G. Litjens, B. Ginneken, A. Gubern-Mérida, C. Sánchez, R. Mann, A. Heeten b, N. Karssemeije. "Large scale deep learning for computer aided detection of mammographic lesions". Medical Image Analysis. Volume 35, January 2017, Pages 303-312, 2016. Available from: https://www.sciencedirect.com/science/article/abs/pii/S1361841516301244?via%3Dihub

[9] B. Efron. "Bootstrap methods: Another look at the jackknife". Annals Stat. 1979. 7,1–26.

[10] "Convolutional Neural Network" [online] from: http://deeplearning.stanford.edu/tutorial/supervised/ConvolutionalNeuralNetwork/

[11] A. Krizhevsky, I. Sutskever, G. E. Hinton, "ImageNet classification with deep convolutional neural networks", Proc. NIPS, vol. 25, pp. 1106-1114, 2012.

[12] K. Simonyan, A. Zisserman, "Very deep convolutional networks for large-scale image recognition", arXiv:1409.1556, 2014, [online] Available: https://arxiv.org/abs/1409.1556.

[13] K. He, X. Zhang, S. Ren, J. Sun, "Deep residual learning for image recognition", arXiv:1512.03385, 2015, [online] Available: https://arxiv.org/abs/1512.03385.

[14] K. He, X. Zhang, S. Ren, J. Sun, "Identity mappings in deep residual networks", arXiv:1603.05027, 2016, [online] Available: https://arxiv.org/abs/1603.05027.

[15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, "Going deeper with convolutions", arXiv:1409.4842, 2014, [online] Available: https://arxiv.org/abs/1409.4842.

[16] J. Hu, L. Shen, S. Albanie, G. Sun, E. Wu, "Squeeze-and-excitation networks", arXiv:1709.01507, 2017, [online] Available: https://arxiv.org/abs/1709.01507.

[17] P. Monkam, S. Qi, H. Ma, W. Gao, Y. Yao, W. Qian, ... "Detection and Classification of Pulmonary Nodules Using Convolutional Neural Networks: A Survey". IEEE. Pages 78075 - 78091. Electronic ISSN: 2169-3536. 13 June 2019. [online] Available: https://ieeexplore.ieee.org/document/8736217/references#references

[18] H. Ishwaran, U. B. Kogalur, E. H. Blackstone, M. S. Lauer, Random survival forests. Ann. Appl. Stat. 2, 841–860 (2008) [online] Available: https://projecteuclid.org/download/pdfview_1/euclid.aoas/1223908043

[19] J. H. Friedman, Stochastic gradient boosting. Comput. Stat. Data Anal. 38, 367–378 (2002).CrossRefWeb of ScienceGoogle Scholar [online] Avalible: https://www.sciencedirect.com/science/article/pii/S0167947301000652

Обработка данных в геоинформационных системах для выбора местоположения рекламы

Ивлев Владислав Александрович
Институт компьютерных наук и технологий
Санкт-Петербургский политехнический университет Петра Великого
Санкт-Петербург, Российская Федерация
nevidd@yandex.ru

Никифоров Игорь Валерьевич
Институт компьютерных наук и технологий
Санкт-Петербургский политехнический университет Петра Великого
Санкт-Петербург, Российская Федерация
i.nikiforov@ics2.ecd.spbstu.ru

Аннотация **- Выбор местоположения для размещения рекламы играет важную финансовую роль в маркетинге, поэтому этот пункт является одним из основных с точки зрения развития компании для повышения прибыли. В работе описан подход создания веб-сервиса, рекомендующего местоположение на карте для размещения рекламы определенной области маркетинга.**

Отличительной особенностью предлагаемого сервиса является предоставление автоматизации процесса принятия решений о выборе местоположения для размещения рекламы.

Для анализа результатов работы веб-сервиса использованы геоинформационные данные города Санкт-Петербург, а именно станций метро, характеристик многоквартирных домов, магазинов разных категорий и маршрутов и расписания общественного транспорта.

Ключевые слова — большие данные, геоинформационные системы, реклама

I. Введение.

Для своего существования коммерческие организации зачастую имеют дело с клиентами, которые в свою очередь пользуется услугами или товарами компании, принося ей прибыль. Исходя из этого - любая подобная организация заинтересована в постоянном увеличении интенсивности потока обслуживания клиентов, роста количества постоянных клиентов, и тем самым повышая прибыль компании. Для этого требуется:

- зарекомендовать производимый товар (или услугу) на рынке как качественный;
- за счет применения рекламы товара (или услуги) привлечь новых клиентов.

Для решения первого пункта компании требуется привлечение высококвалифицированных специалистов требуемой области, которые будут задавать качество производимого товара. Качество в таком случае можно измерить комплексным методом (1), а именно:

$$Q = \frac{Q_{\text{оц}}}{Q_{\text{баз}}}, \tag{1}$$

где:$Q_{\text{оц}}$ - число обращений о некачественном товаре (или услуге) в сервисный центр;

$Q_{\text{баз}}$ - общее числу проданного товара (или услуги) на рынке;

Ко второму пункту следует добавить, что при использовании рекламы в целях популяризации возникают вопросы об эффективности размещении материала и один из них - выбор местоположения для размещения рекламы. Под эффективностью понимается выполнение поставленных компанией задач на период активности рекламы (например, увеличение количества клиентов компании или реализация товара).

При ручном вычислении, имея большое количество критериев выборки. для лица, принимающего решение [1] (далее ЛПР), возрастает время и сложность проведения математического анализа. Поэтому требуется прибегать к вычислительной мощности компьютера для решения подобного рода задач.

Подход, реализованный в предлагаемом программном продукте, позволит снизить трудоемкость принятия решения (далее ПР) выбора местоположения для размещения рекламы. Веб-сервис при выборе определенной области маркетинга предложит наиболее выгодные для размещения её рекламы.

Под выгодой можно рассматривать, насколько эффективно были выполнены поставленные задачи (касательно размещенной рекламы) на период ее пользования.

Для получения и анализа результатов были использованы данные города Санкт-Петербург.

II. Актуальность темы.

С активным развитием сети Интернет, доступ к которой на данный момент осуществляется практически с любой точки мира, геоинформационные системы [2] (далее ГИС) являются одними из распространенных программных продуктов (далее ПО), которые используются повседневно в разных сферах деятельности пользователя. Ведущей из них является область маркетинга, которая активно внедряется в информационное пространство.

На сегодняшний день ГИС активно решают множество задач организаций в сферах маркетинга:

- помощь при поставке сырья на склады, при экспорте готового товара в отделы продаж;
- заказы товаров и их доставка клиентам;
- рекламная деятельность компаний и её размещение.

При решении подобного рода задач ГИС должна опираться на многокритериальные методы принятия решений, чтобы выбор, сделанный системой, был полезным для поставленных компанией целей и аналитически обоснованным.

Перечислим некоторые из таких методов:

- метод Гурвица (комбинирующий методы максимина и максимакса) [3];
- лексикографический метод [4];
- метод последовательных уступок [5];
- метод множества Парето [6] и другие.

III. Обзор существующих ГИС

При рассмотрении ГИС в первую очередь следует перечислить виды ГИС, которые описаны в таблице 1:

Таблица 1. Виды ГИС и их описание

Вид	Описание	Недостатки	Преимущества
Мобильные ГИС	Используется в основном для доступа и сбора экспериментальных данных.	Малый перечень функционала. Невозможность решения аналитических задач из-за низкой производительности	Доступ к данным в полевых условиях без использования доп. оборудования.
Настольные ГИС	Высокопроизводительное ПО. Позволяет вести детальную работу с данными (анализ, контроль, ПР).	Для работы требуется высокая электронно-вычислительная мощность ЭВМ, из чего вытекают затраты на использование (и поддержку) ГИС.	Широкий спектр решаемых задач по анализу, обработке, контролю, ПР и т.д.
Серверные ГИС	Использование общей серверной ГИС для решения задач анализа, обработки данных.		
Встраиваемые ГИС	Встраиваемое ПО, использующееся для расширения приложения (работа с ГИС-инструментарием через другое приложение)	Невозможность работы без ПО, от которого зависит встраиваемая ГИС	Возможность интеграции инструментария для решения задач. Это повышает производительность за счет отсечения неиспользуемых модулей в ПО.

Рассмотрев виды ГИС, можно сделать вывод, что при решении аналитических задач, предусматривающих обработку данных и поддержку принятия решений (далее ППР) лучше всего подходит серверные или настольные ГИС, работающие на более высокопроизводительных ЭВМ (в сравнении с мобильными ГИС), способные проводить обработку большого объема информации (чем мобильные ГИС) за единицу времени.

ГИС ППР помогает решать такие аналитические задачи. Перечислим некоторые из них:

- Анализ, контроль, ПР по территориальному планированию (размещение объектов, строительство и т.д.), планированию в реальном времени (навигация, выбор объектов по критериям);
- Генерация аналитических карт по финансовой, кадастровой политике;
- Сбор и хранение геоданных (параметры объектов и т.д.).

IV. Анализ предметной области

A. *Data science. Data mining*

Наука о данных (англ. data science [7]) является разделом информатики, решающих задачи по анализу и обработки данных в условиях больших объемов.

На сегодняшний день разделы науки о данных (англ. data science) активно используются в работе (в том числе и в ГИС), т.к. за последнее время значительно увеличился объем данных, над которыми проводиться работа. Для ГИС с каждым годом увеличивается отображаемый объем информации на местности, т.к. городская инфраструктура расширяется, осваиваются новые местности в ранее неизученных местах.

ГИС, работающим с методами принятия решений (навигаторы, системы контекстных реклам, анализа транспортной системы и т.д.) требуется анализ данных, а именно:

- Анализ текущего положения на дорогах с целью построения кратчайшего пути из пункта А в пункт В;
- Анализ интересов пользователя на основе его активности в интернете с целью предложения для него потенциально интересной рекламы.

Для подобного рода задач используются методы раздела Data Science – Data Mining.

Интеллектуальный анализ данных (англ. Data mining [7]) включает в себя методы обнаружения в данных знаний, полезных для принятия решений.

Одной из описательных задач Data Mining является кластерный анализ [8], который присутствует почти в каждой ГИС ППР для решения задач анализа, контроля и ПР.

Одно из главных достоинств кластерного анализа - осуществление разбиения объектов по целому набору признаков. В сравнении с большинством математико-статистических методов, кластерный анализ дает полную свободу на вид рассматриваемых объектов, таким образом можно рассматривать множество данных разной структуры и природы[9].

Результат кластерного анализа - набор кластеров, которые содержат элементы исходного множества [9] (см. рис. 1.).

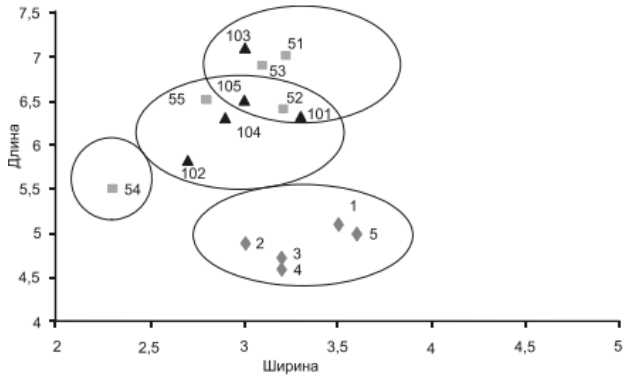


Рис. 1. Пример разбиения множеств на кластеры

Т. о. для кластеризации геоинформационных данных, с целью определения мест с высокой интенсивностью движения потока людей, требуется разбить исходные множества по признакам. Ими могут быть:

- количество участков пересечения транспорта;
- нахождение вблизи станций метро;
- количество населения на заданном участке и т.д.

B. Теория принятия решений для многокритериальных задач

За частую задачи, присутствующие в ГИС, которые требуют принятия решений, носят многокритериальный характер [10]. Это обусловлено тем, что при получении результата, ЛПР опирается на несколько не связанных друг с другом наборов данных. При выборе метода для получения результата (из множества существующих многокритериальных методов, часть из которых перечислена в разделе II) стоит обратить внимание на исходные данные, с которыми будет проводиться анализ. Если требуется наличие критерия ранжирования по важности, то лучше выбрать, например, лексикографический метод [4] или метод последовательных уступок [5]. При отсутствии требования о наличии ранжирования по важности целесообразно применить метод множества Парето [6].

Этот метод используется в задачах, где невозможно объединить несколько частных критериев в один обобщенный, который в дальнейшем должен помочь выбрать из множества объектов один наилучший.

При рассмотрении примера работы метода множества Парето (см. рис. 2.) стоит отметить, что нельзя выбрать одно наилучшее событие (которое будет лучшим по всем критериям. В данном примере критериями являются быстродействие и емкость), потому что характеристики при сравнении могут быть разными и объекты могут быть лучше друг друга в определенных категориях. Например, объект №5 лучше объекта №4 по быстродействию, тем не менее явно уступает по емкости.

Рис. 2. Пример расположения множеств Парето по двум критериям

V. Особенности программной реализации продукта

A. Конциптуальная схема применения ПО

Ниже представлена концептуальная схема работы веб-сервиса на основе модели «Черный ящик» [11] (см. рис. 3):

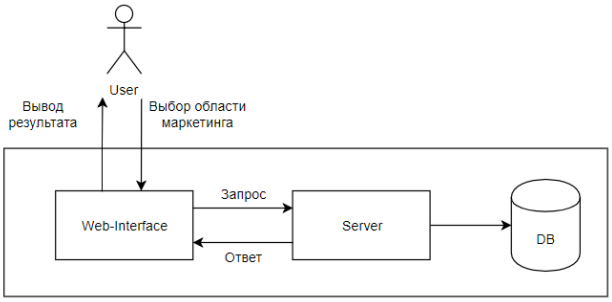


Рис. 3. Концептуальная схема работы веб-сервиса

Рассмотрим подробнее каждый блок концептуальной схемы:

- на вход поступают запросы от пользователя о выборе области маркетинга;
- блок Web-Interface отвечает за формирование запроса на блок Server и отображением для пользователя результатов запроса;
- блок Server обрабатывает запросы пользователя, выполняет аналитические действия, используя хранящиеся в БД данные и формирует ответ для блока web-interface;
- блок Data Base (DB) хранит геоинформационные данные в виде таблиц;
- на выходе пользователь получает результат, а именно отображенные на карте местоположения для размещения рекламы.

B. Укрупненный архитектурный уровень разработки

В основе проекта лежит архитектура Model View Controller [12] (далее MVC) (см. рис. 4.). Особенность

этой архитектуры заключается в том, что разрабатываемое приложение можно разделить на три компонента:

- Model – модель;
- View – представление;
- Controller – контроллер.

Т. о. модификация каждого блока осуществляется независимо.

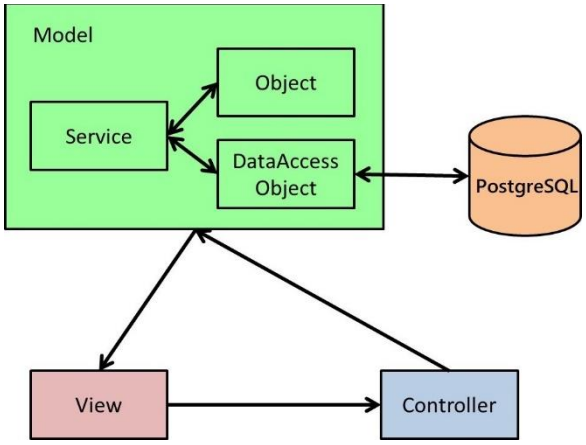


Рис. 4. Диаграмма архитектуры MVC

Подробное рассмотрение каждого слоя архитектурного уровня:

- Model - это представление таблиц базы данных (далее БД) в виде Object-Relational Mapping [13] (далее ORM). Блок Model включает в себя слои Data access object [14] (далее Dao) и Service;
- Dao – слой уровня доступа к БД. В нем формируются Hibernate Query Language [15] (далее HQL) запросы к БД, на основе переданных данных с клиентской части;
- Service – слой бизнес-логики проекта, который включает в себя обращение к Dao и функциям проекта, выполняющим поставленные логические вычисления;
- View – блок, отвечающий за отображение данных и работу с пользователем на клиентской части проекта;
- Controller – блок, обрабатывающий запросы пользователя и передающий их в Model для дальнейшей работы;

C. Проектирование базы данных

БД должна удовлетворять поставленным требованиям:

- контроль целостности данных с использованием связей между таблицами;
- модификация групп и связанных таблиц данных выполнены в рамках транзакций;
- логика работы приложения контролируется триггерами.

Опираясь на поставленные требования, была спроектирована БД, хранящая информацию инфраструктуры города Санкт-Петербург (см. таблицу 2).

ТАБЛИЦА 2.ОПИСАНИЕ ТАБЛИЦ БД ВЕБ-СЕРВИСА

Имя таблицы	Расшифровка
metro	Таблица геоданных станций метро г. Санкт-Петербург.
people	Таблица характеристик многоквартирных домов г. Санкт-Петербург.
shops	Таблица геоданных магазинов разлечных категорий г. Санкт-Петербург.
transport_point	Таблица геоданных маршрутов и расписания общественного транспорта г. Санкт-Петербург.
weight_routes	Таблица кластеров. Каждый кластер – местоположение скопления пересечений маршрутов транспорта г. Санкт-Петербург.

D. Программная реализация

Этап реализации программного продукта разделен на две части:

- разработка серверной части веб-сервиса;
- разработка клиентской части веб-сервиса.

Реализация серверной части состоит в написании программного кода, обеспечивающего функционал работы веб-сервиса и реализации разработанных алгоритмов.

Главная задача серверной части – обработка запросов, поступающих с клиентской части, работа с ними в рамках разработанных алгоритмов и возвращение результатов обратно на клиентскую часть, для их отображения.

Для решения вопроса о маршрутизации запросов, приходящих на серверную часть, был использован DispatcherServlet [16] из выбранного на этапе технической спецификации Framework Spring [16]. Данный менеджер выполняет работу обеспечения функционала, отмечающего за прием и возвращение данных по протоколу HTTP [17].

Реализация клиентской части состоит из написания веб-страницы сервиса для понятного пользователю отображения результатов.

Для отображения карт местности и работы с ней были использованы сервисы API Яндекс.Карт, которые предоставляют доступ для использования технологий Яндекс для других проектов.

Опираясь на анализ ГИС делается вывод, что клиентская часть веб-сервиса должна состоять из главной страницы, пространство которой полностью будет занимать карта местности. Поверх карты будут

отображены инструменты для работы с картой и выбором анализируемой области маркетинга (см. рис. 5.).

Рис. 5. Демонстрация главной страницы разрабатываемого веб-сервиса

Далее продемонстрирован реализованный функционал при работе на клиентской части проекта.

При выборе нужной области из выплывающего списка панели выбора области маркетинга на карте отображаются рекомендованные места для размещения рекламы на основе проведенного анализа кластеризации данных и применения к ним метода множества Парето (см. рис. 6.).

Рис. 6. Демонстрация отображения выбранной категории маркетинга на карте

VI. Результаты работы программного продукта.

При демонстрации результатов работы программного продукта требуется проанализировать полученные данные для дальнейших выводов и предложений по развитию проекта.

Для проведения анализа рассмотрим результат работы сервиса, а именно предлагаемые для размещения рекламы места категории «Спортивные товары» (см. рис. 11).

При анализе стоит обратить внимание на расположение ключевых областей на карте, а именно:

- станций метро (см. рис. 7.);
- исходных данных транспортных передвижений (см. рис. 8) и кластеров, образованных в результате работы алгоритма кластеризации с пересечениями автобусов маршрутов (см. рис. 9.);
- расположения магазинов категории «Спортивные товары» (см. рис. 10.).

Рис. 7. Отображение на тепловой карте станций метрополитена г. Санкт-Петербург

Рис. 8. Исходные данные транспортных передвижений г. Санкт-Петербург на тепловой карте.

Рис 9. Отображение кластеров (результатов метода кластеризации) на карте

Рис. 10. Отображение магазинов категории «Спортивные товары» г. Санкт-Петербург

Стоит отметить, что кластеры, расположенные на карте при анализе, в большинстве случаев, действительно соответствуют реальным участкам скопления транспортных пересечений. Например, некоторые из них:

- перекресток возле станции метро Академическая;
- перекресток возле станции метро Проспект Просвещения;
- участок возле станции метро Купчино.

После выполнения алгоритма метода множества Парето, полученный результат (см. рис. 11.) следует проанализировать.

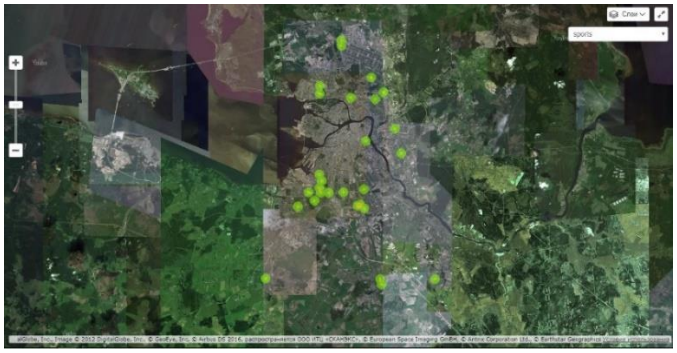


Рис. 11. Результат работы веб-сервиса выбора местоположения для размещения рекламы на примере выбора категории маркетинга «Спортивные товары»

VII. Анализ результатов

Основной целью работы было предоставление готового программного продукта, полностью автоматизирующего ППР при выборе местоположений для размещения рекламы.

Исходные данные для работы системы были взяты с открытых электронных ресурсов:

- открытые данные Санкт-Петербурга [18];
- OpenStreetMap [19];
- портал общественного транспорта Санкт-Петербурга [20].

Ниже перечислены категории и количество данных:

- станции метро – 68 позиций;
- характеристики многоквартирных домов – 22254 позиции;
- магазины различных категорий – 14495 позиций;
- маршруты и расписание движений общественного транспорта - 23675 позиций.

При анализе результатов работы ПП было выявлено, что практически все предложенные места для расположения находятся рядом с кластерами скопления транспортных пересечений и на окраинах города. Основная часть мест расположения магазинов категории «Спортивные товары» (см. рис. 11.) приходится на центральную часть города, т. о. можно говорить о корректной работе алгоритма принятия решений, т. к. метод множества Парето опирается на три условия:

- скопление транспортных пересечений;
- наличие рядом станции метро;
- не более 5 магазинов конкурирующей категории в радиусе 1000 метров.

На данный момент о финансовой выгоде невозможно судить, т. к. требуется получение результатов в ходе эксперимента, поставленного в реальных условиях, который при этом будет учитывать:

- затраты на рекламу;
- сроки ее эксплуатации;
- финансовую прибыль на период пользования рекламной услугой;
- увеличение кол-ва постоянных клиентов и многое другое.

VIII. Выводы и дальнейшее развитие проекта

Проанализировав результаты работы программного продукта можно говорить о адекватности предлагаемых веб-сервисом местах для расположения рекламы выбранной категории маркетинга в городе Санкт-Петербург.

С точки зрения развития проекта акцент стоит поставить на внедрение дополнительных методов при принятии решений. Опираясь на комплексные результаты нескольких алгоритмов можно говорить о более точной аналитике.

Значимость проекта с экономической точки зрения позволяет развиваться бизнесу, что в свою очередь положительно влияет на развитие государства. С научной точки зрения проект показывает пример использования методов разных областей для получения результата, решающего задачи при расчете и анализе выбора места для размещения рекламы.

Рекомендации по развитию проекта включают в себя использование дополнительных методов принятия решений с дальнейшим комплексным анализом результатов этих методов, а также расширение данных для работы с другими городами.

Список литературы

[1] Мушик, Эдвин. Методы принятия технических решений / Э. Мушик, П. Мюллер ; пер. с нем. Н. В. Васильченко, В. А. Душского. Москва : Мир, 1990. 204, [2] с. : ил., табл. ; 23 см. ISBN 5-03-001284-2.

[2] Бугаевский, Лев Моисеевич. Геоинформационные системы : Учебное пособие для вузов / Бугаевский Л., М., Цветков В.Я. Москва : "Златоуст", 2000 - 224с. : ил. 28. ISBN 5-7259-0057-3.

[3] Юдин, Давид Беркович. Вычислительные методы теории принятия решений / Юдин Д. Б. М. : Наука, 1989 г. 319 с. (Теория и методы системного анализа). Библиогр.: с.308-314. ISBN 5-02-014109-7.

[4] Ногин, Владимир Дмитриевич. Принятие решений в многокритериальной среде. Количественный подход /

В.Д. Ногин. Москва : ФИЗМАТЛИТ, 2002. 175 с. : ил. ISBN 5922102745.

[5] Подиновский, Владислав Владимирович. Оптимизация по последовательно применяемым критериям / В.В. Подиновский, В.М. Гаврилов. Москва : Советское радио, 1975. 192 с. : ил.

[6] Черноруцкий, Игорь Георгиевич. Практическая оптимизация и невыпуклые задачи / И. Г. Черноруцкий // Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Сер. : Информатика. Телекоммуникации. Управление = St. Petersburg state polytechnical university journal. Computer science. Telecommunications and control systems : научное издание. Санкт-Петербург. 2013. № 4 (176). С. 79-86. ISSN 1994-2354. ISSN 2304-9766.

[7] Никифоров, Игорь Валерьевич (1988-). Курсовое проектирование по учебной дисциплине "Наука о данных и аналитика больших объемов информации" : учебное пособие / И. В. Никифоров ; Санкт-Петербургский политехнический университет Петра Великого. Санкт-Петербург : Изд-во Политехн. ун-та, 2017. 62 с. : ил. ; 20 см. ISBN 978-5-7422-5638-0.

[8] Гитис, Леонид Хаскелевич. Кластерный анализ в задачах классификации, оптимизации и прогнозирования / Л.Х. Гитис. Москва : МГГУ, 2001. - 104 с. : ил. - (Серия: Практическая статистика для горных инженеров ; вып. 2). Библиогр.: с. 101-102. ISBN 5-7418-0010-6.

[9] Барсегян, Арменак Артаваздович. Анализ данных и процессов : Учебное пособие. 1. СПб : Издательство "БХВ-Петербург", 2009. 512 с. URL: http://znanium.com/go.php?id=350638. ISBN 9785977503686.

[10] Халин, Владимир Георгиевич. Теория принятия решений в 2 т. Том 1 : учебник и практикум для бакалавриата и магистратуры / В. Г. Халин [и др.] ; под ред. В. Г. Халина. М. : Издательство Юрайт, 2018. 250 с. (Серия : Бакалавр и магистр. Академический курс). ISBN 978-5-534-03486-8.

[11] Эшби, У. Р. Введение в кибернетику / У. Р. Эшби : пер. с англ. Москва : Издательство иностранной литературы, 1959. 432 с. : ил.

[12] Кедрин, Виктор Сергеевич. Применение программной архитектуры Model-view-controller для построения системы визуального проектирования и моделирования алгоритмов / В. С. Кедрин, О. В. Кузьмин // Научно-технические ведомости СПбГПУ. СПб. 2011. № 4(128) : Информатика. Телекоммуникации. Управление. С. 36-42 : ил.. ISSN 1994-2354.

[13] Roebuck, Kevin. Object-Relational Mapping (Orm) : High-Impact Strategies - What You Need to Know: Definitions, Adoptions, Impact, Benefits, Maturity, Vendors / K. Roebuck // Tebbo, 2011-634c. ISBN 9781743044759.

[14] Нок, Клифтон. Data Access Patterns: Database Interactions in Object-Oriented Applications / К. Нок // Addison-Wesley, 2003 - 469c. ISBN 9780131401570.

[15] Бауэр, Кристиан. Java Persistence API и Hibernate. 2-е издание / К. Бауэр, Г. Кинг, Г. Грегори : пер. с англ. Д. А. Зинкевич // ДМК Пресс, 2017 – 632с. ISBN 978-5-97060-674-2.

[16] Уоллс, Крейг. Spring in Action / К. Уоллс // ДМК Пресс, 2013 – 752с. ISBN 978-5-94074-568-6.

[17] Столлингс, Вильям. Компьютерные сети, протоколы и технологии Интернета / В. Столлингс ; [пер. с англ. А. Никифорова]. СПб. : БХВ-Петербург, 2005. 817 с. : ил. ; 24 см. ISBN 5-94157-508-4.

[18] Открытые данные Санкт-Петербурга [Электронный ресурс] URL: http://data.gov.spb.ru (дата обращения 6.02.2019).

[19] OpenStreetMap [Электронный ресурс] URL: https://www.openstreetmap.org (дата обращения 6.02.2019).

[20] Портал общественного транспорта Санкт-Петербурга [Электронный ресурс] URL: http://transport.orgp.spb.ru (дата обращения 6.02.2019).

Web-service for processing big data in geographic information systems for selection advertising locations

Vladislav Ivlev, Igor Nikiforov

Annotation - Choosing a location for advertising plays an important financial role in marketing, therefore this item is one of the main from the point of view of the development of the company to increase profits.This paper describes an approach to creating a web service that recommends location on a map to advertise a specific marketing area.

A distinctive feature of the proposed service is to provide automation of the process of making decisions about choosing a location for advertising.

To analyze the results of the web service, the geo-information data of the city of St. Petersburg were used, namely, metro stations, characteristics of apartment buildings, shops of different categories and routes and schedules of public transport.

Keywords - big data, geographic information systems, advertising

Применение процессора AM335x Arm CortexA8 Sitara на платформе BeagleBone Black для обработки аудиосигнала

Vladimir P. Kokarovtsev,
Peter the Great St.Petersburg Polytechnic University

Аннотация – Цель данной работы – обзор одноплатного компьютера BeagleBone Black на базе процессора семейства Cortex AM-335x Sitara с целью рассмотрения возможностей его применения. В нем особенно интересен PRU-ICSS модуль (2 шт.), позволяющий использовать это устройство для решения задач реального времени. Планируется использовать PRU решения задач эхо-компенсации, цифровой фильтрации, изменении частоты дискретизации. обработки сигналов В дальнейшем планируется создание специального синхронного интерфейса для обработки аудиопотока (т.к. возможно только послать цифровой поток по hdmi), используя PRU, поскольку он имеет некоторые отличительные особенности.

I. Введение

Для чего же нужны одноплатные компьютеры? Первоначальная сфера применения одноплатников - образовательно-демонстрационная: при помощи данных устройств можно изучать информатику, основы электроники, схемотехнику и программирование. Благодаря наличию универсальных портов (GPIO, UART) для работы с любыми другими устройствами, одноплатные компьютеры могут применяться в робототехнике, в системах "умный дом" и в любых проектах, требующих программного управления электроникой. Одноплатные компьютеры с установленной операционной системой на базе ядра Linux возможно использовать в построении сетевых устройств, маршрутизаторов (роутеров), различных серверов.

На сегодняшний день одноплатные компьютеры стали достаточно производительными, что ещё больше расширило возможности их применения - большинство современных моделей, имеющих на борту 1-2 Gb и более оперативной памяти, 4-х и даже 8-ми ядерные процессоры, достаточно мощные графические и качественные звуковые подсистемы, могут быть использованы как домашние медиа-центры, в системах видеонаблюдения, а так же как энергоэффективные десктопы, мини-компьютеры под управлением операционных систем на основе ядра Linux - Debian, Ubuntu, Fedora, Android и т.д.

Наиболее доступным и популярным компьютером является BeagleBone Black, оснащенный процессором семейства AM-335x Sitara. BeagleBone Black — это миниатюрный компьютер для электронных проектов, где одновременно нужна и высокая производительность, и широкие возможности для подключения периферии, так же связь с сетью и интернетом посредством операционной системы Linux. Также в нем есть модули реального времени, которые не зависят от основного процессора и способны работать на частоте 200 МГц.

Сама идея обработки аудиосигнала не нова. В случае с платформой BeagleBone Black планируется использовать для этих целей модуль реального времени и определить целесообразность этого метода. Также существуют реализации на платформе FPGA (с использованием устройства Altera 1S10); специальные устройства для обработки сигнала (содержащее обнаружитель разрыва, сконфигурированный для определения появления разрыва исходя из внезапного повышения амплитуды декодированного аудио, полученного путем декодирования первого пакета аудио, который принят корректно после появления потери пакета, и корректор разрыва для корректирования разрыва декодированного аудио); анализаторы аудио (автоматически определяет точки изменения в аудиосигналах, которые нужно обработать; ЦП получает информацию о точке изменения, указывающую положения точек изменения в аудиосигналах, и информация о точке изменения записывается на устройстве хранения данных; ЦП идентифицирует информацию о точке изменения в соответствии с инструкцией, введенной пользователем через ключевой операционный блок).

Ниже показана сравнительная таблица популярных одноплатных компьютеров (Таблица 1). Глядя на нее, можно сделать выводы, что модуль реального времени имеется только на плате BeagleBone Black. Это и является главным критерием выбора именно этой платформы, поскольку дальнейшей целью будет реализация интерфейса с использованием PRU-ICSS для обработки аудиопотока, несмотря на то что некоторые модели обладают лучшими характеристиками (Cores) за меньшую цену.

II. Обзор BeagleBone Black (BBB)

Размер платы — 87×55 мм. Платформа построена на базе процессора Sitara XAM3359AZCZ100 семейства ARM Cortex A8,

который работает на частоте 1 ГГц. Также имеется 512 МБ оперативной памяти DDR3L на шине с частотой 800 МГц и 4 ГБ флеш-памяти eMMC, служащей «жёстким диском».

На плате имеется:

- Слот для microSD-карты, для увеличения встроенной памяти
- micro-HDMI для подключения внешнего монитора или телевизора и воспроизведения звука
- mini-USB для соединения с настольным компьютером и питания
- 2,1 мм гнездо для питания от источника питания на 5 вольт
- RJ45-разъём для подключения к локальной сети
- 2 колодки по 46 пинов для подключения электронных модулей и компонентов
- Из 92 пинов на колодках 65 могут быть использованы для цифрового ввода и вывода общего назначения (GPIO). При этом некоторые из них предоставляют дополнительные возможности:
- 8 каналов ШИМ на 4 независимых таймерах;
- 7 аналоговых входов, подключённых к 12-битному АЦП (4096 градации);
- 4 последовательных интерфейса UART и 1 дополнительный UART только с линией передачи (TX);
- 2 шины TWI/I²C;
- 2 шины SPI;
- 25 портов ввода-вывода реального времени.

Product	Price ($)	Processor	Cores	3D GPU	MCU	RAM	Storage	HDMI or DP-out	OSes
BeagleBone Black	55	TI Sitara AM3358	1x A8 @ 1GHz	PowerVR SGX530	PRU	512MB	4GB eMMC	yes	Linux, Android
BeagleBone Blue	80	TI Sitara AM3358	1x A8 @ 1GHz	PowerVR SGX530	PRU	512MB	4GB eMMC	no	Linux
Raspberry Pi 3 Model B	30	Broadcom BCM43437	4x A53 @ 1.2GHz	VideoCore IV	no	1GB	no	yes	Linux
Tinker Board	50	Rockchip RK3288	4x Cortex-A17 @ 1.8GHz	Mali-T760 GPU	no	2GB	no	yes	Linux
Banana Pi BPI-M64	60	Allwinner A64	4x A53 @ 1.2GHz	Mali-450 MP2	no	2GB	8GB to 64GB eMMC	yes	Linux, Android
Rock64	25 to 45	Rockchip RK3328	4x A53 @ 1.5GHz	Mali-450 MP2	no	1GB to 4GB	empty eMMC socket	yes	Linux, Android
HummingBoard-Base	86 to 216	NXP i.MX6	1x/2x/4x A9 @ up to 1.2GHz	Vivante GC355	no	512MB to 2GB	opt. eMMC or NAND	yes	Linux, Android
Orange Pi One Plus	20	Allwinner H6	4x A53	Mali-T720 MP2	no	1GB	no	no	Linux, Android

Таблица 1 Сравнение одноплатных компьютеров

Порты ввода-вывода реального времени (PRU) подключены к встроенному микроконтроллеру на 200 МГц. Это позволяет управлять ими на низком уровне в реальном времени.

III. Особенности BeagleBone Black

Платформа BBB выделяется на фоне остальных одноплатных компьютеров за счет некоторых особенностей, которые можно назвать ключевыми. Именно они сыграли роль в выборе этой

платформы.

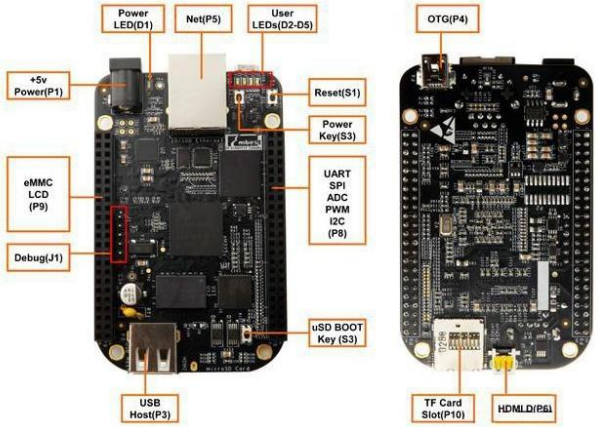


Рис. 1Компоненты BeagleBone Black

Во-первых, это наличие двух дополнительных микроконтроллеров на чипе, и, во-вторых, это open-source проект, поддерживаемый The BeagleBoard.org Foundation. Это означает свободный доступ ко всем исходным схемам, материалам и руководствам, позволяющий модифицировать дизайн платформы под свои нужды и интегрировать его в собственные проекты. Данный микроконтроллер (PRU-ICSS) способен взять на себя часть нагрузки, предназначенной для центрального процессора. В следующих разделах данный модуль реального времени рассматривается более подробно, а также описываются преимущества его использования.

IV. Обзор PRU- ICSS

Интересующий модуль - PRU-ICSS: Programmable Real-Time Unit and Industrial Communication Subsytem – модуль реального времени. Аналогичные PRU решения, из числа популярных, были найдены только для Intel Edison. Но при схожей цене Edison уступает по производительности и характеристикам. Именно наличие PRU делает BeagleBone наиболее предпочтительным для использования в hardware-проектах по сравнению с другими бюджетными одноплатниками типа *Pi (Например, Raspberry Pi 4 – одноплатный портативный компьютер. В нем четырехъядерный 64-битный процессор Cortex-A72 с тактовой 1.5 ГГц в составе SoC Broadcom BCM2711, два USB 2.0 и два USB 3.0, Bluetooth 5.0).

PRU-ICSS состоит из двух 32-битных ядер, имеющих RISC-архитектуру и работающих на частоте 200МГц. Каждое ядро имеет свою область памяти, а также совместную с Linux область памяти, может использовать выводы общего назначения, расположенные на разъемах P8-P9, и формировать прерывания. Этот модуль является важным дополнением всей платформы BeagleBone, позволяющим обеспечивать поддержку для приложений с жесткими временными ограничениями. PRU не является аппаратным ускорителем, позволяющим повысить быстродействие Linux-приложений. Основное преимущество модуля реального времени - короткое время доступа к локальной памяти и периферии.

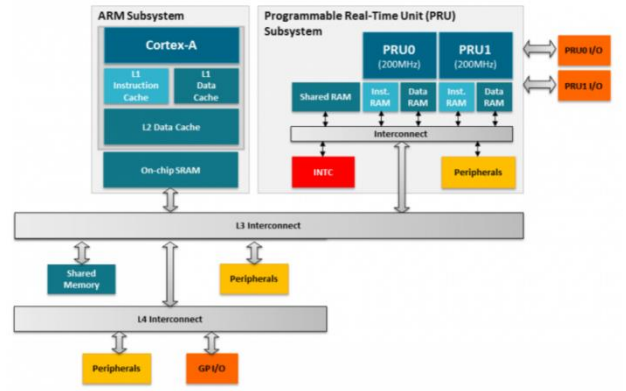


Рис. 2 Programmable Real-Time Unit (PRU) Subsystem

Выбор модуля реального времени обусловлен тем, что он как раз предназначен для разгрузки основного ядра и имеет довольно быстрый доступ к памяти (L3 Interconnect, L4 Interconnect – см. Рис. 2) и периферии, несмотря на довольно невысокую производительность (по сравнению с основным процессором). Другие виды организации цифрового интерфейса выглядят менее практично. Например, один из способов – обработка центральным процессором с помощью набора флажков (устанавливать/сбрасывать, опрашивать состояние и получать сигналы асинхронных прерываний) – годится для низкоскоростной работы. Этот способ дополнительно осложняется тем, что трудно обеспечить требуемое минимальное время реакции в том случае, если события происходят редко, но нуждаются в быстрой обработке. Еще один способ – специализированный контроллер с программным интерфейсом, с помощью которого можно управлять им. Но на данной плате нет контроллера для данной задачи, поэтому этот способ не подойдет. Также существует вариант с FPGA, на которой можно синтезировать нужный контроллер, но он требует дополнительной «надстройки», в то время как PRU уже есть на плате.

V. Архитектура PRU

На Рис.2 представлена архитектура PRU.

Подсистема PRU включает в себя следующие блоки:

- Два ядра PRU, каждое включает в себя:
  - 8KB памяти инструкций;
  - 8KB памяти данных;
  - Высокоскоростной интерфейс шины OCP для доступа к памяти и периферии ARM;
  - Порты ввода/вывода (eGPIO) с поддержкой асинхронного захвата и последовательного вывода;
  - Умножитель с возможностью накопления (MAC);
- Быстродействующая временная память (Scratchpad memory):

    - 3 блока, в каждом 30 32-битных регистров;
    - Прямой доступ обеспечивает возможность быстрой синхронизации между ядрами PRU;
- Один контроллер прерываний (INTC):
    - Прием до 64 внешних событий;
    - 10 каналов прерываний;
    - Аппаратная приоритизация событий;
- Один комплект периферии для промышленного Ethernet:
    - Один таймер с 10 событиями захвата и 8 сравнения;
    - Два сигнала синхронизации;
    - Два 16-битных сторожевых таймера;
    - Цифровые порты ввода/вывода;
- 12KB памяти общего назначения;
- Формирование 16 программных событий;
- Один двухпортовый модуль Ethernet MII;
- Один порт MDIO;
- Один приемопередатчик UART с тактовой частотой 192МГц;
- Один модуль захвата (ECAP);
- Поддержка гибкого управления питанием;

VI. Примеры использования платформы

Платы Beagle идеально подходят для интеграции программного обеспечения высокого уровня и электроники низкого уровня в любой тип проекта, так как они обладают достаточной вычислительной мощностью. Основное преимущество перед более традиционными встраиваемыми системами, такими как микроконтроллеры Arduino, PIC и AVR - использование ОС Linux для проектов.

Данная платформа может использоваться для построения системы Интернета вещей. Также BBB можно использовать в робототехнике (например, робот-змея, предполагаемая область применения которого – жерла, пещеры, лед и скалистые поверхности). Еще этот одноплатный компьютер может применяться в качестве низкозатратного генератора импульсов, точнее для этого используется модуль реального времени (PRU). Для PRU-ICSS доступны протоколы Ethernet, Fieldbusa, а также сертифицированные решения для EtherCAT®, PROFIBUS®, PROFINET®, HSR и PRP и др.

Как было подмечено ранее, основная задача этого модуля (PRU) заключается в принятии на себя части нагрузки, предназначенной для основного ядра, поэтому на PRU можно возложить выполнение следующих отдельных функций и задач. Например, реализация программных высокоскоростных протоколов передачи данных, в том числе и нестандартных, или цифровой обработки сигналов датчиков в режиме реального времени, поскольку главное преимущество модуля реального времени - короткое время доступа к локальной памяти и периферии.

VII. Заключение

В работе была рассмотрен одноплатный компьютер BeagleBone Black (BBB) с процессором AM-335x Sitara и его модуль реального времени (PRU). Также была описана область его применения, и данный модуль был выбран в качестве «фундамента» для создания синхронного интерфейса решения задач реального времени. Дальнейшая цель – создание интерфейса на основе PRU и определение его целесообразности для решения задач обработки аудиосигнала, перечисленных в аннотации.

VIII. Используемые источники

1. Texas Instruments, PRU Read Latencies
2. Texas Instruments, PRU-ICSS / PRU_ICSSG Getting Starting Guide on Linux
3. G. Coley, BeagleBone Black System Reference Manual, 2013
4. D. Molloy, “Exploring BeagleBone: Tools and Techniques for Building with Embedded Linux”, 2nd Edition, 2018
5. Dimitri A. Schreiber, Florian Richter, Andrew Bilan, Peter V. Gavrilov, Hoi Lam Man, Casey H. Price, Kalind C. Carpenter, Michael C. Yip, “ARCSnake: An Archimedes' Screw-Propelled, Reconfigurable Robot Snake for Complex Environments”, 2019
6. Francesco Renna, Joseph Doyle, Vasileios Giotsas, Yiannis Andreopoulos, “Media Query Processing For The Internet-of-Things: Coupling Of Device Energy Consumption And Cloud Infrastructure Billing”, 2016
7. Marcus Fedrizzi, Julio Soria, “Application of a single-board computer as a low cost pulse generator”, 2015
8. Texas Instruments. PRU assembly instructions
9. A. Alanwar, F. M. Anwar, Y. F. Zhang, J. Pearson, J. Hespanha, and M. B. Srivastava, “Cyclops: PRU programming framework for precise timing applications,” 2017
10. Chianese A., Piccialli F., Riccio G. “Designing a Smart Multisensor Framework Based on Beaglebone Black Board”, 2015
11. Antonino Tumeo, Marco Branca, Lorenzo Camerini, Matteo Monchiero, Gianluca Palermo “An Interrupt Controller for FPGA-based Multiprocessors”, 2007
12. Bryan D. Marietta, Gary L. Whisenhunt, Kumar K. Gala, David B. Kramer “Interrupt priority management using partition-based priority blocking processor registers”, 2012
13. Charith Perera, Mahmoud Barhamgi “Augmenting Software Engineering Processes Towards Designing Privacy Aware Internet of Things Applications”, 2019
14. Stanford Research Systems. DG645 digital delay/pulse generator

15. http://linuxgizmos.com/january-2018-catalog-of-hacker-friendly-sbcs/
16. https://patents.s3.yandex.net/RU2651234C2_20180418.pdf
17. https://patentimages.storage.googleapis.com/d5/6f/54/4d4bd296d408ca/US20050182627A1.pdf
18. https://patentimages.storage.googleapis.com/8f/2f/c2/e6930aee115d06/US7490044.pdf
19. https://patentimages.storage.googleapis.com/c3/c0/40/07abab0da6f300/US7260541B2.pdf
20. https://www.sciencedirect.com/science/article/abs/pii/S0141933104000213

АВТОМАТИЗИРОВАННАЯ СИСТЕМА ОБРАБОТКИ ПОЛЕТНЫХ ДАННЫХ БЕСПИЛОТНЫХ ЛЕТАТЕЛЬНЫХ АППАРАТОВ

Васильев В.В.
Санкт-Петербургский политехнический университет Петра Великого
Санкт-Петербург, Россия
v.vasilye@geoscan.aero

Аннотация – в данной статье рассматривается одна из актуальных на сегодняшний день тем – создание инструмента, позволяющего выполнять накопление, систематизацию, статистическую обработку и анализ полётных данных беспилотных летательных аппаратов с целью получения характеристики состояния парка воздушных средств. Разработанный инструмент позволяет с гораздо большей эффективностью обрабатывать данные. Важным в статье является идея полной автоматизации всех процессов по работке с данными воздушных судов в едином решении. Предлагаемое решение апробировано на тестовом наборе полетных данных и был получен положительный результат.

Ключевые слова: беспилотный летательный аппарат, обработка полетных данных, система автоматизации процессов, база данных, наземная станция управления

I. ВВЕДЕНИЕ

Для XXI века характерен бурный рост информационных технологий, который позволяет разрабатывать новый инструментарий для решения актуальных задач в различных сферах и областях деятельности человека. Одной из быстроразвивающихся и востребованных областей на сегодняшний день является беспилотная авиация. Востребованность данной области объясняется тем, что для решения классических задач, таких как мониторинг, картографирование, аэросъёмка применение беспилотных летательных аппаратов (далее БЛА) позволяет затрачивать меньшее количество ресурсов и получать более качественный результат [1,2]. Эффективность применения БЛА обусловлена их малым размером, мобильностью и возможностью применения специализированной полезной нагрузки для решения конкретно поставленной задачи.

С ростом потребности в беспилотной авиации [3], возникает необходимость в своевременной обработке данных полученных с автопилота (далее АП) БЛА. Во время выполнения полетного задания АП БЛА выполняет запись показаний с установленных на БЛА датчиков в режиме реального времени с частотой в сотни ГЦ.

На основе собранных полетных данных (далее ПД) оператору БЛА необходимо определять корректность работы комплекса БЛА. Для этого из необработанных данных требуется получить показатели работы комплекса за текущий полет. Помимо этого, необходимо отслеживать динамику изменения показателей комплекса – т.е. выполнять статистический анализ с учетом уже имеющихся данных за предыдущие полеты. Аналогичная обработка ПД выполняется производителем БЛА для внесения корректировок в алгоритмы работы АП. Обработка данных вручную может доставлять сложности при большом количестве совершенных полетов, поэтому существует востребованность в системах автоматической обработки данных.

Помимо необходимости своевременной обработки и анализа ПД также остро стоит вопрос о визуализации полученных результатов, систематизации и хранении большого количества логов АП БЛА как в обработанном виде, так и в бинарном представлении.

Основной целью статьи является разработка подхода, позволяющего автоматизировать процессы по работе с ПД и объединить их в единое программное решение, которое будет производить сбор, обработку, систематизацию и хранение ПД, выполнять статистическую обработку и предоставлять инструменты для анализа и визуализации полученных результатов.

Отличительной особенностью предлагаемого подхода является полная автоматизация процессов и целостность работы с ПД, т.е. выполняется полный цикл по работе с ПД БЛА.

Применение предлагаемого похода позволяет повысить скорость, качество обработки и эффективность работы с ПД БЛА по отношению к ручному подходу и применению полуавтоматических систем.

В разделе *постановка проблемы* в общем виде рассматривается проблематика исследуемой области и обозревается предлагаемый подход с точки зрения черного ящика.

В разделе *известные методы решения* рассматривается ручной и полуавтоматический подход, а также их реализации в виде программного инструментария и выделяются их недостатки.

В разделе *постановка задачи* исследования формируются требования к разрабатываемой системе обработки ПД БЛА для предложенного подхода.

В разделе *обработка полетных данных* описывается предлагаемый поход. Дается обзор модулей для обработки ПД БЛА и рассматривается их взаимосвязь.

В разделе *архитектура системы обработки ПД АП БЛА* обозревается архитектура реализуемой системы в рамках применения для обработки данных БЛА компании «Геоскан».

В разделе *реализация компонентов системы* описываются применяемые технологии и методы для реализации системы обработки ПД АП БЛА.

В разделе *исследование эффективности* выполняется тестирование реализованной системы на наборе тестовых логов и выполняется сравнение результатов по отношению к ручному подходу обработки данных.

В разделе *заключение* подытоживается результат разработанной системы.

II. ПОСТАНОВКА ПРОБЛЕМЫ

В настоящее время актуальна проблема автоматизации процессов по работе с ПД БЛА. Обработка большого количества логов АП БЛА без использования автоматических систем приводит к увеличению трудозатрат, ухудшению качества результата обработки ПД и соответственно некорректным результатам при анализе ПД. На данный момент востребован подход (система), который позволит полностью автоматизировать процесс обработки и предоставит методы для обработки, анализа, визуализации, систематизации и хранения ПД БЛА в качестве единого решения.

С точки зрения черного ящика такую систему можно описать следующим образом: на вход подаются бинарные логи АП БЛА, а на выходе имеем набор данных характеризующий состояние парка беспилотных воздушных средств (рис.1):

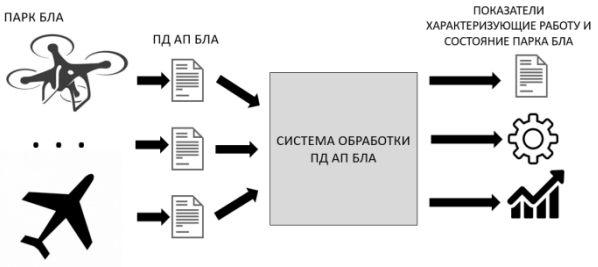


Рис. 5 Вход и выход системы обработки ПД АП БЛА

К данной системе предъявляются следующие требования:

- Система должна работать с бинарными логами АП БЛА;
- Система должна выполнять конвертацию бинарного лога АП БЛА в другие форматы представления;
- Система должна выполнять автоматическую обработку ПД с целью определения показателей комплекса БЛА;
- Система должна систематизировать ПД и хранить обработанные и бинарные представление ПД;
- Система должна предоставлять возможность статистического анализа;
- Система должна предоставлять инструменты анализа полетных данных (визуализация и выделение проблемных областей).

III. ИЗВЕСТНЫЕ МЕТОДЫ РЕШЕНИЯ

В настоящее время в основном практикуется ручной подход обработки и хранения ПД. Ручной подход предполагает вычисления показателей работы комплекса БЛА оператором. Для этого оператор использует набор формул и правил для расшифровки и обработки ПД БЛА. Более подробная информация о том, как расшифровываются и как вычисляются показатели рассмотрены в работах [4,5,6].

Для облегчения процессов обработки ПД БЛА существуют и используются операторами полуавтоматические системы, которые позволяют частично автоматизировать обработку либо собирать краткую статистику по полетам для анализа.

Основные проблемы ручного подхода заключаются в низкой скорости и высокой сложности обработки данных, отсутствии систематизации хранения данных и наличии человеческого фактора: высокая вероятность допущения человеком ошибок при обработке\анализе ввиду большого объема специфичных и однообразных данных.

Для полуавтоматических систем характерна проблема отсутствия полного цикла работы с ПД, что приводит к необходимости обращения к ручным подходам для дальнейшей обработки или накоплению ошибки при переносе результатов между входами и выходами полуавтоматических систем.

Ниже рассмотрены существующие системы, которые используются для работы с ПД БЛА:

Утилита планировщика заданий ArduPilot предоставляет возможность ручной обработки ПД с помощью инструментов просмотра лога АП БЛА. Утилита позволяет конвертировать ПД из бинарного формата файла лога АП БЛА в табличный формат (по каждому параметру) и визуализировать их на графиках [7, 13].

Утилита APM Log Analyser предоставляет возможность полуавтоматической обработки ПД с помощью формирования краткого отчета о полете, однако сформированный отчет не содержит достаточного количества показателей для определения корректности работы АП БЛА [8, 14].

Утилита Airnest предоставляет возможность собирать данные всех полетов, совершенные через конкретно данное мобильное устройство и собирать статистику по данным полетов [9, 15].

В работе [10] рассматривается реализация базы данных, позволяющей производить накопление и систематизацию первичных данных полученных с магнитного самописца

режимов полета для эксплуатации парка воздушных судов Ту-204/214.

Введем критерии, которым должна отвечать автоматизированная система обработки ПД БЛА, и проверим соответствие рассмотренных систем заданным критериям.

1. Наличие модуля интеграции с наземной станцией управления: работа с ПД АП и БЛА должна выполнятся в одной среде;
2. Наличие модуля загрузчика данных ПД АП БЛА непосредственно с БЛА;
3. Наличие модуля конвертации бинарных данных АП БЛА в текстовый и иные форматы;
4. Наличие модуля предобработки ПД АП БЛА: первичная обработка данных и вычисление метрик, характеризующих корректность работы БЛА;
5. Наличие модуля графического представления данных в виде графиков, трека полета и т.д.;
6. Наличие модуля хранения данных полетов парка БЛА;
7. Наличия модуля статистического анализа данных парка БЛА;
8. Наличие общего доступа к обработанным и необработанным данным парка БЛА;
9. Поддержка работы с БЛА различного типа: коптерного и крылатого типа;
10. Выполнение вычислений в облаке.

Ниже представлена таблица, где + отмечена полная реализация требования критерия, - соответственно отсутствие, а +- обозначает частичную реализацию.

Таблица 1

Соответствие решений критериям

№	Ardupilot	APM	Airnest	БД ТУ204/214
1	+	-	+	-
2	+	+	+	-
3	+	-	-	-
4	-	+-	-	-
5	+	+	+	+
6	-	-	+	+
7	-	-	+	+
8	-	-	+	+
9	-	-	-	-
10	-	-	-	+-

На основе информации из таб. 1 видно, что существующие решения делятся на два типа: занимающиеся обработкой и занимающиеся хранением данных. Ни одно из решений не поддерживает работу с парком БЛА состоящего из БЛА различного типа. Предобработка полетных данных выполняется лишь только одним из рассмотренных решений, причем, только частично. Ни одно из решений не соответствует полностью введенным критериям.

Все рассмотренные решения решают проблему, сформулированную в п.2 лишь частично. Ни одно из представленных решений не покрывает полностью требования сформированной к системе, описанной в п.2. и полностью не отвечает сформулированным критериям.

IV. ПОСТАНОВКА ЗАДАЧИ ИССЛЕДОВАНИЯ

В данной работе будет рассматриваться подход, позволяющий автоматизировать процессы по работе с ПД БЛА и объединить их в единое программное решение на примере задачи разработки системы сбора анализа и обработки ПД БЛА компании «Геоскан» и будет выполнено исследование эффективности разработанного решения по отношению к классическому ручному подходу, применяемому в компании «Геоскан».

В рамках данного исследования необходимо рассмотреть и выполнить следующие этапы:

1. Описание общего подход к работе с ПД БЛА;
2. Построение архитектуры системы;
3. Реализация компонентов системы;
4. Исследование эффективности реализованной системы

V. ОБРАБОТКА ПОЛЕТНЫХ ДАННЫХ

Результатом полета БЛА является лог АП БЛА — файл с записями ПД, расположенными в хронологическом порядке, где ПД — показания с датчиков установленных на БЛА и сведения о БЛА. Запись ПД во время работы БЛА осуществляется с частотой в сотни ГЦ. Запись ПД в файл осуществляется в бинарном формате.

Работа с ПД АП БЛА выполняется поэтапно и можно выделить следующие основные процессы:

- Конвертация данных;
- Предобработка данных;
- Хранение данных;
- Анализ данных;
- Визуализация данных

Рассмотрим, как будет выглядеть система обработки ПД АП БЛА и опишем назначение ее модулей (рис. 2).

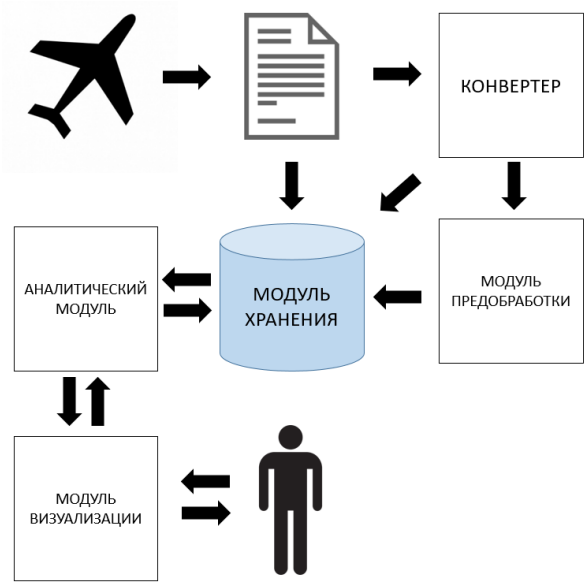


Рис. 2 Система обработки ПД АП БЛА

На первом этапе работы с данными необходимо конвертировать бинарный файл лога АП БЛА в формат пригодный для дальнейшей обработки данных. Данную задачу решает модуль конвертации данных. На вход данному модулю подаются необработанные данные, а на выходе получаем сортированные и структурированные данные в заранее обговоренном формате удобным для обработки. Обычно для данных целей используется табличное представление в файлах формата CSV.

Вторым этапом обработки данных является их предобработка. На данном этапе выделяются или вычисляются показатели работы БЛА, на основе которых можно судить о корректности работы БЛА или использовать для формирования статистики, характеризующей работу БЛА в процессе эксплуатации. Показателем может быть некоторое числовое значение, некоторая запись или таблица с определенными данными.

На третьем этапе работы решается задача хранения и систематизации данных. Для этих целей используется модуль хранения данных, который представляет из себя надежную базу данных.

Для работы с накопленными данными используется аналитический модуль. Данный модуль позволяет производить статистический анализ и другие операции, связанные с анализом данных (вычисление каких-либо показателей и т.д.).

Последним рассматриваемым модулем является модуль визуализации. Данный модуль предоставляет набор инструментов, позволяющий визуализировать результаты обработки в удобной для интерпретации пользователем форме. В основном для вывода информации используется текстовое и графическое представление данных.

Таким образом рассмотренная система предоставляет возможность выполнения работ с ПД АП БЛА полностью в автоматическом режиме, и работа пользователя с данными идет только на уровне загрузки данных и получения качественных результатов, характеризующих состояние парка БЛА.

VI. АРХИТЕКТУРА СИСТЕМЫ ОБРАБОТКИ ПД АП БЛА

В рамках реализации системы обработки ПД АП БЛА было выполнено расширение возможностей наземной станции управления (далее НСУ) БЛА. На рис. 3. представлена архитектура реализованного решения. Система состоит из 2-ух частей, 1-ая, надстройка над НСУ, позволяющая работать с ПД, 2-ая - сервер обработки данных (далее СОД). Разбиение системы на 2 части (клиент-серверную) объясняется по нескольким причинам:

1. Ограниченность ресурсов НСУ. Выполнение обработки данных – трудоемкий процесс в то время, как на НСУ должна вестись работа с БЛА в режиме реального времени.
2. Требуется накапливать данные о совершенных полетах и иметь возможность надстройки сервиса для других реализаций НСУ.

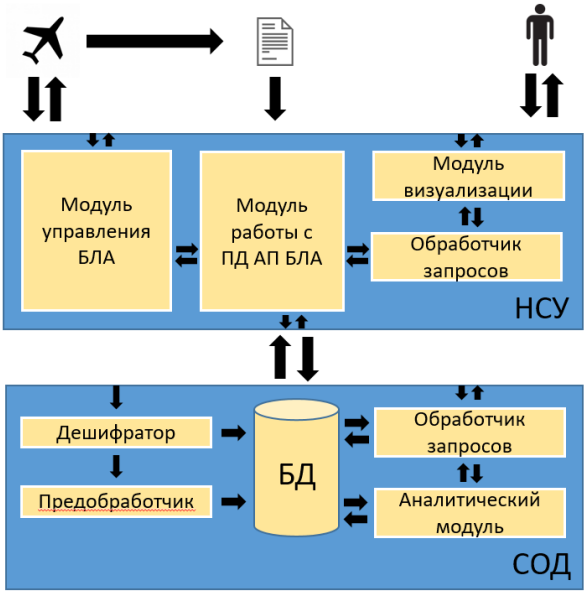


Рис. 3 Клиент-серверная архитектура системы обработки ПД АП БЛА

Реализуемая система состоит из следующих элементов:

Модуль управления БЛА: модуль ответственный за отправку и получения данных от АП БЛА, в нашем случае бинарного лога.

Модуль визуализации: предоставляет пользователю возможность формирования запросов на получения данных по конкретному полету. Данные предоставляются в графическом формате в виде диаграмм, графиков, таблиц с данными.

Обработчик запросов: на стороне НСУ выполняет обработку запроса введенным пользователем и переадресует в модуль работы с ПД АП БЛА. На стороне СОД – получает и обрабатывает запросы от клиентов НСУ.

База данных (БД) – хранилище логов.

Дешифратор и предобработчик выполняют работу по изначальной обработке бинарного лога АП БЛА, после чего сохраняют данные в БД.

Аналитический модуль выполняет преобразование запросов пользователя в команды к БД и соответственно возвращает необходимый результат.

Модуль работы с ПД АП БЛА. Реализует связь между НСУ и СОД. Хранит данные полученные от СОД в рамках текущей сессии работы НСУ.

VII. РЕАЛИЗАЦИЯ КОМПОНЕНТОВ СИСТЕМЫ

Реализация системы была выполнена на высокоуровневом языке программирования Java. Данный выбор обосновывается необходимостью полной интеграции с уже реализованной НСУ. Разработка велась с помощью набора модулей для разработки программного обеспечения RCP предоставленной компанией Eclipce. Разработанная надстройка над НСУ универсальна и выполнена в качестве плагина расширения, которое может переиспользоваться в дальнейшем для интеграции других систем (или НСУ) с СОД.

Для реализации графического модуля был выбран фреймворк визуализации SWT и JFace. Данный набор инструментов предоставляет возможность построения качественного мульти-платформенного пользовательского интерфейса.

Обработчик запросов представляет из себя контроллер, который принимает запросы пользователя и транслирует их в модуль работы с ПД АП БЛА. Данный модуль использует протокол сериализации структурированных данных, предложенный Google (Protobuf) с помощью которого осуществляется передача бинарный логов АП БЛА на СОД. Помимо этого, данный модуль осуществляет связь с СОД посредством GRPC - это высокопроизводительный фреймворк разработанный компанией Google для вызов удаленных процедур. Модуль СОД содержит аналогичный модуль обработчика запросов.

Полученные данные в СОД дешифруются с помощью заранее известных алгоритмов и преобразовываются в табличный формат csv. Расшифрованные данные обрабатываются с помощью специализированного обработчика. Оба данных модуля представлены в виде java-библиотек. Более подробное описание реализации алгоритмов дешифрации и предобработки данных рассматриваются в работе [11]

Полученные данные хранятся в БД. В качестве системы управления данных используется PostgreSQL. Запись и статистическая обработка информации осуществляются с помощью библиотеки обработки данных, подробное описание которой представлено в работе [12]. Работа с данными осуществляется посредством Hibernate — самая популярная реализация спецификации JPA. Java Persistence API — спецификация API Java EE, предоставляет возможность сохранять в удобном виде Java-объекты в базе данных.

VIII. ИССЛЕДОВАНИЕ ЭФФЕКТИВНОСТИ

Исследование эффективности предложенного подхода в рамках реализованной клиент-серверной системы обработки ПД АП БЛА по отношению к ручному подходу будем производить на основе результатов обработки тестового набора данных, выполненной на тестовом стенде. Тестовым набором данных является набор логов АП БЛА, состоящий из 1000 файлов, занимающих объем дискового пространства в размере 40 Гб. Тестовым стендом является компьютер (который одновременно выполняет роль НСУ и СОД) со следующей конфигурацией (таб. 1). Исследование заключается в вычислении времени, требуемого на обработку тестового набора данных и сравнение с затратами для другого подхода.

Исследование производится следующим образом:

1. Фиксируется время начала работы.
2. На вход системы подается лог АП БЛА.
3. Выполняется цикл по обработке ПД АП БЛА;
4. Шаги 2-3 повторяются до тех пор, пока не будет обработан весь тестовый набор данных.
5. Фиксируется время окончания работы.

Таблица 2

Конфигурация тестового стенда

Параметр	Значение
Операционная система	Windows 10
Процессор	Intel® Core(TM) i5-8250U CPU @ 1.60Ghz 1.80 GHz
Оперативная память	12 ГБ
Разрядность системы	64

В результате исследования было выявлено, что на обработку 1000 логов объемом 40 Гб для реализованной системы требуется затратить 300 мин. 45 сек.

Сравним количество времени, затрачиваемое на обработку тестового набора данных вручную и с помощью системы обработки ПД АП БЛА.

Ввиду сложности точного измерения времени, требуемого для ручного способа предобработки полетных данных, данный параметр считается оценочно. На обработку одного лога АП БЛА специалист затрачивает приблизительно 15 минут. Таким образом на обработку 1000 логов требуется ориентировочно 15000 мин.

В итоге имеем следующее: на обработку тестового набора данных с помощью системы обработки ПД АП БЛА было затрачено 300 мин. 45 сек. Отсюда, в среднем, предобработка одного лога АП БЛА с использованием системы обработки занимает 18.045 секунд. В случае ручной обработки ПД, обработка одного лога АП БЛА занимает у специалиста в среднем 15 минут. Таким образом получаем, что автоматическая обработка ПД АП БЛА с помощью предложенного подхода быстрее в ~50 раз по отношению к ручному подходу.

На рис. 4 представлен график зависимости требуемого на обработку времени от количества логов для ручного и автоматического подхода.

В результате исследования можем утверждать следующее:

- Обработка лога АП БЛА с помощью реализованной системы в среднем занимает ~18 сек.;
- Использование реализованной системы повышает производительность процесса обработки ПД на 4987,53% по отношению к ручному способу;

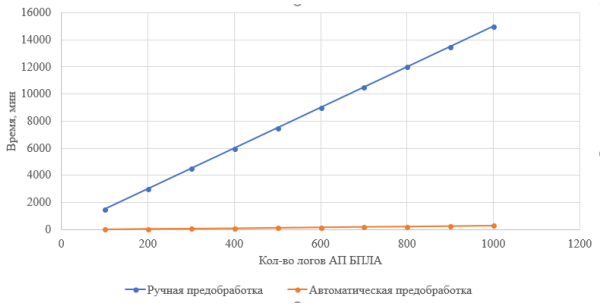


Рис. 4 График зависимости времени обработки от количества логов

IX. ЗАКЛЮЧЕНИЕ

На сегодняшний день является актуальной проблема сложности обработки большого количества ПД АП БЛА. Существует множество подходов, призванных облегчить или сделать возможной работу с ПД, но большинство предложенных решений не предоставляют полной автоматизации процессов по работе с ПД БЛА.

В работе предложен подход к реализации системы, которая позволяет полностью автоматизировать процессы обработки ПД АП БЛА и предоставляет инструментарий для накопления, систематизации, обработки и визуализации ПД БЛА.

Реализованная система была опробован на тестовом наборе ПД АП БЛА, и сравнена с ручным подходом. В результате сравнения было выяснено, что использование предложенного подхода позволяет ускорить обработку данных приблизительно в 50 раз. В дальнейшем планируются доработка и оптимизация разработанного прототипа приложения.

СПИСОК ИСТОЧНИКОВ

1. Федосеева Наталья Алексеевна, Загвоздкин Матвей Викторович Перспективные области применения беспилотных летательных аппаратов // Научный журнал. 2017. №9 (22).
2. Неверова Алина Романовна Использование беспилотных летательных аппаратов в кадастре, землеустройстве и градостроительстве // Интерэкспо Гео-Сибирь. 2017. №2.
3. The market of Unmanned Aerial Vehicles (UAV, drones) in Russia and in the world, 2017
4. M. D. SkubilinO. B. SpiridonovA. V. Pis'menov "A system of flight data acquisition, processing, and recording" June 2008, Volume 51, Issue 2, pp 218–222
5. Омельяненко, О. В., & Лапин, А. Е. (2012). Современные методы и подходы к использованию полетной информации для оценки состояния систем и двигателей воздушного судна. Известия Самарского научного центра Российской академии наук, 14 (4-2), 497-500.
6. Мурин М.В., Клепцов И.Л., и Надтокин Л.А.. "Анализ применения математических моделей при обработке полетных данных" Решетневские чтения, vol. 1, no. 17, 2013, pp. 370-372.
7. José de Sousa BarrosThyago Oliveira FreitasVivek NigamAlisson V. Brito "Analysis of design strategies for unmanned aerial vehicles using co-simulation" Design Automation for Embedded Systems December 2017, Volume 21, Issue 3–4, pp 157–172
8. Wael R. Abdulmajeed, Omar A. Athab and Ihab A. Sattam "implementation and evaluation of apm 2.6 – controlled quadcopter with aerial imagery as a case study" ARPN Journal of Engineering and Applied Sciences VOL. 11, NO. 19, OCTOBER 2016
9. Ebeid, Emad Samuel Malki & Skriver, Martin & Jin, Jie. (2017). A Survey on Open-Source Flight Control Platforms of Unmanned Aerial Vehicle.
10. Смелянский Антон Эдуардович. "Создание базы данных для накопления, систематизации и анализа результатов первичной обработки данных МСРП" Известия Самарского научного центра Российской академии наук, vol. 14, no. 4-2, 2012, pp. 758-761.
11. Васильев В.В. «Предобработка полетных данных беспилотных летательных аппаратов» Санкт-Петербург, 2018
12. Прозоров С.К. « Анализ полётных данных беспилотных летательных аппаратов» Санкт-Петербург, 2018
13. Ardupilot: [Электронный ресурс]. URL: http://Ardupilot.com (Дата обращения 10.10.2019).
14. APM Log Analyser: [Электронный ресурс]. URL: https://www.rcgroups.com/forums/showthread.php?2151318-APM-Log-Analyser (Дата обращения 10.10.2019).
15. Airnest. [Электронный ресурс]. URL: http://www.airnest.com/ (Датаобращения 10.10.2019).

Автоматическая классификация пользователей социальных сетей по общим интересам

Kirill Kobyshev
Peter the Great St. Petersburg Polytechnic University
St. Petersburg, Russia
Kobyshev.ks@edu.spbstu.ru

***Аннотация*. В последние два десятилетия в связи с развитием социальных сетей, тематических интернет-сообществ и медиа-порталов является востребованным автоматизация определения людей с похожими интересами, а также автоматизация выявления интересов конкретного человека. Данная задача является частным случаем рекомендательных систем, которые используются в таких сферах, как: поиск кандидатов на вакантную должность, подбор публикаций научных сообществ, медиа-контента, поиск целевой аудитории на тематическое мероприятие. Существует множество решений, среди которых большинство сложны в реализации и требуют больших вычислительных ресурсов. В данной статье рассматривается пример успешного комбинирования алгоритмов, в котором удалось построить систему рекомендаций, которая проводит анализ интересов человека за оптимальное время и при использовании относительно небольших вычислительных ресурсов. Разработанный алгоритм использует модель Word2Vec, которая была обучена на текстовом корпусе портала «Википедия». Прототип приложения, использующий разработанный алгоритм, был успешно опробован на десятках профилей людей в социальных сетях.**

Ключевые слова: модель Word2Vec, машинное обучение, рекомендательные системы, ETL-системы, анализ интересов пользователя социальной сети.

VIII. ВВЕДЕНИЕ

В последнее десятилетие являются востребованными системы для автоматической агрегации данных о пользователях; востребованы системы, составляющие рекомендации для пользователей исходя из данных о них. Основное преимущество подобных систем заключается в автоматизации анализа пользовательских данных, которое проводится на больших выборках (данные порядка нескольких тысяч и более), которые проблематично обработать вручную. Анализ информации о пользователях из больших выборок данных вручную, без использования автоматизированных программных средств, приводит к увеличению трудозатрат, к ухудшению качества результатов анализа.

Рекомендательные системы используются в таких сферах, как:

- Подбор для пользователя публикаций, материалов научных сообществ, социальных сетей, фотохостингов
- Поиск рекомендуемых на вакантную должность кандидатов
- Поиск целевой аудитории для организации тематического мероприятия

При разработке рекомендательных алгоритмов необходимо прибегать к использованию, а также доработке существующих алгоритмов, разработке новых математических моделей для выявления интересов. Каждый алгоритм может имеет собственные особенности в зависимости от решаемой задачи и не существует универсального подхода для разработки автоматизированной системы рекомендаций.

Проектирование систем рекомендаций ведется с использованием алгоритмов и методов, связанных с такими научными дисциплинами, как:

- теория вероятностей
- математическая статистика
- машинное обучение
- анализ и обработка данных
- классические алгоритмы и структуры данных
- дискретная математика
- теория оптимизации.

IX. ПОСТАНОВКА ПРОБЛЕМЫ

Одной из возможных, но редко используемых моделей работы рекомендательной системы может быть разбиение данных на три типа: пользователи, тематики и материалы. Для каждого из пользователей определяется список тематик, интересных ему. Также для каждого из материалов (публикаций, рекламы, мероприятий) определяется список соответствующих им тематик. В дальнейшем тематики, материалы и пользователи могут быть добавлены в граф в качестве узлов [1]. Над данным графом в дальнейшем может работать рекомендательная система в процессе поиска рекомендуемых изображений и похожих пользователей [2]. Возможны также другие представления данных о пользователях и материалах и их обработка, например, самым распространенным вариантом является матрица предпочтений [3], где не требуется генерировать тематики.

В данном исследовании акцентировано внимание именно на способе, использующем графовое представление. В данном исследовании рассмотрен один из способов преобразования данных о пользователе в тематики (или интересы). Полученные из данных о пользователе тематики в дальнейшем можно включить в

графовую структуру, используемую для системы рекомендаций.

Разработанный алгоритм был применен в веб-приложении, предназначенном для объединения пользователей с общими интересами в группы, который формирует тематики исходя из информации о пользователе и ищет похожих пользователей.

Программное средство, решающее задачу определения интересов пользователя, можно описать методом черного ящика. На вход подается выборка из данных о людях, на выходе системы формируются интересы людей и социальные группы людей с общими интересами.

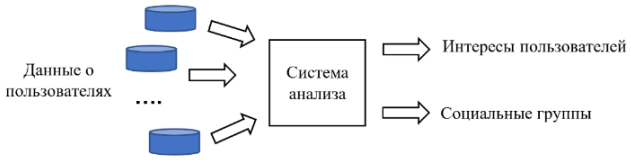


Рис. 3. Вход и выходы системы анализа данных о пользователях

Кроме того, необходимо решить подзадачу, связанную со сбором информации о пользователях. Разработанное программное средство должно содержать модуль, представляющий собой ETL-систему, которая должна в автоматическом режиме собирать и сохранять данные из внешних источников – веб-сайтов.

X. СУЩЕСТВУЮЩИЕ РЕШЕНИЯ ПО АВТОМАТИЗИРОВАННОМУ АНАЛИЗУ ИНТЕРЕСОВ ПОЛЬЗОВАТЕЛЙ

На данный момент существует небольшое число решений, связанных с агрегацией информации о пользователе в виде списка его интересов, отсутствует единый подход. Рассмотрим некоторые существующие решения:

1) Компания Яндекс занимается сбором информации о пользователе через cookies [4]. Затем подает информацию о пользователе в алгоритм MatrixNet вместе с факторами, который задает пользователь алгоритма (помимо этого, пользователь алгоритма настраивает чувствительность алгоритма к факторам на разных диапазонах входных значений алгоритма). Подробное описание алгоритма не приведено в источнике. Тем не менее, данный алгоритм не занимается классификацией пользователей по общим интересам. Область применения данного алгоритма – подбор подходящей для пользователя рекламы.

2) В источнике [5] описано мобильное приложение, которое проводит кластеризацию множества пользователей из нескольких групп в зависимости от их демографических признаков (возраст, профессиональный статус и т.д.). Как и в п. 2. в данном приложении был использован алгоритм иерархической кластеризации.

3) В источнике [6][7] рассмотрен алгоритм, разработанный компанией Facebook. Данный алгоритм делится на 2 этапа, приведенных в Рис. 2: предобработка сообщений и подсчет весов интересов пользователя.

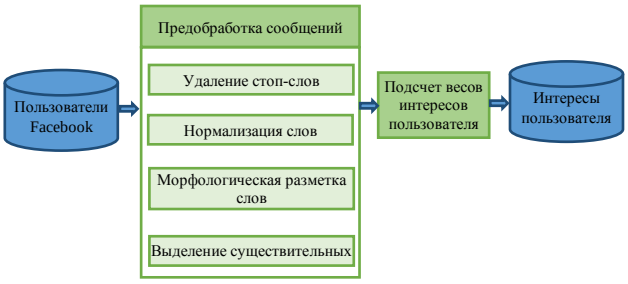


Рис. 2. Алгоритм определения интересов пользователя, разработанный компанией Facebook

Предобработка сообщений включает в себя: удаление стоп-слов, нормализацию слов, морфологическую разметку и выделение существительных.

Под стоп-словами подразумеваются слова, не несущие смысловой нагрузки: артикли, предлоги и т.д.

Нормализация слов – это процесс приведения слов к словарному виду: к единственному числу и начальной форме [8]. Нормализованный текст проще анализировать, так как если исходная выборка содержит слова в разных формах, то алгоритм может сгенерировать одинаковые слова-интересы в разных формах, либо схожие слова в другой форме могут быть не учтены и интерес не будет сгенерирован.

Морфологическая разметка – процесс автоматического определения части речи. После морфологической разметки удаляются слова всех частей речи, кроме существительных.

После получения набора слов в начальной форме начинается этап подсчета весов интересов пользователя по формуле для каждого из слов:

$$W(i) = \log(TF+1) * (\log(Like + 1) +1),$$

TF – отношение числа использований слова к общему количеству слов;

Like – число публикаций с данным словом, которые понравились пользователю;

i – номер слова из полученного набора слов в начальной форме в результате этапа предобработки сообщений;

Данный алгоритм был взят за основу в данной исследовательской работе, он достаточно точен, но не выявляет группы пользователей с интересами, имеющими общий смысловой контекст. В следующих разделах будет рассмотрены улучшения данного алгоритма.

XI. ЗАДАЧИ И МЕТОДЫ ИССЛЕДОВАНИЯ

В данном исследовании будем рассматривать один из возможных подходов к анализу интересов пользователя, каким образом можно подготовить и проанализировать данные пользователя из его профиля в социальной сети.

Сформулируем в рамках какой прикладной задачи необходимо реализовать автоматическое определение интересов пользователя и выделение пользователей в социальные группы. Данная задача возникла при реализации веб-приложения, которое предназначено для объединения пользователей с одинаковыми интересами в тематические онлайн-беседы.

В рамках данного исследования необходимо выполнить следующие этапы:
1) Определение алгоритма, позволяющего из данных о пользователях определить их интересы.
2) Реализовать алгоритм анализа данных о пользователях.
3) Построение архитектуры системы анализа данных пользователей. При этом необходимо рассмотреть существующие функции, библиотеки, позволяющие реализовать части алгоритма.
4) Реализация архитектуры данной системы анализа в виде прототипа приложения (приложения с минимальным функционалом), интеграция со внешними системами, реализация пользовательского интерфейса.
4) Пробное развертывание приложения на общедоступном сервере, получение результатов.

5) Анализ полученных результатов.

6)* В дальнейшем возможна доработка приложения, оптимизация производительности, развертывание, но данные задачи выходят за рамки исследования.

XII. АЛГОРИТМ ОПРЕДЕЛЕНИЯ ИНТЕРЕСОВ ПОЛЬЗОВАТЕЛЯ

Предлагаемое в данной статье решение предполагает следующий алгоритм, схематично представленный на Рис. 3. и являющийся модификацией алгоритма из источника [6][7].

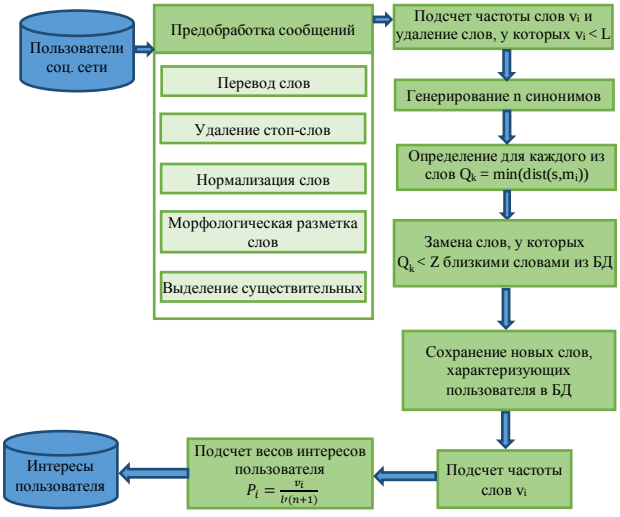


Рис. 3. Алгоритм анализа данных пользователей

Данный алгоритм достаточно прост в реализации, позволяет легко доработать составные части. Этап преобразования возможно ускорить за счет параллельной обработки частей текста, содержащего информацию о пользователе.

На входе алгоритма – данные пользователя, которые предобработке. Слова переводятся на русский язык, если содержат английские слова. Затем удаляются неинформативные стоп-слова (заданы в файле в виде списка из примерно 700 слов). Затем слова подвергаются нормализации, приводятся к начальной форме. Затем остаются только имена существительные, другие части речи удаляются. Впоследствии подсчитывается частота встречаемости слов и удаляются слова, частота которых меньше L (для ускорения алгоритма). Остается l` слов. Величина L подбирается алгоритмом так, чтобы l` не был слишком мал, был не менее порогового значения X.

Затем перебираются все из полученных l` слов. Для каждого из них подбираются n синонимов. Генерация синонимов нужна в тех случаях, когда из похожих слов можно выделить какой-либо интерес пользователя. Например, слова «футбол», «хоккей», «баскетбол» могут образовать общий синоним – «спорт». На данном промежуточном этапе всего имеется l`(n+1) слов, которые характеризуют пользователя.

Теперь для каждого из n синонимов определяем минимальное расстояние до слов, которые уже сохранены в базе данных. Заменяем те слова, которые имеют синоним в базе данных, который находится на расстоянии, меньшем порогового расстояния Z, на данный синоним. Данную операцию необходимо проделать для того, чтобы проделать кластеризацию выявленных из пользовательских профилей слов, определить главные слова в кластерах. Такая методика позволяет ускорить алгоритм и повысить его точность, а также учитывать общий контекст интересов пользователей. Пример, когда алгоритм удаляет слова, которые в некотором абстрактном смысловом пространстве ближе Z приведен на Рис. 4.

Рис. 4. Удаление слов, расстояние от которых до существующих в базе данных слов ближе Z

Если синонимы, ближе порога Z не были найдены в базе данных, слово из профиля пользователя является для рекомендательной системы новым, его необходимо сохранить в базе данных.

Теперь необходимо посчитать частоту встречаемости слов в итоговой выборке, и после – посчитать веса интересов Pi, которые будут варьироваться от 0 до 1 (сумма весов составляет 1 процент). Веса, имеющие наибольшие величину характеризуют главные интересы пользователя.

Теперь имея информацию о весах интересов, можно найти в базе данных пользователей с такими же интересами.

XIII. РЕАЛИЗАЦИЯ СИСТЕМЫ АНАЛИЗА ИНТЕРЕСОВ ПОЛЬЗОВАТЕЛЕЙ

Рассмотрим, как будет выглядит архитектура разрабатываемой системы анализа (см. Рис. 5).

Рис. 5. Основные части системы анализа

Разработанная система имеет клиентскую часть (Desktop и мобильные платформы); frontend-сервер, backend-сервер, сервер базы данных (СУБД MySQL), сервер для анализа пользовательских данных. Frontend и backend сервера взаимодействуют с такими внешними системами, как Yandex Translate API, VK Api, VK Mini Apps. Frontend сервер, Backend сервер, базу данных, анализатор информации о пользователях возможно разместить на разных виртуальных машинах и для каждой выделить соответствующие аппаратные ресурсы, разработанная система поддерживает горизонтальную масштабируемость. Frontend был разработан с использованием языка ReactJS, Backend сервер разрабатывался на языке Java на платформе Spring Boot [11], Анализатор информации разработан на языке Python с использованием фреймворка Flask, клиентская часть реализована на платформе VK Mini Apps. Взаимодействие между частями приложения осуществляется посредством REST-запросов по HTTP протоколу, запросы содержат сообщения в формате JSON.

При реализации алгоритма анализа интересов пользователя большая часть действий была выполнена при помощи стандартных функций, предоставляемых языком. Нормализация слов и поиск синонимов реализованы при помощи сторонних библиотек. Нормализацию слов осуществляет библиотека MorphyAnalyzer [13].

Поиск синонимов к текущему слову осуществляется при помощи модели нейронной сети Word2Vec, зарекомендовавшим себя в последнее время, как надежная модель, не занимающая слишком много места в памяти (в отличие от других классических NLP моделей, которые были разработаны ранее). Данная модель была разработана и запатентована в 2013 году исследователем Томом Миколовым [19][20].

Идея данной модели состоит в векторном представлении слова (слова естественного языка) в n-мерном пространстве. Наглядный пример векторного отображения слов на двумерной проекции n-мерного пространства показан на Рис. 6.

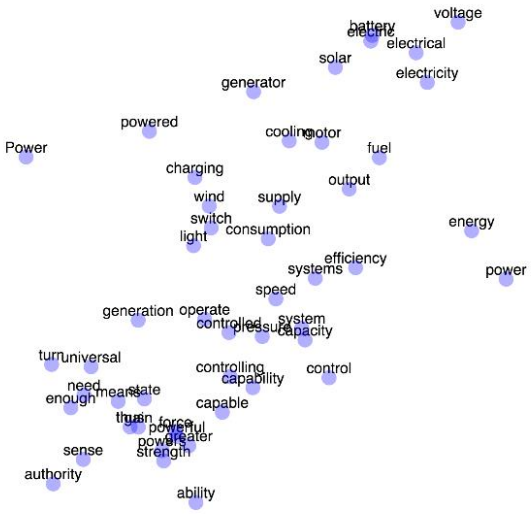


Рис. 6. Векторное представление слов естественного языка в модели Word2Vec

Координаты векторов слов в данном пространстве не поддаются интерпретации и являются абстрактными. Близкие по значению слова располагаются на близком расстоянии в пространстве. При этом над векторами слов возможно проводить операции вычитания и сложения. Между векторами можно измерять расстояние, таким образом определяя, насколько синонимичны друг другу слова. Также данная модель позволяет *определять наиболее близкие в пространстве слова* (синонимы), что и потребовалось для алгоритма определения интересов пользователя.

Модель нейронной сети схематично представлена на Рис. 7.

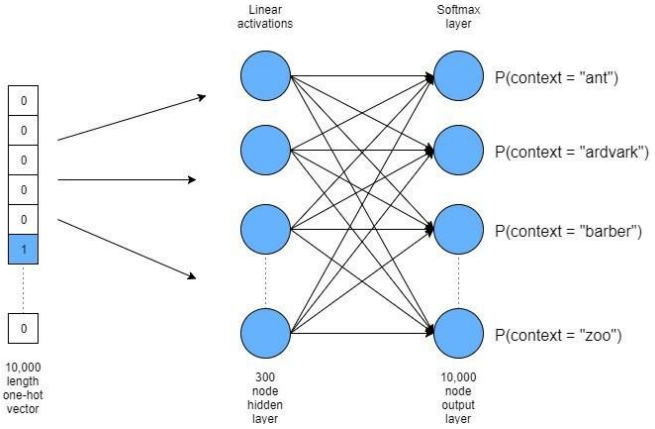


Рис. 7. Модель нейронной сети Word2Vec

У нейронной сети всего один скрытый слой, весовые коэффициенты которого представляют собой координаты слова в пространстве. Слова на входе представляют собой вектора, длина которых равна количеству слов в словаре. К выходным коэффициентам нейросети применяется

логистическая функция Sofmax, по которой формируется вектор вероятностей, суммы которых равны 1:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{k=1}^{K} e^{z_k}}$$

z_i – значение полученного выхода из нейросети под номером i;

z_k – значение полученного выхода из нейросети под номером k;

K – количество коэффициентов на выходном слое нейросети (равно количеству слов в словаре);

Обучение нейросети происходит двумя способами: Skip-Gram и CBOW (Continuous Bag of Words). Нейросеть обучается на словах, взятых из какого-либо текстового источника: новости, литературное произведение, статьи и так далее. Во время обучения происходит чтение текстового источника с помощью перемещающегося по тексту окна (обычно шириной не более 10 слов). Пример чтения текста приведен на Рис. 8.

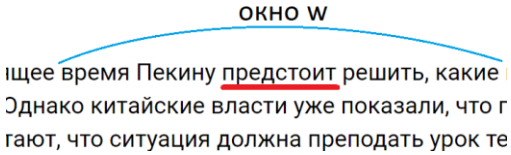


Рис. 8. Чтение текста из новостного источника посредством окна, шириной в 5 слов

Подчеркнутое слово располагается в центре окна. С большой вероятностью слова, располагающиеся в окне, являются синонимами центрального слова – на данном предположении построена модель Word2Vec.

Метод обучения Skip-Gram предполагает подачу на вход нейросети центрального слова. Нейросеть при обучении должна предсказать слова из окна, то есть в результате выполнения Softmax на выходных коэффициентах, соответствующих словам из окна, должна быть максимальная вероятность. Данный способ рекомендован для обучения на небольших исходных текстах. Данный метод схематично отображен на Рис. 9.

В алгоритме определения интересов пользователя, которому посвящено исследование, использовался метод CBOW из-за более высокой точности результатов на большой выборке данных по сравнению с Skip-Gram. При обучении методом CBOW, на вход нейросети подаются слова из окна, нейросеть должна предсказать центральное слово. Данный метод схематично отображен на Рис. 10.

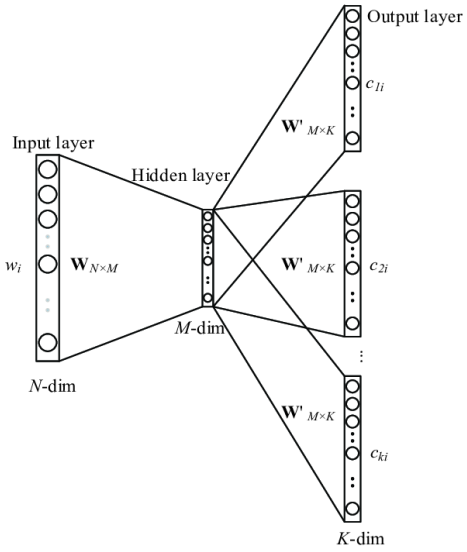


Рис. 9. Метод обучения Skip-Gram

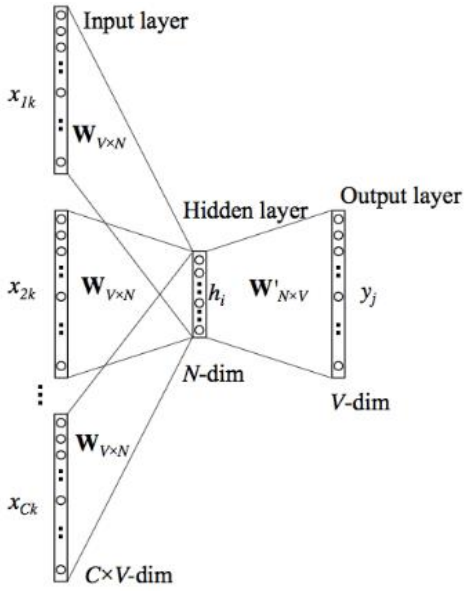


Рис. 10. Метод обучения CBOW

Нейросеть была обучена на текстовом корпусе статей из портала Википедия, размером в 6 Гб.

XIV. ВЫВОДЫ

В отличие от существующих систем анализа данных, данная система дополнительно подбирает синонимы к словам, встречающимся в профиле пользователя. Такой подход оказался полезен и использован на практике. Данное приложение было опробовано на нескольких десятках страниц пользователей в социальных сетях.

Выявлены следующий недостаток: модель Word2Vec не распознает некоторые слова, либо определяет для них неверные синонимы, если слово малоизвестное или если оно не существует. Остается открытым вопрос, как нужно подобные слова фильтровать. Возможно, эту проблему позволят решить запросы к API открытых словарей.

Также необходимо доработать фильтрацию прилагательных, либо нужно найти способы преобразования прилагательных в существительные. Фильтрацию прилагательных возможно сделать с помощью регулярных выражений. Для преобразования прилагательных в существительное необходимо провести анализ существующих решений.

Также возможно, стоит пересмотреть алгоритм анализа пользователей полностью. Например, выявлять интересы не отдельных пользователей, а формировать группы пользователей динамически, из общего числа пользователей, используя алгоритмы кластеризации.

XV. ЗАКЛЮЧЕНИЕ

На сегодняшний день является актуальной проблема определения интересов пользователя социальной сети в связи с развитием социальных сетей, тематических интернет-сообществ и медиа-порталов в последние два десятка лет. Существует множество решений, среди которых большинство сложны в реализации и требуют больших вычислительных ресурсов. Предложен подход к анализу интересов пользователя с использованием нейронной сети, которая определяет синонимы к слову. Реализованный алгоритм был опробован на нескольких десятках людей. В дальнейшем планируется доработка и оптимизация разработанного прототипа приложения.

СПИСОК ИСТОЧНИКОВ

1. Graph-Based Recommendation System, Kaige Yang, Laura Toni, Dept. of Electronic and Electrical Engineering University College London, 2018
2. Graph databases boost customer service with real-time insight [Электронный ресурс], Alaa Mahmoud, 2017 г., URL: https://developer.ibm.com/dwblog/2017/recommendation-engine-customer-insight-graph-database/
3. Building a Book Recommendation System using Matrix Factorization and SV Decomposition [Электронный ресурс], Samia Haimoura, 2019 г, URL: https://towardsdatascience.com/building-a-book-recommendation-system-using-matrix-factorization-and-sv-decomposition-d3541112d53e
4. Яндекс, метод машинного обучения Матрикснет [Электронный ресурс]. Ссылка: https://yandex.ru/company/technologies/matrixnet/
5. Построение графа интересов пользователей на основе мобильного приложения SAZAN, Магистерская диссертация, СПБ ПУ, Г.Ф.Минигалиева, 2017 г.
6. Моделирование интересов пользователей социальных сетей, Магистерская диссертация, МГУ, Александров Н.А., 2015.
7. Jeongin Kim, Dongjin Choi, Byeongkyu Ko, Eunji Lee, and Pankoo Kim. Extracting User Interests on Facebook // International Journal of Distributed Sensor Networks Volume 2014 (2014), Article ID 146967. 2014
8. Speech and Language Processing. Daniel Jurafsky & James H. Martin. August 25, 2019.
9. Natural Language Processing: Python and NLTK, Nitin Hardeniya, Jacob Perkins, Deepti Chopra
10. Описание Translate API, Yandex[Электронный ресурс]. Ссылка: https://tech.yandex.com/translate/
11. Документация Spring Boot [Электронный ресурс]. Ссылка: https://spring.io/projects/spring-boot
12. «Инфраструктура территориально распределенной корпоративной сети», «Демилитаризованная зона» [Электронный ресурс]. Ссылка: https://studbooks.net/2080950/informatika/demilitarizovannaya_zona
13. Руководство пользователя MorphAnalyzer [Электронный ресурс]. https://pymorphy2.readthedocs.io/en/0.2/user/index.html
14. Introduction to word embedding and Word2Vec, Dhruvil Karani, 2013
15. Масштабирование серверов [Электронный ресурс]. Ссылка: https://servakoff.ru/masshtabirovanie-serverov/
16. Моделирование интересов пользователей социальных сетей. Александров Н.А., 2015.
17. Анализ данных пользователей социальных сетей. Антон Коршунов, 2013.
18. BaseGroupLabs, Social Mining – персонификация предложений в социальных сетях [Электронный ресурс]. Ссылка: https://basegroup.ru/community/articles/customization
19. Towards datascience. 2018, Introduction to Word Embedding and Word2Vec [Электронный ресурс]. URL: https://towardsdatascience.com/introduction-to-word-embedding-and-word2vec-652d0c2060fa
20. How exactly does word2vec work, David Meyer, 2016 [Электронный ресурс], URL: http://www.1-4-5.net/~dmm/ml/how_does_word2vec_work.pdf

Разработка модели цифровой копии энергообеспечения мышечной деятельности спортсмена с использованием методов машинного обучения

Е.А. Павлов
Санкт-Петербургский политехнический университет
Петра Великого
Pavlov.e.a.3@gmail.com

***Аннотация*. Исследование феномена анаэробного порога – один из центральных вопросов спортивной медицины и спортивной селекции. Проблема остается актуальной, несмотря на большое количество публикаций по этой теме и несмотря на то, что биохимия обмена углеводов и молочной кислоты в процессе извлечения энергии в живых организмах является хорошо изученной. Понимание способов количественного отображения аэробных и анаэробных источников энергии, факторы их реализации характеризующие системы энергообеспечения, позволяет рассматривать их не только изолированно друг от друга, но и во взаимосвязи между собой. В данной работе рассматривается проблема определения порога анаэробного обмена. Описывается решение этой проблемы с помощью метода машинного обучения Random Forest Regression.**

Ключевые слова: ПАНО, порог анаэробного обмена, машинное обучение, тренировочный процесс, спорт, здоровье, цифровая копия.

1. ВВЕДЕНИЕ

В специальной научно- методической литературе неоднократно показано, что в видах спорта, требующих проявления выносливости, критерий порога анаэробного обмена (ПАНО) является эффективным средством контроля и управления тренировкой, индивидуализации подготовки спортсменов. Повышение ПАНО тесно связано с ростом тренированности спортсмена и многие авторы рекомендуют использовать характеристики ПАНО для развития аэробных возможностей мышц и как показатель, количественно характеризующий уровень специальной выносливости в циклических видах спорта [1,2,3,5,8].

Оценка физической работоспособности, ее моделирование и прогнозирование, изучение особенностей реализации энергетического потенциала организма человека является важнейшим вопросом физиологии экстремальных видов деятельности (спортивной физиологии). Под влиянием экстремальной (тренировочной и соревновательной) деятельности складывается комплекс структурно-функциональных изменений направленных на оптимизацию системы энергетического обеспечения, который следует рассматривать с точки зрения максимальной производительности выражающийся в достаточной мощности, емкости, экономичности с учетом реципрокных отношений внутри этой системы.

Реализация и эффективное использование энергетического потенциала возможно только при адекватной активности аэробных, лактатных и алактатных путей ресинтеза АТФ [12].

Создание программного комплекса (модели) позволит ответить на многие вопросы, связанные с определением уровня подготовленности спортсменов, повышением эффективности использования энергетического потенциала спортсменов, выявления лимитирующих звеньев. В методическом аспекте позволит оптимизировать подходы диагностики, снизить тестирующую нагрузку, повысить информативность текущих тестов и разработать новые сквозные тесты, разработать новые интегральные показатели подготовленности.

2. ОПИСАНИЕ АЛГОРИТМА

Для предсказания порога анаэробного обмена использовался метод машинного обучения Random Forest Regression.

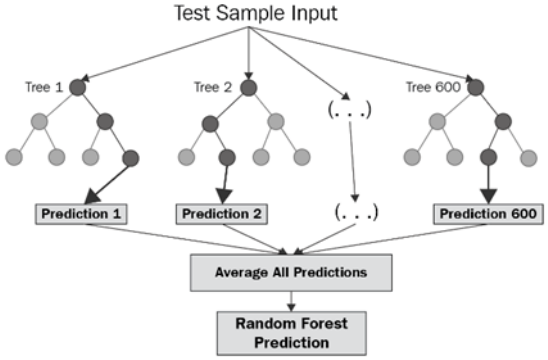


Рисунок 1 Структура алгоритма случайный лес

Random Forest Regression работает по следующему принципу:

Для i= 1,2,...,B(здесь B количество деревьев в ансамбле) выполнить

•Сформировать бутстреп выборку S размера по исходной обучающей выборке D={xi,yi} l i=1;

•По бутстреп выборке S индуцировать неусеченное дерево решений Ti с минимальным количеством наблюдений в терминальных вершинах равным nmin, рекурсивно следуя следующему под алгоритму [13]:

(а) из исходного набора n признаков случайно выбрать p признаков;

(б) из p признаков выбрать признак, который обеспечивает наилучшее расщепление;

(в) расщепить выборку, соответствующую обрабатываемой вершине, на две под выборки

2. В результате выполнения шага 1 получаем ансамбль деревьев решений {Ti} Bi

Формула предсказание новых наблюдений для регрессии :

$$\hat{f}_{rf}^{B}(\mathbf{x}) = \frac{1}{B}\sum_{i=1}^{B} T_i(\mathbf{x}) \ ;$$

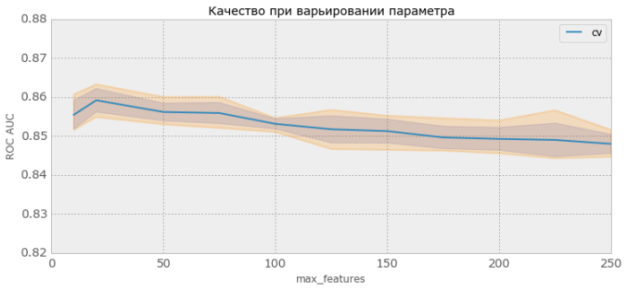
ОПИСАНИЕ МОДЕЛИ

Создание и обучение модели осуществляется с помощью Scikit-learn. Импортируем модель случайного регрессионного леса из skicit-learn, создаем экземпляр модели и тренируем:

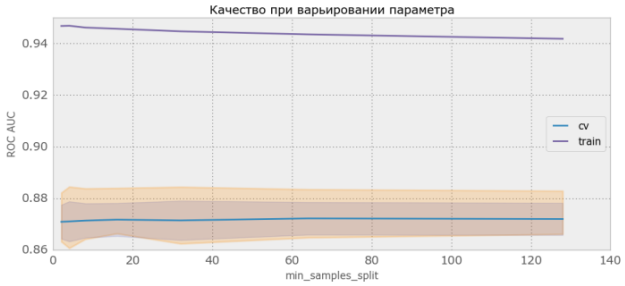
Экспериментальным путем были определены оптимальные настройки для модели.

Наиболее важным параметром является количество деревьев в лесу (n_estimators). Чем больше деревьев, тем лучше качество, но время настройки и работы RF также пропорционально увеличиваются. Выло выбрано ограничение в 1000 деревьев так как с этого момента качество предсказаний на тестовой выборке вышла на асимптоту.

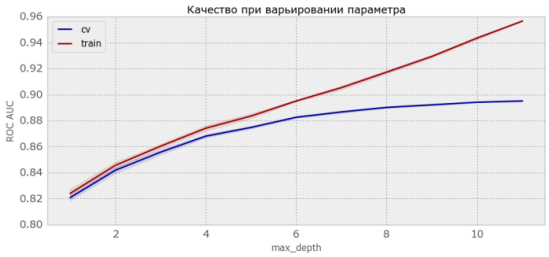
Число признаков для выбора расщепления (max_features) рано n/3



Минимальное число объектов, при котором выполняется расщепление min_samples_split. Этот параметр, как правило, не очень важный и оставляем значение значение по умолчанию (2). При увеличении параметра качество на обучении падает, а время построения RF сокращается.



Максимальная глубина деревьев — max_depth. При увеличении глубины резко возрастает качество на обучении, но и на контроле оно, как правило, увеличивается. Используем максимальную глубину.



Критерий расщепления — criterion. Используется критерий mse.

3. ОПРЕДЕЛЕНИЕ ПАНО

Для построения модели энергообеспечения мышечной деятельности спортсмена использовались анонимизированные данные предоставленные кафедрой лечебной физкультуры и спортивной медицины СЗГМУ им. И.И. Мечникова.

Данные представляют из себя набор показателей энергообеспечения действующих спортсменов во время прохождения кардиореспираторного теста (КРТ). КРТ выполняли на трэдмиле с использованием аппаратуры для эргоспирометрических исследований «Oxycon Pro», Jaeger, Germany [4].

Модель состоит из следующих показателей: динамика потребления O2 (VO2) и выделения CO2 (VCO2) и их соотношение (RER=VCO2/VO2), объем минутной вентиляции (VE), вентиляторный эквивалент по CO2 (VE/VCO2), отражающий объем минутной вентиляции необходимый для выделения 1л CO2 [8,14].

Для тестирования и конечной валидации датасет из 600 человек был разделен соотношением 20/80%, в результате было получено 120 тестовых и 480 тренировочных данных.

В качестве метрики для оценки была использована метрика под названием коэффициент детерминации (R^2 — R-квадрат) — это доля дисперсии зависимой переменной, объясняемая рассматриваемой моделью зависимости, то есть объясняющими переменными. Более точно — это единица минус доля необъяснённой дисперсии (дисперсии случайной ошибки модели, или условной по факторам дисперсии зависимой переменной) в дисперсии зависимой переменной.

Коэффициент детерминации для модели с константой принимает значения от 0 до 1. Чем ближе значение коэффициента к 1, тем сильнее зависимость. При оценке регрессионных моделей это интерпретируется как соответствие модели данным [3].

Для приемлемых моделей предполагается, что коэффициент детерминации должен быть хотя бы не меньше 50 % (в этом случае коэффициент множественной корреляции превышает по модулю 70 %). Модели с коэффициентом детерминации выше 80 % можно признать достаточно хорошими (коэффициент корреляции превышает 90 %). Значение коэффициента детерминации 1 означает функциональную зависимость между переменными [3].

После прогонки модели на обучающей выборки был получен коэффициент детерминации равный 0.97. Можно сказать, что наша модель достаточно хорошо предсказывает порог анаэробного обмена на обучающей выборке. Визуализация предсказаний на обучающие выборке приведена на рисунке 2.

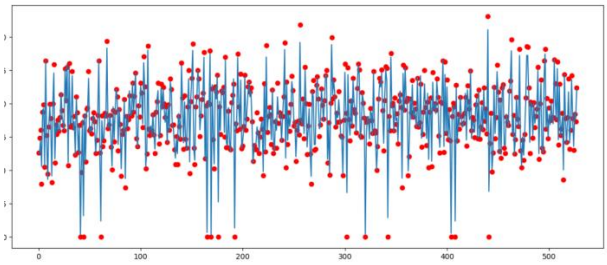


Рисунок 6 Результат предсказания на обучающей выборке

На тестовой выборке мы получили коэффициента детерминации равный 0.93, что является хорошим результатом. Визуализация результатов представлена на рисунке 3.

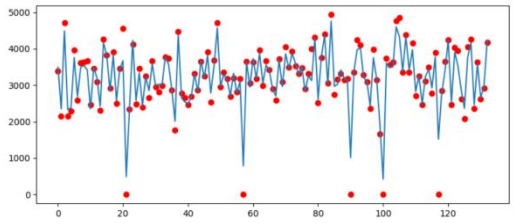


Рисунок 7 Результат предсказания на тестовой выборке

4. ЗАКЛЮЧЕНИЕ

В работе был представлен новый подход определения порога анаэробного обмена с помощью методов машинного обучения. Построенная модель показала хорошие результаты предсказания.

Определение индивидуального анаэробного порога с помощью машинного обучения облегчает и повышает точность определения уровня подготовленности спортсмена. Дальнейшая цель – улучшения качестваа предсказаний ПАНО, поиск взаимосвязей между показателями энергообеспечения, выявление лимитирующих звеньев у конкретных спортсменов и реализация полноценной цифровой копии энергообеспечения спортсмена.

СПИСОК ЛИТЕРАТУРЫ

[1] Baillargeon, Kuhl (2014). "The Living Heart Project: A robust and integrative simulator for human heart function". European Journal of Mechanics A/Solids. 48: 38–47.

[2] McArdle, W.D., Katch, F.I., and Katch, V.L. (2000), Essentials of Exercise Physiology: 2nd Edition, Philadelphia PA, Lippincott, Williams & Wilkins.

[3] Wilmore, J.H. and Costill, D.L. (2005), Physiology of Sport and Exercise: 3rd Edition, Human Kinetics.

[4] American Thoracic Society (2003), American College of Chest Physicians ATS/ACCP Statement on Cardiopulmonary Exercise Testing, Clinical Respiratory diseases and Critical Care Medicine, vol. 167, pp.211-277.

[5] Goh, Eng Lim (July 2018). "How Digital Twins of the Human Body Can Advance Healthcare". Hewlett Packard Enterprise.

[6] Sahli Costabal,..., Kuhl (2019). "Machine learning in drug development: Characterizing the effect of 30 drugs on the QT interval using Gaussian process regression, sensitivity analysis, and uncertainty quantification". Computer Methods in Applied Mechanics and Engineering. 348: 313–333

[7] "Healthcare solution testing for future | Digital Twins in healthcare". Dr. Hempel Digital Health Network. December 2017

[8] Tao, F., Zhang, M. & Nee, A. Y. C. Digital Twin Driven Smart Manufacturing (Academic Press, 2019).

[9] F Tao, Y Cheng, L Xu, L Zhang, BH Li CCIoT-CMfg: cloud computing and internet of things-based cloud manufacturing service system IEEE Transactions on Industrial Informatics, 10 (2) (2014), pp. 1435-1442

[10] F Tao, Y Cheng, J Cheng, et al. Theories and technologies for cyber-physical fusion in digital twin shop-floor Computer Integrated Manufacturing Systems, 23 (8) (2017), pp. 1603-1611

[11] Tao F, Sui F, Liu A, Qi Q, Zhang M, Song B, Guo Z, Lu S, Nee A. Digital twin-driven product design framework. International Journal of Production Research, 2018; Doi: 10.1080/00207543.2018.1443229.

[12] Kühnert, C., Schleipen, M., Okon, M., Henben, R., & Bischoff, T. (2016). A Modular Architecture for Smart Data Analysis using Automation ML, OPC-UA and Data-driven Algorithms. In J., Beyerer, O., Niggemann, C., Kühnert (Eds.), Machine Learning for Cyber

Physical Systems, Selected papers from the International Conference ML4CPS. (pp. 25-33). Berlin: Springer Vieweg.

[13] Lazovskaya, T. N., Tarkhov, D. A., & Vasilyev, A. N. (2017). Parametric Neural Network Modeling in Engineering.Recent Patents on Engineering, Vol. 11, 10–15.

[14] Barber, D. (2012). Bayesian Reasoning and Machine Learning. Cambridge: Cambridge University Press.

[15] Fernando, I.A. (2014). Journey from Big Data to Smart Data. Digital Enterprise Design & Management. In Benghozi, P., Krob, D., Lonjon, A., & Panetto, H. (Eds.) Proceedings of the Second International Conference on Digital Enterprise Design and ManagementDED&M. (pp 25-33).

Формализация процесса оценки стоимости программного обеспечения с использованием аппарата нечеткой логики и алгоритма Мамдани

Serafim S. Tkachenko, Galina Y. Silkina
Peter the Great St.Petersburg Polytechnic University
St. Petersburg, Russia
Seraphim.tkachenko@gmail.com, silkina_gyu@spbstu.ru

***Аннотация* — Объектом исследования являются современные гибкие методологии управления проектами.**

Цель работы заключается в формализации ключевого процесса Agile-методологий – процесса оценки стоимости проекта, итерации проекта – построении соответствующей формальной модели и разработке предложений по оптимизации экономического процесса управления проектами в подразделении крупной компании, связанной с разработкой программного обеспечения.

В ходе работы была построена математическая модель, формализующая процесс оценки стоимости проекта или проектной итерации в Story Points, используя аналитический инструментарий, основанный на теории нечетких множеств, нечеткой логике, алгоритме нечеткого вывода Мамдани.

Полученная модель была протестирована в течение нескольких итераций разработки внутреннего проекта рассматриваемой компании и продемонстрировала положительные результаты относительно применения классических подходов для процессов оценки стоимости в Story Points в Agile-методологиях. Данные результаты позволили повысить экономическую эффективность внутренних проектов компании и повысить эффективность работы проектной команды.

На основе построенной модели были разработаны предложения по оптимизации процесса оценки стоимости итерации разработки программного обеспечения. Разработанную модель можно применять в производственном процессе при оценке стоимости проекта или проектной итерации в Story Points в семействе гибких методологий Agile.

Ключевые слова— AGILE, НЕЧЕТКИЕ МНОЖЕСТВА, НЕЧЕТКАЯ ЛОГИКА, ГИБКИЕ МЕТОДОЛОГИИ, УПРАВЛЕНИЕ ПРОЕКТАМИ, ПРОЦЕСС ОЦЕНКИ СТОИМОСТИ, SCRUM

II. Введение

В настоящее время число проектов, управление которыми осуществляется с использованием гибких методологий, неуклонно растет. Отчасти это связано со стремительным ростом IT-индустрии, которая фактически задала тенденцию развития методологий управления проектами на несколько десятилетий вперед.

Однако гибкие методологии значительно отличаются от классических методологий, которые были повсеместно распространены в сфере управления проектами ранее. В связи с этим возникает ряд проблем, связанных с оценкой стоимости (cost estimation) программного обеспечения или проекта, управляемого с использованием гибких методологий, в целом. Все алгоритмы и методы, которые были применимы в классических подходах, таких как, например, весьма популярная водопадная (каскадная) модель.

Разнообразные модели оценки стоимости выполнения проекта, например, проекта, связанного с разработкой программного обеспечения, были представлены относительно недавно, в течение последних 30 лет. Несмотря на бурное развитие отрасли информационных технологий, методологий управления проектами, направление оценки стоимости все еще находится в зачаточном состоянии, особенно для гибких методологий, которые постоянно меняются и улучшаются. Оценивать стоимость проекта, управляемого с помощью гибких методологий, невозможно используя достаточно распространенные инструменты для последовательных методологий управления проектами, такие как метод функциональных точек (FP) [1] или классические алгоритмы оценки усилий (Effort Estimation).

Необходимо отметить, что само понятие оценки стоимости (Cost Estimation) в данном контексте характеризуется несколькими ключевыми вопросами: сколько усилий нужно применить для выполнения каждой проектной деятельности (project activity), какое количество календарного времени необходимо затратить для выполнения каждой проектной деятельности и какова суммарная стоимость каждой такой деятельности.

Очевидно, что оценка стоимости проекта очень важна, так как определяет ключевые характеристики проекта, такие как его сроки, стоимость в денежном эквиваленте, объем работ и как следствие формирование команды проекта с учетом вышеперечисленных характеристик.

На практике при использовании гибких методологий чаще всего применяются субъективные оценки участников команды проекта для каждой рассматриваемой характеристики, основанные на опыте

участников процесса оценивания. Для простоты и удобства оценки в таких гибких методологиях как Scrum [2] применяют некие абстрактные единицы, такие как Story Point (SP), которые обозначают некоторую абстрактную единицу сложности, трудоемкости и ценности для заказчика. Очень часто на практике при анализе проекта (при использовании гибких методологий) и конкретных требований заказчика они оцениваются сначала в абстрактных единицах трудоемкости, после чего оцениваются примерные временные затраты, после чего уже оценивается стоимость проекта в денежном эквиваленте, опираясь на вышеописанные значения. Так как подобные методы оценивания являются неточными и содержат достаточно большое значение потенциальной ошибки, часто используются специализированные шкалы для оценки трудоемкости в Story Point или человеко-часах. Одним из наиболее распространенных типов подобных шкал является шкала Фибоначчи, где каждое последующее значение формируется из суммы двух предыдущих по типу: 1, 1, 2, 3, 5, 8 и так далее. Использование подобных шкал позволяет учесть потенциальную ошибку и выбор конкретного значения n_i из данного множества характеризует, что фактически значение может располагаться в пределах от предыдущего до последующего значения: $n_{i-1} \leq n \leq n_{i+1}$

Исходя из вышеописанного, можно сделать вывод, что оценка стоимости проекта при использовании гибких методологий является достаточно неточной. По этой причине целью данной работы является попытка разработать рекомендации для одной из крупных мировых IT-компаний, основываясь на попытке формализовать гибкие модели, а точнее их ключевую особенность – оценку стоимости проекта/программного обеспечения с использованием современного математического аппарата.

Актуальность данной проблемы подтверждена практическими запросами компании, в которой используется гибкая методология Scrum, так как после анализа ряда ретроспектив (Retrospective) – одних из ключевых встреч в течение спринта, одной итерации разработки в Scrum, - были обнаружены серьезные расхождения между предполагаемыми оценками командой стоимости итерации разработки и фактическими значениями. Так как одной из ключевых особенностей и преимуществом гибких методологий управления проектами является непрерывная поставка результатов деятельности команды заказчику, неправильная оценка стоимости может привести к нарушению сроков выполнения проекта, выходом за рамки бюджета проекта и другими исключительными ситуациями, связанными с ошибками планирования и оценки стоимости проекта.

Для снижения количества подобных ошибок, необходимо рассмотреть возможность формализации процесса оценки стоимости для проекта или итерации выполнения проекта, проанализировать существующие решения, если таковые имеются и разработать рекомендации для компании, которые помогут снизить ошибки оценки стоимости и повысить эффективность управления проектами с использованием гибких методологий. В качестве практической реализации гибкой методологии необходимо выбрать Scrum и рассмотреть его особенности.

Более формально цели данной работы можно представить в виде следующего перечня:

– кратко рассмотреть организационную структуру компании, рассмотреть структуру ее информационной системы, определить ее особенности;

– проанализировать текущие процессы управления проектами в компании, текущий механизм оценки стоимости проекта или итерации разработки;

– собрать статистические данные об эффективности оценки стоимости проекта, в данном случае программного обеспечения, используя интегрированные системы отслеживания задач (Jira Software [3]).

После сбора и предварительного анализа статистических данных появляется возможность выполнить два шага, позволяющих формализовать рассматриваемый процесс и разработать соответствующие предложения для рассматриваемой компании с целью повышения эффективности управления проектами и, как следствие, получении экономической выгоды для рассматриваемой компании:

– проанализировать существующие аналитические, математические инструменты, которые могут быть применены для формализации оценки стоимости проектов при использовании реализации гибких методологий Scrum;

– на основе собранных статистических данных и с использованием выбранных аналитических инструментов разработать рекомендации по оптимизации процессов оценки стоимости проекта, которые могут быть применены в ближайших итерациях планирования внутренних проектов компании, а в дальнейшем могут быть использованы во внешних проектах компании.

III. Обзор литературы

Формализация процессов Agile-методологий применяется достаточно редко. В подавляющем большинстве случаев решение принимается как некоторое обобщение суммы индивидуальных оценок величин каждого из участников проектной команды.[4] В рассматриваемой компании используется методика Scrum Poker для оценки стоимости задач/проекта в идеальных часах или согласно методологии Story Points Estimation.[5]

Суть данного метода заключается в скрытом оценивании каждой позиции каждым участником команды и одновременном оглашении результатов. Идеальная цель такого подхода – достижение одинакового значения оценки в Story Points или идеальных часах. Если значения разнятся, то происходит обсуждение наиболее отличающихся от большинства оценок, авторы оценок аргументируют свою позицию,

после чего происходит повторная оценка всеми участниками команды.

Процесс оценки стоимости является достаточно сложным и зависит от большого количества разнообразных факторов. Примерами таких факторов могут служить опытность команды, взаимодействие команды между собой. Опытность команды достаточно сложный фактор, так как состоит из многих недетерминированных составляющих – каждый участник команды может быть опытен в каком-то узком кругу задач.

В классических подходах к оценке стоимости в методологии Scrum не учитывается огромное количество разнообразных факторов. Например, вышеупомянутая опытность команды, взаимодействие и отношения между участниками команды не анализируются и обычно описываются с использованием нескольких словесных категорий: низкий уровень, средний уровень, высокий уровень.[6]

Следует рассмотреть, какие категории моделей оценки стоимости программного обеспечения существуют в настоящее время. Глобально данные модели можно поделить на 2 группы, алгоритмические и неалгоритмические. Алгоритмические модели, прежде всего, используют анализ статистических/исторических данных проекта. Данные модели принимают на вход некоторые данные в строго определенном виде и на основе этого выполняют вычисления, позволяющие получить строго определенные результаты, например, предсказания сроков проекта, трудоемкости и так далее в конкретных числовых значениях. В последнее время многие команды отдают предпочтение новым подходам к оценке стоимости, которые используют неалгоритмические модели, использующие такое понятие как мягкие вычисления.[7] Мягкие вычисления – подход к приближенному решению задач, которые не имеют решение за полиномиальное время. Аппарат мягких вычислений построен на использовании нечеткой логики (Fuzzy logic), нечетких множеств (Fuzzy sets), генетических алгоритмов, нейронных сетей и эволюционного моделирования.

В таблице 1 представлены основные методологии оценки стоимости программного обеспечения, разделенные на две категории: алгоритмические и неалгоритмические.

Таблица 1 – Алгоритмические и неалгоритмические модели оценки стоимости программного обеспечения

Алгоритмические модели	Неалгоритмические модели
Story Point Estimation	Метод экспертной оценки
Метод функциональных точек	Закон Паркинсона
Модель издержек разработки (COCOMO, COCOMO II)	Дельфийский метод
SLIM-методика	Оценка до победы
Метод объектных точек	Правило большого пальца
Количество строк кода	Командное планирование программного обеспечения

Метод функциональных точек - Анализ функциональных точек — стандартный метод измерения размера программного продукта с точки зрения пользователей системы. Метод разработан Аланом Альбрехтом (Alan Albrecht) в середине 70-х. Метод был впервые опубликован в 1979 году. В 1986 году была сформирована Международная Ассоциация Пользователей Функциональных Точек (International Function Point User Group — IFPUG), которая опубликовала несколько ревизий метода[8]. Метод предназначен для оценки на основе логической модели объема программного продукта количеством функционала, востребованного заказчиком и поставляемого разработчиком.

COCOMO - COnstructive COst MOdel - (модель издержек разработки) – это алгоритмическая модель оценки стоимости разработки программного обеспечения, разработанная Барри Боэмом (Barry Boehm).[5] Модель использует простую формулу регрессии с параметрами, определенными из данных, собранных по ряду проектов. Базовый уровень рассчитывает трудоемкость и стоимость разработки как функцию от размера программы. Размер выражается в оценочных тысячах строк кода (KLOC - kilo lines of code).

SLIM – Software LIfecycle Model - нелинейная модель расчета трудоемкости программного средства была разработана Лоуренсом Патнамом в 1978 г. на основе эмпирических данных программных разработок Министерства обороны США [7]. Согласно представленной модели, трудоемкость прямо пропорциональна размеру проекта (в LOC-оценке или FPA-оценке) и обратно пропорциональна уровню применяемых технологий, производительности персонала и прочих факторов технологической среды реализации проекта. Расчет затрат на оплату труда базируется на учете трудоемкости разработки программного продукта и тарифной ставке привлекаемых к проекту разработчиков.

Все вышеперечисленные модели не слишком хорошо подходят для решения задач в терминах Agile из-за использования строгих метрик, которых в гибких проектах обычно не бывает.

Из-за смещения акцента в сторону неалгоритмических решений точность оценки слегка снизилась, что обусловлено отсутствием большого количества значимых, но не поддающихся численной интерпретации факторов.[8] Данные неалгоритмические решения получили распространения после массового распространения Agile-технологий в проектах, связанных

с информационными технологиями и разработкой программного обеспечения.

Подробнее следует рассмотреть модель Story Point Estimation, которая используется практически в каждом проекте компании. В данной модели существуют несколько дополнительных факторов, которые участвуют в математических вычислениях: скорость команды (velocity - V) – абстрактная количественная величина, характеризующая максимально возможное количество задач, которые команда может выполнить за одну итерацию, один спринт.

Процесс оценки в Story Points происходит в несколько этапов:

– проставление оценок всеми членами команды для каждой пользовательской истории или ключевой задачи, используется шкала Фибоначчи в одном из нескольких представлений: числовом (1,1,2,3,5,8…144) или «методом футболок» (XS, S, M, L, XL, XXL)[10];

– расчет скорости команды, используя несколько дополнительных факторов – сложность/запутанность задачи и мера оценки скорости команды;

– расчет итогового значения стоимости задачи или пользовательской истории в Story Points.

Достаточно часто на исследуемых проектах вторая и третья фаза значительно сокращались и упрощались, что вело к повышению неточности оценки, снижению эффективности работы команды вследствие снижения общего числа задач, которые были взяты на исполнение на одну итерацию разработки. В связи с этим была сформирована идея о возможном формировании оценки в Story Points, используя более формальные подходы. Идея заключалась в формировании некоторого количества входных параметров и ожидании выходного результата, эквивалентного метрике Story Points.

Из-за наличия однозначной корреляции в методиках оценивания компании между оценкой в Story Points и временной оценкой в perfect hours, данный подход позволил бы более четко определять скорость команды, объем задач, который команда может выполнить в течение спринта.[11]

После проведения исследования доступных аналитических решений, которые могли бы быть применены для формализации процесса оценки стоимости, было принято решение использовать математический аппарат нечетких множеств и нечеткой логики совместно с методологией оценки точек пользовательских историй (Story Points Estimation).

Аппарат нечеткой логики хорошо подходит для формализации процессов оценки стоимости в Agile-методологиях из-за того, что в Scrum и других реализациях огромное количество значимых факторов не имеют детерминированного числового представления, а определяются лишь словами естественного языка или принадлежностью к некоторой категории.[12]

IV. Формализация модели Software Cost Estimation

Нечеткое множество можно представить в виде пары (U, μ), где U – некоторое множество, а μ: $U \to [0,1]$ функция, называемая функцией принадлежности. [9] Иными словами, нечеткое множество можно представить в виде совокупности всех пар,

$$U = \{(x, \mu_U(x)) \,|x \in X\}, \quad (1)$$

где X – множество, называемое универсальным [13].

В нечеткой логике важным является понятие лингвистической переменной. Фактически, лингвистическая переменная представляет собой кортеж $(x, T(x), X, G, M)$, где x – имя переменной, $T(x)$ - множество значений лингвистической переменной x, каждое их которых в свою очередь является нечеткой переменной на множестве X.

G характеризует синтаксическое правило для формирования новых имен новых значений, а M-семантическая процедура, позволяющая преобразовать имя, созданное синтаксическим правилом G в нечеткую переменную, то есть задать вид функции принадлежности.

Существует несколько основных разновидностей функций принадлежности[14]:

– треугольной формы – функция принадлежности, график которой имеет максимум в одной точке, а такую функцию можно задать соотношением

$$f_\Delta(x, a, b, c) = \begin{cases} 0, x \le a \\ \frac{x-a}{b-a}, a \le x \le b \\ \frac{c-x}{c-b}, b \le x \le c \\ 0, c \le x \end{cases}, \quad (2)$$

где a, b, c – произвольные параметры, заданные соотношением $a \le b \le c$;

– трапециевидной формы – функция принадлежности, график которой имеет интервал в качестве максимума и задается системой уравнений (3).

$$f_t(x, a, b, c, d) = \begin{cases} 0, x \le a \\ \frac{x-a}{b-a}, a \le x \le b \\ 1, b \le x \le c \\ \frac{d-x}{d-c}, c \le x \le d \\ 0, d \le x \end{cases}, \quad (3)$$

где также a, b, c, d - произвольные параметры, связанные соотношением $a \le b \le c \le d$.

Рассмотренные типы функций принадлежности называются кусочно-линейными функциями принадлежности, так как состоят из отрезков прямых, составляя непрерывную (кусочно-непрерывную)

функцию. Данные функции используются для определения меры схожести величин, определения среднего значения множества, представления нечетких чисел и интервалов. Данный класс функций принадлежности подходит для решения поставленной задачи формализации.

На рисунке 1 изображен график функции принадлежности треугольной формы, где

$$\forall x \in X \rightarrow x \in [0{,}100] \qquad (4)$$

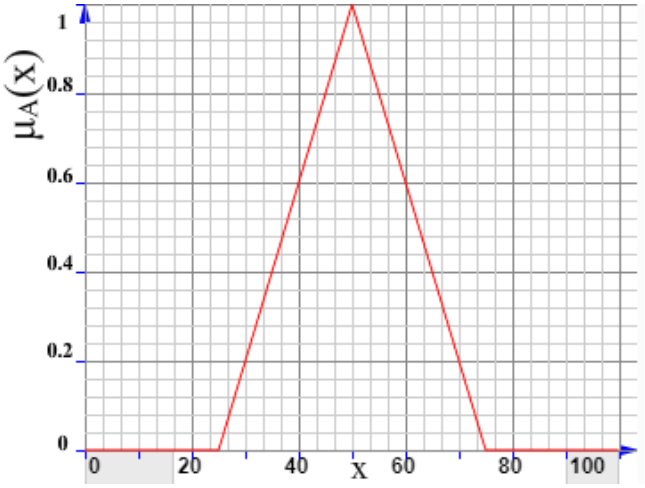


Рисунок 1 – Иллюстрация функции принадлежности треугольной формы

Кроме рассмотренных выше кусочно-линейных функций принадлежности существуют Z-образные и S-образные функции принадлежности, которые были названы так по виду графиков этих функций. Z-образные функции также называются Z-образными сплайн-функциями. Данные функции используются для отражения неопределенности типа меры малости: низкий показатель, небольшая стоимость, скромный доход. S-образные функции, наоборот, подчеркивают меру значимости и большого значения чего-либо, например, такие виды неопределенности как: высокий показатель, значительный темп роста, большое значение.

Помимо этого существуют П-образные функции - произведения двух сигмоидальных функций, а также функция принадлежности, соответствующая функции плотности нормального распределения.

После краткого рассмотрения основных видов функций принадлежности, определяемых в теории нечетких множеств, было принято решение воспользоваться функциями трапециевидной формы. Данный тип функции принадлежности наиболее точно отражает возможное распределение значений параметров при построении модели.

Рисунок 2 – Иллюстрация функции принадлежности трапециевидной формы

Сама система состоит из нескольких модулей:
– фаззификатор;
– система вывода;
– база правил;
– дефаззификатор.

Данные модули будут подробно рассмотрены ниже, а вначале стоит отметить особенности входных параметров.

Входные параметры должны поступать от каждого разработчика проектной команды, участвующего в анализе задачи. Это является достаточно важным условием для корректной работы модели и получения корректных выходных результатов.

Первым параметром является опыт участника команды – количество опыта, которое имеет каждый разработчик команды. Данный параметр имеет три значения, терма лингвистической переменной «опыт участника команды»: «начинающий», «средний», «опытный».

Вторым параметром был выбран размер конкретной задачи. Данный параметр имеет одно из следующих значений-термов: «маленький», «средний», «большой», «огромный». Можно заметить, что такое разбиение соответствует общепринятому в Scrum подходу оценки значений размерами футболок.

Третьим параметром является сложность задачи. Это очень сложное понятие, влияние на которое оказывают огромное количество факторов: формат задачи, ясность понимания будущих действий командой и многое другое. Данный параметр может принимать значения-термы: «очень простой», «простой», «средний», «сложный», «очень сложный».

Четвертый входной параметр это точность оценки на предыдущем этапе. Данный параметр необходим для совершенствования модели в новых итерациях. Может принимать основные значения-термы: «недооценка», «точная оценка», «переоценка».

На рисунке 3 изображена структурная схема построенной модели, на которой отображены основные компоненты модели. Необходимо рассмотреть каждый из данных компонентов подробнее.

Важно отметить, что каждый из входных параметров имеет своё конкретное числовое значение, соответствующее различным экспертным шкалам измерений. То есть, например, размер задачи в проекте в некоторых проектных командах может измеряться в шкале от 1 до 20. Для удобства и наглядности работы модели следует нормализовать значения входных параметров, привести их к одной общей и наглядной шкале измерения. В рамках этапов разработки и построения модели была выбрана шкала [0,100] для нормализации векторов входных параметров.

Фаззификатор предназначен для трансформации входных скалярных параметров (образующих N-мерный вектор параметров) полученных от каждого разработчика в некоторое нечеткое множество с трапециевидной функцией принадлежности. Фактически, процедура фаззификации генерирует новое множество, образованное значениями функции принадлежности от значений входных параметров. Соответственно, вектор входных параметров образует множество нечетких множеств, соответствующих определенным лингвистическим переменным. Фактически, фаззификатор формирует нечеткие множества, основываясь на предварительно сформированной базе правил. В рассматриваемом случае в роли предварительной базы правил для фаззификации используется набор входных параметров для построения функции принадлежности лингвистической переменной. Данный набор параметров определяется методом экспертной оценки для первой итерации работы программы, оцениваются границы значений лингвистических переменных с помощью наиболее опытных участников команды разработки, Scrum-мастера команды.

Рисунок 3 – Схема построенной модели, основанной на нечеткой логике

Изначальные значения границ категорий входных параметров можно представить в виде набора таблиц, соответствующих каждому из входных параметров. То есть необходимо сопоставить значения лингвистических переменных с некоторыми конкретными значениями. Для удобства расчета значений и наглядности представления параметров можно использовать шкалу кратную числу 100 для отображения значений параметров на графике трапециевидной функции по оси абсцисс.

Следует подчеркнуть, что начальное значение лингвистической переменной «точность оценки» должно соответствовать значению нечеткой переменной «точная оценка» для корректной работы модели. Ошибка оценки на данном этапе отсутствует и равняется 0.

В таблице 2 представлены значения параметра «Опыт участника команды» и соответствующие параметры для построения трапециевидной функции принадлежности.

Таблица 2 – Нечеткие переменные параметра «Опыт участника команды»

Нечеткая переменная	Параметры функции принадлежности
Начинающий	0, 14, 25, 35
Средний	26, 39, 54, 65
Опытный	58, 69, 82, 97

Подобные параметры позволяют получить общий график трапециевидной функции принадлежности, на котором будут отображены все лингвистические переменные. Данный график, построенный с использованием инструмента MATLAB, проиллюстрирован ниже на рисунке 4.

Таблицы для всех остальных входных параметров, а также соответствующие им графики трапециевидной функции принадлежности можно посмотреть в Приложении А.

Продолжая рассматривать структуру модели, необходимо перейти к следующему элементу – системе вывода. На самом деле, данный компонент представляет собой всего лишь конечное множество условий типа «ЕСЛИ-ТО». Система нечеткого вывода является преобразователем входного нечеткого множества в выходное нечеткое множество [14]. При преобразовании при этом используются методы нечеткой логики. Выходным нечетким множеством при этом является нечеткое множество оценок в Story Points в соответствии с принятой в компании шкалой Фибоначчи, в которой значения могут находиться в интервале от 1 до 144, на практике оценки имеют фактические значения от 1 до 89.

Аналогично предыдущим преобразованиям, нечеткое множество выходного результата – оценки в Story Points – формируется за счет построения еще одной трапециевидной функции принадлежности.

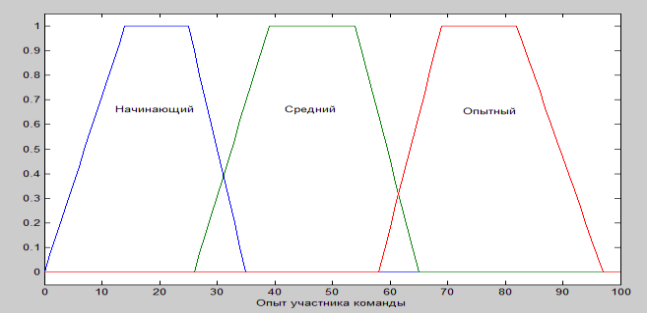


Рисунок 4 – График трапециевидной функции принадлежности для входного параметра «Опыт участника команды»

При повторении итераций происходит добавление поправок в базу правил, которая влияет на качество выходных результатов модели. Данная база правил содержит набор утверждений, который сопоставляет лингвистические переменные входных параметров лингвистическим переменным, фактически, категориям, оценок в Story Points.

База правил формируется из утверждений типа IF (CONDITION1 AND CONDITION2) THEN (CONCLUSION)(F), где F – некоторый весовой коэффициент, обозначающий степень истинности получаемого заключения. По умолчанию данный весовой коэффициент принимается равным 1. В составлении правил для данной базы используются обычные логические операции, такие как конъюнкция, импликация и их всевозможные комбинации. Данный принцип построения и работы системы нечеткого вывода называется алгоритмом нечеткого вывода Мамдани [15]. Принцип данного вывода базируется на нескольких основных этапах:

– формирование базы правил;
– фаззификация;
– агрегирование подусловий;
– активизация подзаключений;
– аккумулирование заключений;
– дефаззификация.

Рассмотрим основные ключевые этапы. Агрегирование подусловий применяется при наличии конъюнкции подусловий. Для каждого рассматриваемого условия необходимо найти минимальное значение истинности всех его подусловий:

$$k_j = \min\{f_i\}, \tag{5}$$

где f_j – значение функции принадлежности, рассчитанное после фаззификации, j – номер правила от 1 до n, i – номер подусловия, соответствующего правилу j.

Интересно отметить также механизм активизации подзаключений в построенной модели. Данная фаза характеризует переход от подусловий и условий к подзаключениям. Степень истинности каждого подзаключения может быть определена как произведение:

$$m_i == F_i * k_i, \tag{6}$$

где k_i – коэффициент истинности, полученный на этапе агрегирования подусловий, а F_i - весовой коэффициент истинности получаемого заключения, который по умолчанию в данной модели равен 0 в исходной базе правил.

Активизация подзаключений состоит из построения нового множества L_i с новой функцией принадлежности $\mu_i^*(x)$, значение которой можно вычислить как

$$\mu_i^*(x) = \min(m_i, \mu_i(x)), \tag{7}$$

где $\mu_i(x)$ – «неактивизированная» функция принадлежности терма.

Таким образом, полученное нечеткое множество L_i для каждого из подзаключений. В построенной модели и разработанной первоначальной базе данных пока отсутствуют механизмы подзаключений, и все сводится к прямому формированию заключений из подусловий. При наличии подзаключений применяется дополнительный шаг - аккумулирование подзаключений, которое состоит в формировании нечеткого множества объединения двух нечетких множеств, функция принадлежности которого задается как максимум значений функций принадлежности подзаключений для i-ой выходной переменной.

Последним этапом является дефаззификация – получение числового значения Story Points из нечеткого множества, сформированного на этапе нечеткого вывода. Было исследовано и применено несколько методов дефаззификации, первый из которых называется методом центра тяжести, где результат дефаззификации можно рассчитать по формуле:

$$y_i = \frac{\int_{min}^{max} x*\mu_i(x)dx}{\int_{min}^{max} \mu_i(x)dx}, \tag{8}$$

где y_i – выходной результат дефаззификации, $\mu_i(x)dx$ - функция принадлежности соответствующего нечеткого множества, а min и max – границы универсального множества (универсума) нечетких переменных. Для дискретного случая применим следующий вид данной зависимости:

$$y_i = \frac{\sum_{i=1}^{n} x*\mu_i(x)dx}{\sum_{i=1}^{n} \mu_i(x)dx}, \tag{9}$$

где n – количество правил вывода, соответственно результирующих нечетких множеств.

Стоит отметить, что сама база правил пополняется, в основном анализируя результаты прошлых итераций моделирования. При наличии особой необходимости также могут изменяться параметры функций принадлежности для входных параметров модели, однако данные действия могут привести к снижению эффективности утверждений в базе правил и необходимости ее пересмотра.

Каждый участник команды генерирует свой набор входных данных для каждой задачи в спринте, выбирая одно из значений параметров, после чего данное значение нормализуется и подается на вход системы. Модель работает по принципу MISO – Many Input Single Output, что позволяет получить количественную оценку в Story Points для каждой задачи.

Для расчета суммарной стоимости (трудоемкости) проекта в Story Points используется расчет суммы оценок полученных задач от каждого разработчика, после чего происходит вычисление арифметического среднего значения, которое и является оценкой стоимости спринта, итерации разработки проекта.

Наконец, необходимо рассмотреть анализ погрешности, неточности предсказываемого моделью результата. Для этого были использованы две общепринятые метрики: значение относительной погрешности (ОП) и оценка истинности предсказания (ОИП).

Значение относительной погрешности можно вычислить по формуле:

$$\delta_i = \frac{|A_i - P_i|}{A_i}, \quad (10)$$

Где P_i – предсказанное моделью значение Story Points для текущего спринта/задачи. Исходя из определения значения относительной погрешности, можно вывести еще одну важную метрику, которая применяется при неоднократных наблюдениях – среднюю относительную погрешность. Данную величину можно несложно рассчитать, используя формулу 3.11:

$$\bar{\delta} = \frac{1}{N}\sum_1^N \delta_i, \quad (11)$$

где N – общее число оценок, что эквивалентно количеству задач для одного спринта.

Показатель истинности предсказания можно оценить, используя соотношение:

$$\gamma(x) = \frac{L}{N} * 100, \quad (12)$$

где L можно описать как количество измерений/предсказаний, в которых относительная погрешность была меньше заданного числа x. Оптимальным аргументом для данной функции на каждой итерации может служить показатель средней относительной погрешности исследуемой итерации разработки.

Стоит отметить, что входным параметром ошибки для каждого разработчика является средняя относительная погрешность оценки разработчиком заданий на прошлой итерации расчета модели. Чем ниже классифицируется точность оценки конкретного разработчика, тем ниже весовой коэффициент его оценки в совокупной оценке стоимости следующей итерации проектной разработки.

V. ЗАКЛЮЧЕНИЕ

После формального описания и построения модели, следует перейти к описанию тестирования разработанной модели на экспериментальных данных, то есть описать применение построенной модели для нескольких спринтов крупного внутреннего проекта компании и оценить ее качество.

Результат анализа показал, что средняя погрешность оценки в процентах за 20 спринтов 3 крупных внутренних проектов компании составляет 47%. Точность оценки в данном контексте характеризует разброс значений от фактически полученного значения Story Points. То есть, +-47% от фактического значения стоимости итерации разработки. Данное высокое значение, безусловно, связано с небольшим опытом участников команды, однако характеризует то, что команда очень неэффективно использует подходы к планированию спринтов. Также важно отметить, что 86% из полученных результатов связаны с явлением переоценки, что приводит к простою команды в течение спринта и значительному замедлению темпов выполнения проекта. При условии, что компания оплачивает 40-часовую рабочую неделю сотруднику простои команды, даже на внутренних проектах, означает снижение экономической эффективности и возможную потерю прибыли.

Стоит заметить, что фактическая стоимость в Story Points в рассматриваемой компании оценивается путем. После тестирования модели в течение 4 спринтов была собрана и агрегирована информация, представленная в таблице 3:

Таблица 3. Результат тестирования модели

Метрика	Номер спринта			
	1	2	3	4
Количество задач	14	17	11	13
Количество членов команды	9	9	9	9
Предсказываемая суммарная стоимость итерации в SP	96	92	96	80
Фактическая суммарная	78	82	83	75

стоимость итерации в SP				
Средняя относительная погрешность	0,24	0,13	0,16	0,07
Показатель истинности предсказывания	57%	70%	63%	77%

Анализируя таблицу 3, можно сделать вывод о том, что качество модели улучшается с течением времени, однако в 3 спринте этого не произошло из-за наличия более сложных, объемных задач, которые намного труднее оценивать в Story Points, так как Story Points обычно подразумевает сравнение с некоторым эталоном. Стоит заметить, что в рамках оценки стоимости итерации во всех спринтах команда сталкивалась с переоценкой стоимости, что потенциально приводит к простою команды.

Полученные с использованием построенной модели результаты превосходят точность оценки игрой в Scrum Poker малоопытной командой, как это часто используется на внутренних проектах компании, что позволяет предположить оптимальность использование подобного подхода для оценивания стоимости итерации/проекта в Agile-методологиях. При использовании подхода к оценке стоимости с использованием разработанной модели возможно снижение количества недооценок или переоценок на проектах, которые особенно свойственны проектам с большим количеством малоопытных участников, что приведет к экономической оптимизации процесса, снижения уровня недополученной прибыли или нерациональных затрат на оплату труда из-за неверной оценки стоимости.

Источники

1. Джонс, Чад Экономика качества программного обеспечения / Джонс Ч., Бонсигнор О. М.: ЭКОМ Паблишерс , 2012. 367 с.
2. Швабер Кен The Scrum Guide // Официальный сайт методологии Скрам. 2018. URL: https://www.scrum.org/ - (дата обращения 08.11.2019).
3. Техническая документация к системе отслеживания ошибок Atlassian Jira // Официальный сайт компании Atlassian. 2019. URL: https://confluence.atlassian.com/jirasoftwarecloud/jira-software-documentation-764477791.html - (дата обращения: 17.10.2019)
4. Project Management Body of Knowledge 6th Edition / Институт проектного менеджмента / Пенсильвания: PMI Puiblishers. 2017. 913с.
5. Хелдман, Ким Профессиональное управление проектами. / Ким Хелдман. М.: Бином, 2005. 517 с.
6. ДеМарко, Том Deadline. Роман об управлении проектами / Том ДеМарко. М.: Вершина, 2006. 415 с.
7. Макконнелл, К. Р. Экономикс: принципы, проблемы и политика. / Крис Макконелл. М.: Республика, 1992. 507 с.
8. Орлов, А. И. Эконометрика. Учебник. / А. И. Орлов. М.: Экзамен, 2002. 576 с.
9. Самарский, А. А. Математическое моделирование: Идеи. Методы. Примеры. / А. А. Самарский, А. П. Михайлов М.: Наука, 1997. 320 с.
10. Дэвид Штраубе Ежегодный отчет о результативности компаний // Cision PR news journal. 2019. URL: https://www.prnewswire.com/news-releases/epam-reports-results-for-fourth-quarter-and-full-year-2018-300795612.html - (дата обращения: 21.11.2019)
11. Ирани, Зубин. 5 общих проблем в CICD и как избежать их // Computer Science journal InfoWorld. 2018. URL: https://www.infoworld.com/article/3113680/5-common-pitfalls-of-cicd-and-how-to-avoid-them.html - (дата обращения: 18.05.2019)
12. Ньюэлл, Майкл Управление проектами для профессионалов. Руководство по подготовке к сдаче сертификационного экзамена. / Майкл Ньюэлл М.:Кудиц-пресс, 2008. 416 с.
13. Леоненков, А.В. Нечеткое моделирование в среде MATLAB и fuzzyTECH / А. Леоненков. / СПб: БХВ-Петербург, 2003. 736 с.
14. Круглов, В. В. Нечеткая логика и искусственные нейронные сети. / В. В. Круглов, М. И. Дли, Р. Ю. Голунов. М.:Физматлит, 2000. 224 с.
15. Штовба, С.Д. Проектирование нечетких систем средствами MATLAB / С. Штовба. М.: Горячая линия Телеком, 2007. 288 с.

Программно-аппаратный комплекс сбора, хранения и визуализации метеорологических данных с метеостенда

Панков Павел Александрович
Институт компьютерных наук и технологий
Санкт – Петербургский Политехнический университет Петра Великого
Санкт – Петербург, Российская Федераци
pankov.pavel.a@gmail.com

Никифоров Игорь Валерьевич
Интститут компьютерных наук и технологий
Санкт – Петербургский Политехнический университет Петра Великого
Санкт – Петербург, Российская Федерация
i.nikiforov@ics2.ecd.spbstu.ru

Аннотация **– В современном мире применение измерительных датчиков и микроконтроллеров растёт с каждым годом. Диапазон их использования начинается с бытовых приборов и заканчивается промышленной автоматикой. В статье описан процесс разработки и создания прогрммно-аппаратного комплекса предназначенного для сбора, хранения и визуализации метеорологичесих данных. Система включает в себя стенд с метеодатчиками, микроконтроллер и программное обеспечение, позволяющее принимать данные со стенда с микроконтроллером и работать с базой данных MongoDB. В результате исследования и работы был разработан тестовый образец программно–аппаратного комплекса, обеспечивающего отслеживание атмосферных показателей окружающей среды, их обработку, хранение и визуализацию. Кроме того, описаны возможности дальнейшей разработки проекта.**

Ключевые слова – микроконтроллер, датчик, метеоданные, хранение данных, большие данные, обработка данных.

I. Введение

Устройства, включающие в себя датчики и микроконтроллеры, прочно заняли место в нашей жизни. Нас везде окружают измерительные датчики, такие как климатические, датчики пространства, света и др., а также окружают и микроконтроллеры. Данные компоненты применяются как в простейших бытовых приборах, таких как кондиционер, так и в сложнейших системах управления сборочными цехами на промышленных предприятиях [1] или могут быть использованы в медицинских сферах [2].

Люди ежедневно сталкиваются с подобными устройствами. Это могут быть интерактивные зеркала, отображающие новостную ленту, или системы биометрической идентификации.

В повседневной жизни людям важно учитывать погодные условия, однако прогнозы погоды по телевизору или в сети Интернет не всегда точны, а привязка к местности может быть неточной. Поэтому актуальной является необходимость в программно-аппаратном комплексе, для получения информации о настоящей погоде за окном, а именно температуре, давлении и влажности, проводя обработку полученной информации в программной системе и делая прогноз метеорологической ситуации на будущее.

Программно-аппаратный комплекс должен собирать актуальную информацию с датчиков, стоящих внутри помещения или на улице, и иметь возможность визуализировать результаты работы на мобильном приложении или стационарной рабочей станции.

Кроме того, подобную систему можно использовать в таких местах как автомобильный гараж, где полезно, знать метеорологические показатели в гараже [3] для проведения работ, на даче, для проведения строительных работ или иметь подобные стенды в промышленных помещениях, чтобы контролировать и соблюдать технологические процессы [4]. Также, такой программно-аппаратный комплекс можно включить в систему управления кондиционерами и отопительными системами. Примером может служить поддерживание заданной температуры или влажности в помещении, основываясь на показаниях датчиков.

II. Архитектура программно-аппаратного комплекса

В статье предлагается следующая структура программно-аппаратного комплекса, представленная на рис.1., в которую входит стенд с метеорологическими датчиками и микроконтроллером, собирающим и обрабатывающим данные с датчиков и программное обеспечение для работы с этими данными для локальной или мобильной рабочей станции.

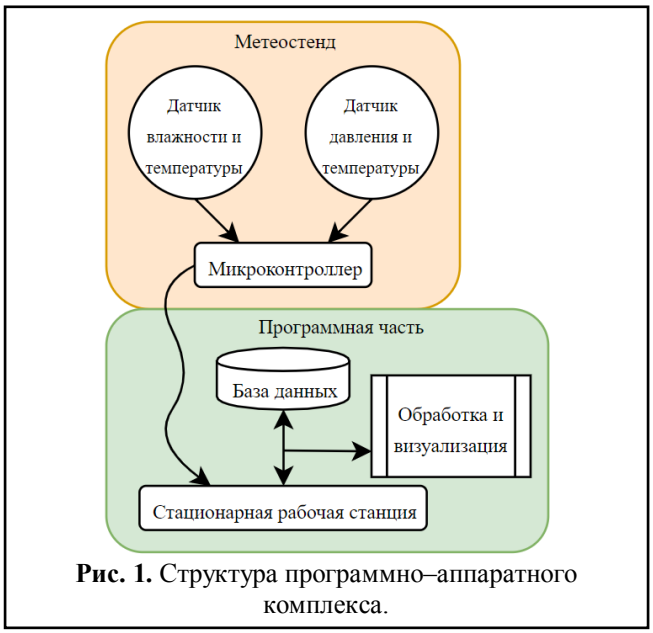


Рис. 1. Структура программно–аппаратного комплекса.

Можно выделить следующие основные модули:

- датчики окружающей среды, а именно датчик температуры, датчик атмосферного давления и датчик влажности;

- контроллер, который отвечает за опрос датчиков, сбор информации и её отправку по сети на компьютер;
- программная система, которая обеспечивает приём данных с контроллера, обработку принятых данных, её запись в базу данных и визуализацию.

В свою очередь программная система состоит из:

- базы данных;
- модуля приёма и записи данных;
- модуля обработки и визуализации данных.

База данных хранит информацию, полученную с микроконтроллера.

Модуль приёма и записи данных принимает пакеты данных с микроконтроллера по сети, обрабатывает принятые пакеты и записывать данные в базу.

Модуль визуализации обеспечивает графическое представление данных, полученных и обработанных по необходимости из базы.

Главное отличие разрабатываемого проекта от существующих метеорологических станций на рынке, таких как Oregon Scientific BA900[5], заключается в том, что данные с датчиков не просто собираются и отображаются, а весь объём данных хранятся в базе определённое время для возможности их дальнейшего анализа, визуализации статистики [6].

III. Выбор микроконтроллера

Одной из задач, решаемых в работе, является выбор элемента, который будет собирать данные с датчиков и отправлять их в локальную рабочую станцию.

Было рассмотрено несколько вариантов микроконтроллеров [7]: Arduino [8-9], STM 32 [10], ESP 32[11], Raspberry Pi[12-14].

Введем основные критерии выбора микроконтроллеров:

- стоимость (К1). Учитывается стоимость не самого микроконтроллера, а отладочной платы, то есть микроконтроллер и необходимые компоненты для его полноценной работы;
- тактовая частота (К2). Тактовая частота – это количество тактов (операций) процессора в секунду. Чем выше тактовая частота микроконтроллера, тем выше его производительность;
- поддержка каналов передачи информации по Wi-Fi (К3). Наличие доступа к сети Ethernet, используя Wi – Fi, обязательно для беспроводного обмена данными;
- готовые библиотеки для работы с выбранными датчиками (К4). Данный критерий так же важен, т.к. наличие готовых библиотек для работы с тем или иным датчиком значительно сокращает время разработки. Обычно, подобные библиотеки находятся в открытом доступе, и в них описаны все возможные функции, поддерживаемые датчиком. Готовую библиотеку можно взять за основу разрабатываемой, оптимизировать и оставить только необходимые функции;
- наличие сообщества для решения проблем инженеров-разработчиков (К5). Накопленный опыт участников сообщества, находящийся в открытом доступе, и поддержка форумов разработчиков упрощает процесс разработки и ускоряет решение возникающих проблем.

В таблице 1 приведены характеристики рассматриваемых микроконтроллеров, где “+”означает присутсвие того, что описано в критерии и “-“ означает его отсутвие.

Таблица 1. Характеристики микроконтроллеров.

Микроконтроллер	К1, руб	К2, МГц	К3	К4
Arduino	100 – 1000	8 – 48	Доп. плата	+
STM 32	1000 – 6200	24 – 216	Доп. плата	-
ESP 32	400 – 600	160/240	+	+
Raspberry Pi	2900 – 3800	1400	+	-

К преимуществам микроконтроллера Arduino. можно отнести:

- из рассмотренных микроконтроллеров имеет самую низкую стоимость отладочных плат;
- наличие готовых библиотек для работы с требуемыми датчиками;
- сообщество Arduino большое и в интернете можно найти ответы практически на любые возникшие вопросы при разработке на русскоязычных и англоязычных форумах 1,2.

Однако, у данных микроконтроллеров есть ряд недостатков:

- низкая тактовая частота микроконтроллеров, по сравнению с рассматриваемыми аналогами;
- отсутствие готовых решений, имеющих Wi-Fi.

Следующими были рассмотрены микроконтроллеры STM 32:

- тактовая частота данных микроконтроллеров выше микроконтроллеров Arduino;
- для каждого микроконтроллера есть полноценное описание его периферии и возможностей.

Однако, с другой стороны:

- микроконтроллеры STM 32 требуют высококвалифицированных специалистов для корректной разработки и написания кода;
- как видно из таблицы, отладочные платы STM 32 дороже Arduino и ESP 32;
- также, как в случае с Arduino, STM32 не имеет

[1] http://arduino.ru/forum

[2] https://forum.arduino.cc/

готовых отладочных решений для работы с Wi-Fi, а требует использования дополнительных плат.

Так же, были рассмотрены микроконтроллеры ESP 32. Они имеют следующие преимущества:

- высокая производительность. На ESP 32 установлены 32-х разрядные двухъядерные процессоры с тактовой частотой до 240 МГц;
- наличие Wi-Fi модуля, без использования дополнительных плат;
- существует набор инструментов Platform IO[3] и оболочка, позволяющая писать код под ESP 32 используя код и библиотеки, написанные для Arduino с небольшими изменениями или без таковых.

Последними были рассмотрены микрокомпьютеры Raspberry Pi. Они являются другим классом устройств, но могут выполнять те же функции по работе с датчиками и имеют возможность подключения к сети. Кроме того, на такие устройства есть возможность поставить полноценную операционную систему Raspbian[8], основанную на Linux Debian, имеющую незначительные недостатки в условиях задачи. Однако, стоимость данных устройств в несколько раз выше перечисленных ранее. Так, Raspberry Pi 3 B/B+ в 7-9 раз дороже микроконтроллера ESP 32, поэтому данный вариант не учитывался на начальных этапах.

В результате проведённого сравнительного анализа, был выбран микроконтроллер ESP 32, т.к. его возможностей достаточно для выполнения поставленной задачи, а стоимость является одной из самых низких из рассмотренных.

IV. Выбор Датчиков

В работе был проведен выбор датчиков для получения следующих данных:

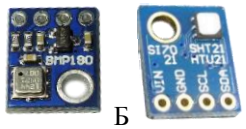
- температура окружающей среды;
- атмосферное давление;
- влажность.

Для сравнения были выбраны следующие датчики:

- BMP180 GY-68. Совмещает в себе датчик атмосферного давления и датчик температуры. Интерфейс датчика I2C. Точность определения давления 0.02 hPa. Точность определения температуры 0.01 градуса по Цельсию;
- SI7021 HTU21. Совмещает в себе датчик влажности и датчик температуры. Интерфейс датчика I2C. Точность определения влажности 3% RH. Точность определения температуры 0.4 градуса по Цельсию.

Таким образом, датчик BMP180 используется как датчик атмосферного давления и датчик температуры, а датчик SI7021 выступает в роли датчика влажности, так как его точность определения температуры гораздо ниже, чем точность датчика BMP180. Датчики изображены на рис.2.

[3] https://platformio.org/



А Б

Рис. 2. А Датчик атмосферного давления и температуры BMP180 GY-68. **Б** Датчик влажности и температуры SI7021 HTU21.

V. Система сбора, хранения и видуализации метеорологических данных с метеостенда

A. *Работа с базой*

Для хранения данных была выбрана документно-ориентированная база данных Mongo DB [15-16].

В связи с тем, что Mongo DB не позволяет работать напрямую с микроконтроллером ESP 32, и из рассмотренных библиотек не было найдено готовой для решения этой задачи, было решено осуществлять запись данных в локальную базу MongoDB с компьютера, а данные передавать по сети, чтобы исключить использование проводной связи.

B. *Этапы разработки системы*

Процесс разработки включал в себя следующие стадии:

- конструирование и сбор стенда с метеодатчиками и микроконтроллером ESP 32;
- написание прошивки для микроконтроллера ESP 32, выполняющую сбор данных с датчиков и отправку данных по сети, используя домашнюю сеть Wi-Fi;
- написание программного обеспечения связи компьютера с микроконтроллером и записи данных в базу;
- написание программного обеспечения чтения данных с базы, их обработка, анализ и визуализация.

C. *Конструирование стенда*

Как было указано ранее, стенд включает в себя микроконтроллер ESP 32, датчик атмосферного давления и температуры BMP180 GY-68 и датчик влажности SI7021.

Интерфейс коммуникации с датчиками I2C [17-18]. На микроконтроллере ESP32 одна шина данных с интерфейсом I2C, тем не менее, данный интерфейс позволяет использовать множество датчиков, а для выбора необходимого датчика используется его адрес.

Из – за того, что датчики необходимо расположить на улице, их использование вне корпуса невозможно, так как осадки могут привести к короткому замыканию или другим ситуациям, которые приведут к негативным последствиям, от поломки датчика до поломки всей *системы.*

С другой стороны, заключать датчики в полностью герметичный корпус также нельзя, в таком случае, показания датчиков будут отличаться от тех, которые будут вне корпуса.

Учитывая вышесказанное, в корпусе были сделаны

технологические отверстия в противоположной от датчиков стенке, чтобы показания датчиков внутри корпуса были максимально схожи с окружающими.

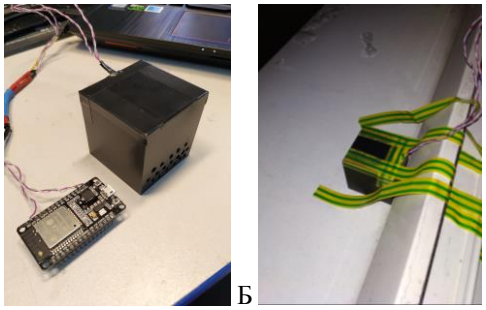
Кроме того, для защиты датчик атмосферного давления был защищён чёрным поролоном, так как этот датчик чувствителен к свету, влаге и ветру, которые могут исказить его показания.

Выбирая источник питания и расположение микроконтроллера, было два решения:

- Расположить микроконтроллер в корпусе и использовать для питания аккумуляторы;
- Расположить микроконтроллер отдельно от датчиков вне корпуса и использовать для питания аккумулятор или бытовую сеть с адаптером питания.

Размещение микроконтроллера с элементами питания внутри корпуса потребовало бы не только значительного увеличения размеров самого корпуса, но и усложнило бы способы крепления и герметизации отсека элементов питания в корпусе. Для перезагрузки или перепрошивки необходимо бы было каждый раз вытаскивать корпус со всем содержимым в помещение, что опять же искажало бы показания датчиков, так как требуется определённый промежуток времени, что бы показания датчиков стабилизировались и показывали реальные данные. С учётом того, что надёжно закрепить корпус снаружи помещения не было возможности, возникал риск потери всего стенда в едином корпусе.

В результате было принято решение разместить датчики в корпусе, закреплённом вне помещения, а микроконтроллер с питанием внутри, что обеспечило быстрый доступ к микроконтроллеру для перезагрузки и перепрошивки. Собранный стенд показан на рис.3.



А Б

Рис. 3. А. Собранный стенд с датчиками в корпусе. **Б.** Корпус с датчиками, установленный вне помещения

D. Алгоритм сбора и отправки данных на микроконтроллере

К прошивке микроконтроллера были следующие требования:

- Требуется постоянный опрос датчиков и сбор актуальных данных;
- Должна быть реализована отправка данных на компьютер по сети, по команде готовности к приёму данных от компьютера.

Прежде всего, в работе были реализованы библиотеки для взаимодействия с датчиками.

В начале работы микроконтроллер осуществляет тестовые запросы к датчикам для определения их присутствия и корректной работы. В случае, если какой-то из датчиков не отвечает или отвечает некорректно, микроконтроллер сообщит об этом. В случае же, если оба датчика корректно отвечают, микроконтроллер продолжит свою работу.

После того как микроконтроллер осуществил подключение к домашней Wi-Fi сети, делаются тестовые запросы к датчикам и запускается сокет – сервер.

После подключения к серверу клиента, микроконтроллер ожидает от клиента команды готовности к приёму данных, продолжая опрашивать датчики для получения последних актуальных данных.

Приняв команду о готовности к приёму данных, микроконтроллер отправляет 4 пакета с актуальными показаниями датчиков. Алгоритм изображён на рис.4.

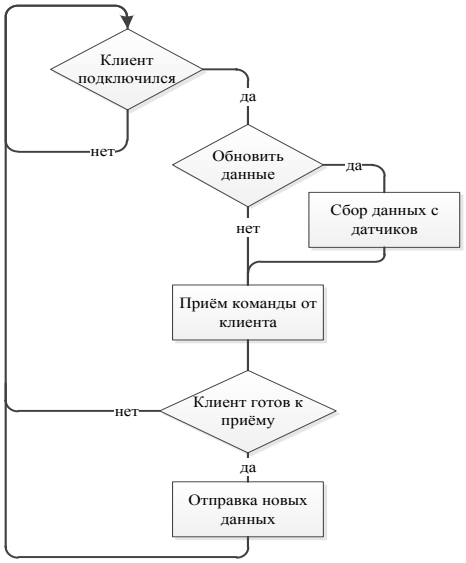


Рис. 4. Алгоритм обновления данных с датчиков, приёма команды с от клиента и отправка ему данных с датчиков

Пакет данных содержит в себе префикс пакета, идентификационный номер команды, количество «полезных байт», содержащих в себе данные с датчиков, сами данные и символ конца пакета.

После отправки данных, микроконтроллер снова ждёт команды готовности к приёму от клиента, обновляя при этом данные с датчиков.

E. Приём данных и запись данных в базу на компьютере

Для приёма данных с микроконтроллера и записи данных в базу был написан скрипт на языке Python [19-20].

После запуска скрипта, производится подключение к локальной базе данных Mongo DB, для работы с которой используется модуль pymongo.

После подключения к базе осуществляется подключение к серверу, используя IP адрес, который ESP 32 получает после подключения к Wi-Fi сети. Затем скрипт отправляет команду готовности к приёму данных и ожидает ответа от микроконтроллера.

В случае, если время ожидания новых данных превышает заданное время, скрипт продолжает свою работу, сообщив об ошибке.

После приёма сообщения, полученный пакет разбирается в соответствии с описанным протоколом приёма – передачи данных.

Если принятый пакет данных был корректно разобран и идентифицирован, полученные данные записываются в базу. В противном случае, выводится сообщение об ошибочном пакете или несуществующем идентификаторе. Алгоритм изображён на рис.5.

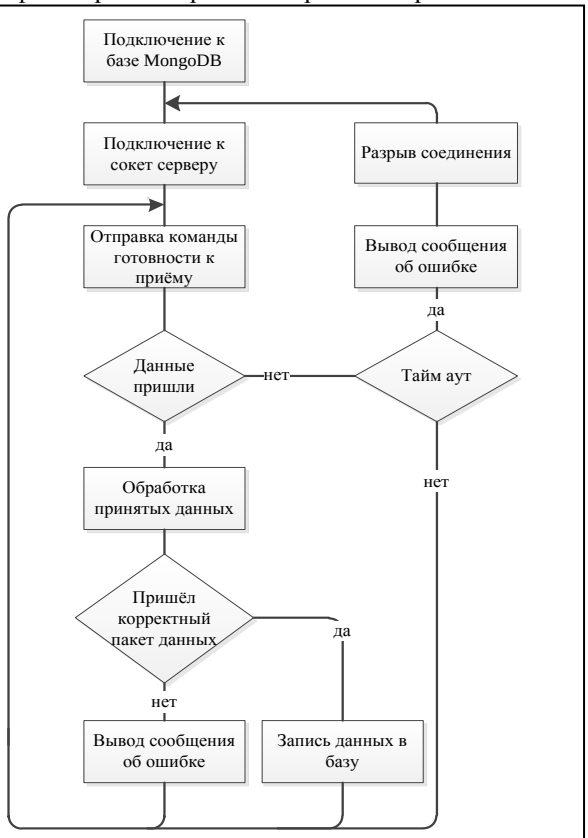


Рис. 5. Алгоритм приёма данных от клиента, их разбор и запись в базу

Ниже приведён пример формата записи данных в базу в формате JSON:

```
{
    sensor: "HTU Temperature"
    data: "-0.16"
    date: "2018_12_12"
    time: "23:33:12:565"
}
```

F. Чтение и визуализация данных из базы

Для чтения и визуализации данных из базы данных MongoDB было написано следующее программное обеспечение:

- Python скрипт, использующий для подключения к базе Apache Spark [14] в виде модуля pyspark[4];
- Python скрипт, использующий для подключения к базе стандартные методы модуля pymongo[5];
- Python скрипт, который в реальном времени строит график изменения показаний датчиков по последним актуальным данным из базы;
- Программа, написанная на языке C#, также в реальном времени демонстрирует последние актуальные данные с датчиков в виде простого виджета с тремя полями и графиков изменения показателей.

Первый Python скрипт демонстрирует работу с базой MongoDB с использованием модуля pyspark. Скрипт получает все данные указанного датчика и строит график изменения показателей во времени с указанием среднего за всё время, за которое были сняты показания. Пример настройки SparkSession модуля pyspark и запроса представлен в Приложении А.

Второй Python скрипт демонстрирует возможности использования стандартного модуля pymongo. Запуск скрипта возможен с указанием параметров:

- Название датчика, данные которого требуются;
- Дата, за которую требуются данные. Данным параметром также можно указать «today» или «yesterday», в результате чего данные будут собраны за сегодняшний или вчерашний день;
- Если данным параметром указана дата, то она является концом периода съёма данных, а второй параметр его началом. Данный параметр также предусматривает вариант «today».

В случае, если скрипт запускается без указания каких-либо параметров, запрашиваются все данные по датчику, указанному по умолчанию.

После разбора переданных параметров, скрипт получает из базы данные по указанному датчику за указанный период времени и строит по ним график, также указывая средний уровень.

Третий Python скрипт также принимает параметром название датчика. После запуска скрипт в реальном времени строит график изменения показаний датчика с момента запуска. Подключение к базе осуществляется так же, как во втором примере, с использованием модуля pymongo.

Программа, написанная на языке C#, представляет собой два виджета в виде двух полупрозрачных окон. В первом окне отображается актуальная информация по трём датчикам, запрашиваемая с базы. Второе окно содержит в себе 3 графика изменения показаний датчиков с момента запуска программы.

После запуска программы и установления связи с базой, запускается таймер, запрашивающий данные из базы и отображающий эти данные на формах.

Для проекта был создан репозиторий на GitHub, где содержится весь исходный код. Репозиторий доступен по ссылке[6].

[4] https://spark.apache.org/docs/0.9.0/python-programming-guide.html

[5] https://api.mongodb.com/python/current/

[6] https://github.com/DucklingDark/Meteo_Station

VI. Примеры работы системы

A. *Python скрипты*

Для запуска скрипта, использующего pyspark, необходимо использовать утилиту spark – submit с указанием коннектора, совместимого с установленной версией Spark и MongoDB. В конце указывается файл Python скрипта. Пример запуска Python скрипта с использование spark – submit представлен в Приложении Б.

Для запуска остальных Python скриптов этого не требуется.

Пример визуализации данных средствами Python представлен на рис.1 Приложения В.

B. *Проект, написанный на языке C#*

После запуска программы открываются два окна. Оба окна выполнены в виде полупрозрачных виджетов с возможностью перемещения по всему экрану:

- Первое окно содержит в себе три поля с названием показаний и само значение, актуальное на текущий момент;
- Второе окно содержит в себе 3 графика, на которых отображаются изменения показаний датчиков.

При необходимости, каждое из окон можно закрыть, оставив второе. Программа остановит свою работу только после закрытия обоих окон.

Пример работы программы представлен на рис.2 Приложения В.

Заключение

В результате проделанной работы был создан программно-аппаратный комплекс, с помощью которого можно собирать метеорологические данные, хранить, визуализировать и обрабатывать. В работе были описаны сравнительные анализы микроконтроллеров и датчиков, исходя из которого были выбраны ESP 32 и датчики BMP 180 и SI7021 HTU 21D.

Для первых этапов разработки микрокомпьютеры Raspberry Pi не рассматривались из-за своей дороговизны, однако, их использование может упростить и улучшить систему, поспособствовав её развитию.

Кроме того, для полноты показаний стенда и системы можно добавить датчик скорости и направления ветра, однако данные датчики дороги и производятся в большинстве случаев для использования в промышленных сферах.

Следующими этапами разработки может быть перенос базы данных в облачные сервисы, а вместо ESP 32 использовать Raspberry Pi. Кроме того, в записи в базу необходимо добавить координаты стенда или краткое название помещения. С другой стороны, можно использовать данные стенды в разных помещениях, в которых требуется знать подобные показатели, например, для системы поддержания температуры и влажности. Это позволит объединить несколько систем в одну, централизованную.

Подобное развитие системы позволит получать доступ к данным не только с компьютера, на котором расположена база данных, а из любого места при наличии доступа в сеть Интернет.

Также, в базу данных необходимо отправлять сообщения об ошибках и отслеживать их появление, чтобы своевременно исправлять неполадки в работе стенда и системы, а по возможности прогнозировать их [20].

Список Литературы

1. Kapustin, N. M. Automation of production processes in mechanical engineering: Proc. for technical colleges / Ed. N. M. Kapustina. - M.: High School , 2004. — 415 p.
2. Salvatore Vitabile, Michal Marks, Dragan Stojanovic, Sabri Pllana, Jose M. Molina, Mateusz Krzyszton, Medical Data Processing and Analysis for Remote Health and Activities Monitoring, https://link.springer.com/chapter/10.1007/978-3-030-16272-6_7
3. I.Kasatkin, M. Egorov, E. Kotov, E. Zakhlebaev. Cooling of a battery pack of a car, working on renewable energy. MATEC Web Conf., Volume 245, Article number 15003, 2018. https://doi.org/10.1051/matecconf/201824515003.
4. Helmi A. Youssef, Hassan A. El-Hofy, Mahmoud H. Ahmed, Manufacturing Technology: Material, Processes, and Equipment, 2011 – 948 p. BA900 Weather station. URL: https://mobile-review.com/articles/2007/oregon-scientific-ba900.shtml
5. L.U. Laboshin, A.A. Lukashin, V.S. Zaborovsky. The Big Data approach to collecting and analyzing traffic data in large scale networks. Procedia Computer Science, Volume 103, 2017, pp. 536-542. DOI: 10.1016/j.procs.2017.01.048
6. Markus Knoll, Philipp Breitegger, Alexander Bergmann, Low-Power Wide-Area technologies as building block for smart sensors in air quality measurements, https://link.springer.com/article/10.1007/s00502-018-0639-y
7. Jack Purdem. Beginning C for Arduino, Second Edition: Learn C Programming for the Arduino, 2nd Edition, 2012 – 276 p.
8. Jeremy Blum. Exploring Arduino, 1st Edition, 2017 – 387 p.
9. Carmine Noviello, Mastering the STM 32 Microcontroller, 2016 – 782 p.
10. Neil Kolban, Kolban's book on ESP 32, 2018 – 1228 p.
11. Simon Monk, Raspberry Pi Cookbook, 2013 – 412 p.
12. Simon Monk, Programming the Raspberry Pi: Getting Started with Python, 2012 – 192 p.
13. William Harrington, Learning Raspbian, 2015 – 154 p.
14. Kristina Chodorow, MongoDB: The Definitive Guide: Powerful and Scalale Data Storage, 2013 – 432 p.
15. Katarina Grolinger, Wilson A Higashino, Abhinav Tiwari, Miriam AM Capretz, Data management in cloud environments: NoSQL and NewSQL data stores, https://link.springer.com/article/10.1186/2192-113X-2-22
16. Jonathan Valdez, Jared Becker, Understanding the I2C Bus, 2015 – 8p.
17. Vincent Himpe, Mastering the I2C Bus, 2011 – 247 p.
18. Wes McKinney, Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython, 2012 – 466 p.
19. Brian Jones, David Beazley, Python Cookbook, 3rd Edition. Recipes for Mastering Python 3, 2013 – 706 p.
20. L.V. Utkin, V.S. Zaborovskii, S.G. Popov. Detection of anomalous behavior in a robot system based on deep learning elements. Automatic Control and Computer Sciences. Volume 50, Issue 8, 2016, pp. 726-733. DOI: 10.3103/S0146411616080319

Software and hardware complex for collecting, storing and visualizing meteorological data from a weather stand

Pavel A. Pankov, Igor V. Nikiforov

In the modern world, the use of measuring sensors and microcontrollers is growing every year. The range of their use begins with household appliances and ends with industrial automation.

The article describes the process of developing and creating a software and hardware complex designed for collecting, storing and visualizing meteorological data. The system includes a stand with weather sensors and a microcontroller and software that allows you to receive data from the stand with a microcontroller and work with the MongoDB database.

As a result of the research and the work, a test sample of a software and hardware complex was developed, which ensures the tracking of atmospheric environmental indicators, their processing, storage and visualization. In addition, the possibilities for further project development are described.

Keywords - microcontroller, sensor, weather data, data storage, big data, data processing.

Приложение А: Настройка SparkSession модуля pyspark

```
spark = SparkSession \
    .builder \
    .appName("Meteo_Station") \
    .config("spark.mongodb.input.uri", uri_conf) \
    .config("spark.executor.memory", "4g") \
    .config("spark.executor.number", "4") \
    .config("spark.executor.cores", "4") \
    .config("spark.network.timeout", "800s") \
    .config('spark.yarn.executor.memoryOverhead', "4096m") \
    .config("spark.driver.memoryOverhead",) \
    .getOrCreate()

df = spark.read.format("com.mongodb.spark.sql. DefaultSource").load()
pressure = spark.sql('SELECT * FROM temp WHERE sensor = \'BMP Pressure \' ')
```

Приложение Б: Пример запуска Python скрипта с использование spark – submit

```
spark-submit-packages org.mongodb.spark:mongo-spark-connector_2_11:2.2.0 .\mongo_spark.py
```

Приложение В: Пример работы разработанных приложений и визуализация данных

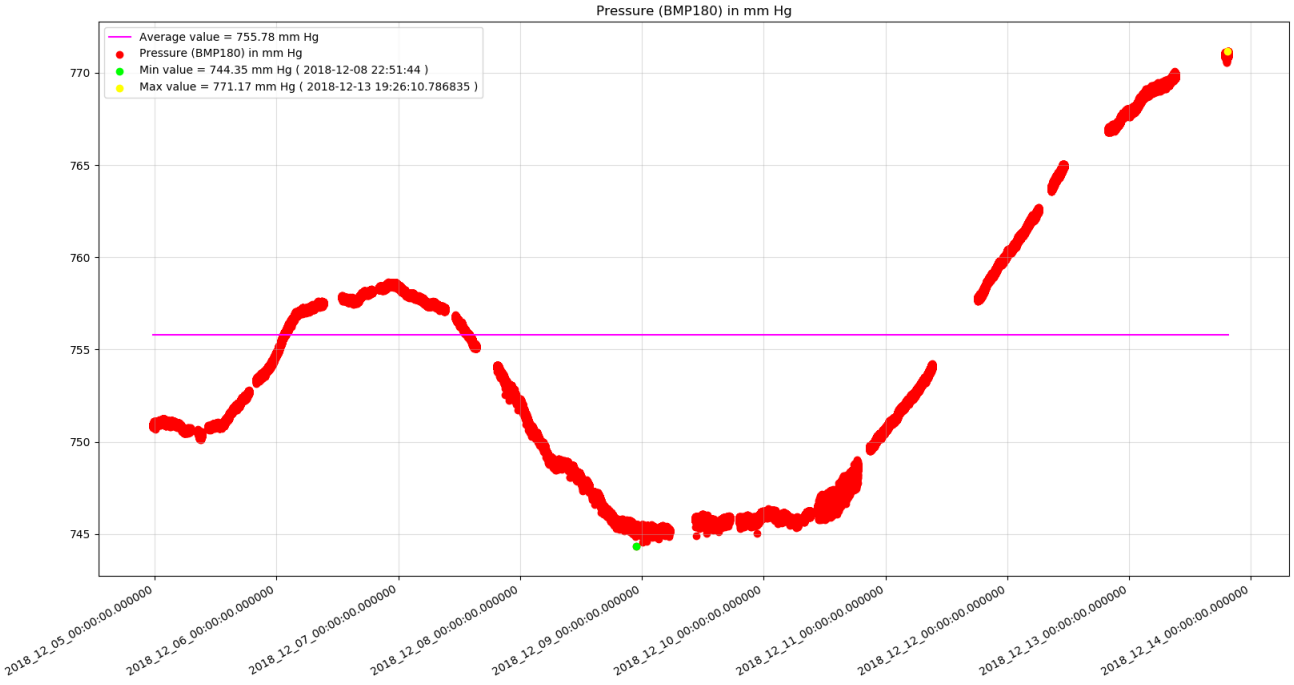
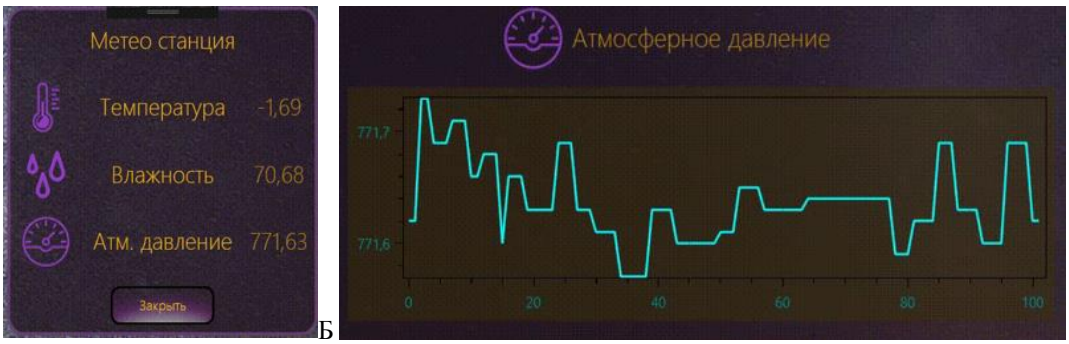


Рис. 1. Пример визуализации данных Python скрипта



А Б

Рис. 2. А Окно программы, написанной на C# с актуальными цифровыми значениями датчиков. **Б** График изменения атмосферного давления в течении 10 секунд.

Мобильное приложение для поиска рекомендаций целевого контента

Абылкасымова А.С.
Санкт-Петербургский политехнический университет Петра Великого
Санкт-Петербург, Россия

Аннотация. Очень часто пользователь при просмотре аниме сталкивается с проблемой: что же посмотреть далее? Какой фильм из огромного количества постоянно растущего материала с наибольшей долей вероятности понравится пользователю при последующем просмотре. В настоящее время большое количество популярных видеохостингов используют рейтинговые системы рекомендаций, которые опираются на совокупность субъективных оценок сообщества. Для решения подобного спектра задач используются различные методы, основанные на контенте. Основным недостатком подобных методов является необходимость дополнительной информации, которую обычно дорого и долго собирать. В этой статье мы используем технику глубокого обучения, Illustration2Vec, для простого извлечения информации о тегах из постеров манги и аниме (например, меча или хвоста). Прототип приложения был опробован на десятках профилей людей на сайте и крупной базе по аниме MyAnimeList и в ближайшее время запланирована доработка приложения. Также в статье рассматриваются существующие системы формирования индивидуальных рекомендаций.

I. ВВЕДЕНИЕ

Рекомендательные системы в сфере видео-досуга – отличный инструмент, помогающий пользователю ответить на вопрос: как можно приятно провести время в следующий раз, что стоит посмотреть? В случае аниме и манги (японского мультипликационного кинематографа и литературы) пользователи обычно перегружены огромным количеством постоянно обновляющегося материала, что мешает им отыскать фильмы и материалы, которые могут им очень понравиться. Частым решением данной проблемы является коллаборативная фильтрация, опирающаяся на существующие рейтинговые данные (оценки) пользователей для попытки предсказания рейтинга не просмотренного материала[1]. Однако большие трудности возникают при использовании данного подхода в отношении объектов, для которых имеется очень мало информации. Данная проблема известна под названием «проблема холодного старта».

Холодный старт (ввод) объектов - это классическая проблема, возникающая при разработке различных рекомендательных систем. Для решения данной проблемы часто используются различные метаданные, например, для фильмов подобными данными могут являться режиссер, композитор, актеры и так далее. К сожалению, подобная информация далеко не всегда доступна и недавно вышедшие аниме-проекты могут иметь минимальное количество подобной информации. Однако практически всегда у такого проекта будут представлены трейлер, постер и заголовок. Эти три вещи являются ключевыми, так как символизируют первый контакт пользователя с контентом и играют большую роль в решении пользователя: смотреть данный материал или не смотреть.

В манге и аниме постеры содержат большое количество информации о персонажах и сюжете, чтобы максимально заинтриговать потенциальных потребителей, увеличив визуальную привлекательность. Таким образом, постеры можно рассматривать как источник дополнительных метаданных.

За последние годы сверточные нейронные сети (CNN) зарекомендовали себя как де-факто самый распространенный и эффективный метод извлечения семантической информации из содержимого изображения в широком спектре задач. В ходе данного исследования было принято решение использовать сверточные нейронные сети для извлечения значимых элементов метаданных (тэгов) из постера, соответствующего аниме или манги. Такие тэги позволят построить граф зависимостей между различными произведениями, что может быть полезно, когда количество информации об объекте мало.

В данной работе предполагается разработка алгоритма, базирующегося на альтернативных/улучшенных методах наименьших квадратов для решения проблемы холодного старта и повышения эффективности рекомендаций малоизвестных объектов. В рамках исследования было проанализировано достаточно большое количество разнообразных существующих решений и проведены соответствующие сравнения, показывающие, что во многих случаях метод обеспечивает более точные прогнозы рейтинга и дает интерпретируемое представление о пользовательских предпочтениях.

II. СУЩЕСТВУЮЩИЕ РЕШЕНИЯ

Использование дополнительной информацией (метаинформации) для улучшения качества рекомендаций довольно давно является основой существующих исследований[2], направленных на разработку подходов получения дополнительной информации о пользователях и предметах из текста[3], профилей в социальных сетях, где задача факторизации методами машинного обучения заключается в минимизации взвешенного квадрата отклонений для всех элементов исходной матрицы R:

$$\min \sum_{i,j} c_{ij} (r_{ij} - x_i^T y_j)^2 ,$$

где x_i^T i x – транспонированная i – строка матрицы «пользователь - латентная переменная»; y – строка j матрицы «латентная переменная – объект». На этапе факторизации матриц

предлагается использовать метод ALS (Alternating Least Squares Method). Особенность данного метода состоит в поочередном нахождении минимума: то для элемента «пользователь – переменная» при фиксации элемента «переменная – объект», то для элемента «переменная – объект» при фиксированных элементах «пользователь – переменная» [4]. Также изображений и других типах данных [5,6,7]. В последнее время для этих целей все чаще применяются методы глубокого обучения. Портал Youtube использует историю просмотров своих пользователей, чтобы на основе этого улучшить свои рекомендации[8]. Некоторые исследователи анализировали музыкальный контент для получения метаинформации. Подобные исследования позволили выявить скрытые закономерности в музыке без какого-либо человеческого вмешательства[9].

Дополнительная информация, полученная из любого из вышеприведенных источников очень важна при попытке уменьшить воздействие проблемы холодного запуска[10,11,12]. Гибридный подход, описанный в данных статьях позволяет сначала взвешивать результаты согласно контентной фильтрации, а затем смещать эти веса по направлению к коллаборативной фильтрации (по мере наполнения доступного набора данных по конкретному пользователю).В контексте видеоматериала и использования постеров процесс получения дополнительной информации заключается в обработке и извлечении метаинформации, скрытых источников информации, невидимых на первый взгляд, из постеров с помощью нейронных сверточных сетей и использование данной информации для улучшения рекомендаций. Полученные из постера тэги (элементы метаинформации) не несут никакого семантического значения вне определенного контекста и не могут быть использованы для объяснения пользователю, почему ему рекомендуется тот или иной материал.

Несколько исследований пытались совместить семейство алгоритмов на основе контента и подходы коллаборативной (совместной) фильтрации[17]. Основная идея подобных гибридных методов состоит в том, чтобы объединить различные модели рекомендаций, чтобы преодолеть их ограничения и построить на их основе более надежные модели с более высокой прогнозирующей способностью. Существующие методики могут быть названы по-разному: смешивание[19,20], наложение или обобщенные методологии ансамбля, использующиеся в машинном обучении. Эти методы используют данные различных моделей в качестве аргументов для функций высшего порядка моделей более высокого уровня[18]. Подобная модель высокого уровня обычно является линейной моделью[16].

$$P(u,i) = w_1 P_1(u,i) + w_2 P_2(u,i) + \ldots + w_n P_n(u,i)$$

Веса обучаются на выборке. Как правило, для этого используется логистическая регрессия.

Stacking в общем виде:

$$P(u,i) = f_1(u,i)P_1(u,i) + f_2(u,i)P_2(u,i) + \ldots + f_n(u,i)P_n(u,i)$$

Такие гибридные методы сыграли важную роль в достижении наиболее производительных результатов в соревнованиях Netflix Prize[17, 21].

Подход, описанный в данной работе, базируется на идеях смешанных моделей, однако результирующая модель не является линейной. Классическую систему рекомендаций по совместной фильтрации предлагается использовать совместно с резервной моделью, которая компенсирует ошибку прогнозирования для трудно предсказуемых точек данных, то есть элементов с небольшим количеством оценок.

III. ИССЛЕДУЕМЫЙ ПОДХОД

Предположим, что имеется типичная настройка (вводные данные) для совместной (коллаборативной) фильтрации: у нас есть доступ к $n \times m$ матрице оценок R, содержащей оценки n пользователей по m элементам, которые могут быть как в разряде манге, так и в аниме: r_{ij} представляет рейтинг, который *i*-й пользователь дал элементу j. На практике,

поскольку пользователи оценивают только те немногие элементы, с которыми они взаимодействовали, рейтинговая матрица R имеет тенденцию быть очень разреженной: в наборе данных, рассмотренном в этой статье, известно менее 1% записей; другие популярные наборы данных в этой области сообщают о схожих уровнях разреженности. Поэтому сложно сделать вывод о недостающих записях R (такими записями как раз являются малоизвестные аниме/манга).

Еще одним предположением является предположение о том, что вся матрица рейтинга R может быть представлена как несколько скрытых профилей, то есть каждый вектор пользовательского рейтинга может быть разложен на комбинацию нескольких скрытых векторов. Скрытый вектор или скрытый профиль называется так, потому что данный вектор является неизвестным, определяемым по каким-то неявным критериям. Дополнить недостающие значения матрицы R можно путем матричной факторизации: попытки представить исходную матрицу в виде произведения двух матриц, где U – матрица размера n x r представляет пользователя (векторы потенциальных пользовательских оценок), а матрица V размера m x r представляет соответственно материал. Когда данная модель обучена, то есть когда известные значения матрицы R соответствуют аналогичным значениям в матрице UV^T. Тогда поиск недостающего значения (I, j) матрицы R просто выполняется путем поиска соответствующего значения в матрице UV^T.

Наконец, также предполагается, что при исследовании есть доступ к постерам некоторых объектов (аниме, манги). Это весь контент, который имеется для данной модели.

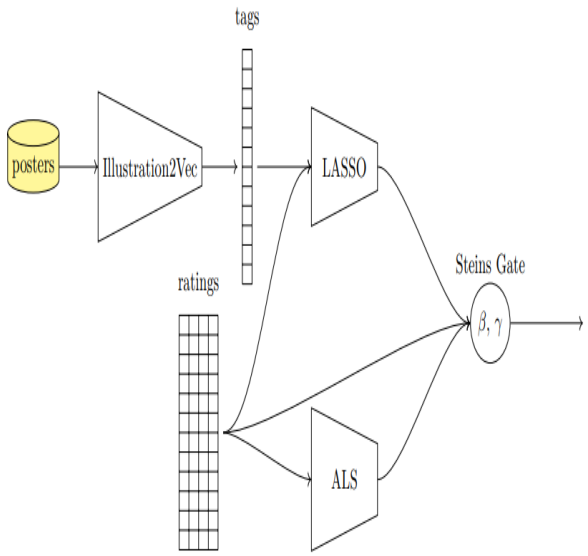


Рис. 1 – Архитектура BALSE

Рассмотрим теперь усовершенствованный алгоритм метода наименьших квадратов. Основная идея заключается в использовании матрицы рейтинга тогда, когда это возможно и на постеры к аниме или манге, когда информации в матрице рейтинга практически нет. Предполагается нелинейное связывание двух моделей для достижения большей точности и производительности. Рассмотрим структуру разработанной модели:

- Illustration2Vec блок или блок трансформации изображения в вектор, который является сверточной нейронной сетью, которая принимает на вход постер и на выходе производит вероятностную оценку потенциальных тегов – в данном случае тегами являются ключевые слова, характеризующие мангу или аниме
- ALS блок или блок матричной факторизации, который производит факторизацию матрицы рейтинга для коллаборативной фильтрации, используя при этом алгоритм ALS – Alternating Least Squares – чередование наименьших квадратов с λ взвешенной регуляризацией
- LASSO (least absolute shrinkage and selection operator) блок, отвечающий за выполнение регуляризованной линейной регрессии для каждой строки матрицы рейтинга, используя стохастические значения тэгов для вывода пользовательских предпочтений
- Steins Gate – блок, отвечающий за смесь выходных результатов двух моделей, позволяющий преодолеть их ограничения и рассчитать финальное рейтинговое значение Архитектура модели изображена на рисунке 1.

Рассмотрим немного подробнее строение каждого отдельного блока. Начать стоит с первого блока, блока преобразования изображения в вектор, сверточной нейронной сети. Данный блок получает информацию о тэгах из изображения[15]. Каждый тэг характеризуется своим весовым вероятностным коэффициентом, который говорит о веряотности нахождения конкретного тэга в некотором посте. Формально из постера необходимо получить матрицу T размером m x t, где m – количество элементов, а t – вероятность появления тэга в изображении. Все вычисления данного блока производятся с использованием сверточной нейронной сети VGG-16, которая позволяет предсказать большое количество тэгов для различных иллюстраций. Предварительное обучение данная нейронная сеть прошла с использованием датачетов ImageNet, а также была дополнительно обучена пользователями портала Danbooru, которые соотносили конкретные тэги с различными произведениями аниме/манги. Данная реализация свободна в использовании и была применена в рамках работы как наиболее подходящая. Выходным результатом работы данного блока является строка матрицы T, что можно обозначить таким понятием как «предсказание тэга» или вероятностная оценка появления тэга в произведении. На рисунке 2 представлен результат работы блока Illustration2Vec.

Далее, стоит рассмотреть работу следующего блока, блока LASSO. Данный блок аппроксимирует матрицу рейтинга R относительно регуляризованной линейной регрессионной моделью, называемой LASSO. Данную модель необходимо обучать независимо для каждого пользователя в обучающем множестве. $R \approx PT^T$, где P – параметры для обучения, а T – заданная матрица вероятностной оценки появления тэгов для каждого элемента.

LASSO идет вместе с параметром α, который вызывает L1 термин регуляризации, чтобы предотвратить переоснащение и дать объяснение предпочтений пользователя, как мы покажем позже. Следовательно, для каждого пользователя i из набора обучения, мы оцениваем параметры Pi для минимизации:

$$\frac{1}{2N_i}\|R_i - P_i T^T\|_2^2 + \alpha\|P_i\|_1$$

,

Где Ni число тайтлов, оцененных пользователем i.

Выходные параметры блока LASSO являются прогнозом рейтинга для каждой пары (i,j):

$$\hat{r}_{ij}^{LASSO} = \tau(P_i^T T_j).$$

,

где τ: x 7 → max (min (x, 2), −2) - это функция, которая сокращает свой вход до значений между -2 и 2. Такая функция предотвращает предоставление регрессором неверных прогнозов, выходящих за пределы диапазона значений рейтинга.

ALS блок производит матричную факторизацию матрицы рейтинга R.

$R \approx UV^T$ - где U, матрица пользовательских неявных векторов, а V – матрица неявных векторов относительно заданных элементов. Регуляризация необходима для предотвращения переобучения, на дальнейших шагах алгоритма происходит минимизация функции потерь.

Поэтому, поскольку мы хотим минимизировать квадратичную ошибку, минимизируемая функция потерь имеет следующий вид:

$$\sum_{i,j|r_{ij}\neq 0} (r_{ij} - U_i^T V_j)^2 + \lambda \left(\|U_i\|_2^2 + \|V_j\|_2^2 \right)$$

,

где Ui для каждого i = 1,. , , , n - строки U и Vj для каждого j = 1,. , , , m - строки V, а λ - параметр регуляризации. Эта оценка производится с использованием альтернативных наименьших квадратов с взвешенной λ-регуляризацией (ALS-WR) (Y. Zhou et al., 2008)[16].

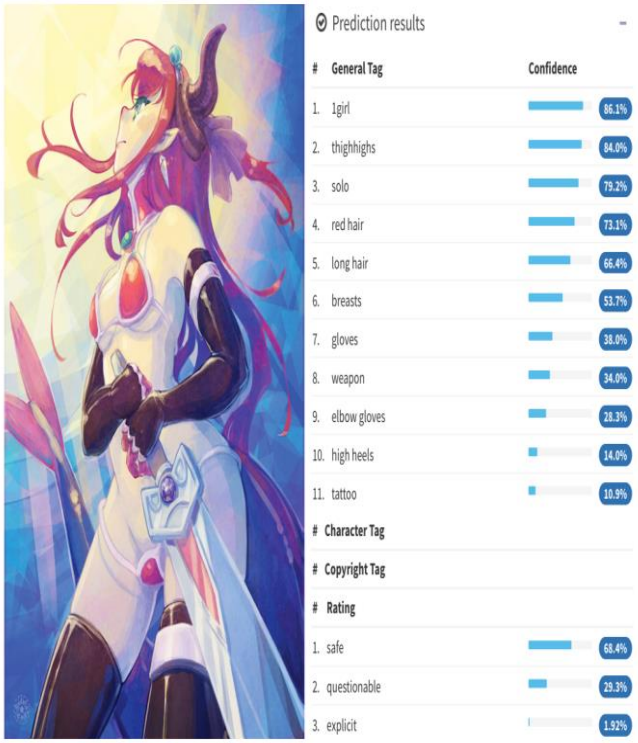
После того, как параметры были изучены, прогноз для оценки пользователя i на элементе j:

$$\hat{r}_{ij}^{ALS} = U_i^T V_j.$$

Steins Gate – блок совмещения выходных результатов двух обученных моделей с блоков LASSO и ALS. Основной целью данного блока является улучшения предсказательной способности результирующей модели для чего существует специальный самообучающийся алгоритм, позволяющий выбрать наиболее правильную модель (с наилучшей предсказательной способностью) в зависимости от количество известных значений в матрице рейтинга.

IV. ЗАКЛЮЧЕНИЕ

Мы предложили BALSE, модель для рекомендации аниме и манги, которая использует информацию, которая автоматически извлекается из постеров. Мы показали, что наша модель работает лучше, чем базовые модели, особенно в сценарии холодного запуска. Эта статья является подтверждением концепции, и полученные результаты очень обнадеживают. Steins gatea таков, что любое улучшение, сделанное в любом блоке, улучшит общую производительность подхода.. В дальнейшем планируется доработка и оптимизация разработанного приложения. Используя BALSE, рекомендательные системы могут автоматически пополнять свою базу данных, где новые элементы проходят трек прогнозирования тегов и модель, чтобы оправдать рекомендации для своих первых пользователей, и автоматически переходят на основной трек, когда набраны достаточные рейтинги.



СПИСОК ЛИТЕРАТУРЫ

1. Aggarwal, Charu C (2016). Recommender systems. Springer
2. Kula, Maciej (2015). “Metadata embeddings for user and item cold-start recommendations”. In: arXiv preprint arXiv:1507.08439 .
3. Alexandridis, Georgios, Georgios Siolas, and Andreas Stafylopatis (2017). “ParVecMF: A Paragraph Vector-based Matrix Factorization Recommender System”. In: arXiv preprint arXiv:1706.07513
4. Fang, Yi and Luo Si (2011). “Matrix co-factorization for recommendation with rich side information and implicit feedback”. In: Proceedings of the 2nd International Workshopon Information Heterogeneity and Fusion in Recommender Systems. ACM, pp. 65–69
5. Nedelec, Thomas, Elena Smirnova, and Flavian Vasile (2017). “Specializing Joint Representations for the task of Product Recommendation”. In: arXiv preprint arXiv:1706.07625 Introduction to word embedding and Word2Vec, Dhruvil Karani, 2013
6. Kim, Yong-Deok and Seungjin Choi (2014). “Scalable variational bayesian matrix factorization with side information”. In: Artificial Intelligence and Statistics, pp. 493–502
7. Xu, Miao, Rong Jin, and Zhi-Hua Zhou (2013). “Speedup matrix completion with side information: Application to multi-label learning”. In: Advances in Neural Information Processing Systems, pp. 2301–2309
8. Covington, Paul, Jay Adams, and Emre Sargin (2016). “Deep neural networks for youtube recommendations”. In: Proceedings of the 10th ACM Conference on Recommender Systems. ACM, pp. 191–198

9. Van den Oord, Aaron, Sander Dieleman, and Benjamin Schrauwen (2013). “Deep content-based music recommendation”. In: Advances in Neural Information Processing Systems, pp. 2643–2651
10. Wei, Jian, Jianhua He, Kai Chen, Yi Zhou, and Zuoyin Tang (2017). “Collaborative filtering and deep learning based recommendation system for cold start items”. In: Expert Systems with Applications 69, pp. 29–39
11. Biswas, Sampoorna, Laks VS Lakshmanan, and Senjuti Basu Ray (2017). “Combating the Cold Start User Problem in Model Based Collaborative Filtering”. In: arXiv preprint arXiv:1703.00397
12. Bobadilla, JesuS, Fernando Ortega, Antonio Hernando, and ´
13. JesuS Bernal (2012). “A collaborative filtering approach to ´
14. mitigate the new user cold start problem”. In: KnowledgeBased Systems 26, pp. 225–238
15. Tibshirani, Robert (1996). “Regression shrinkage and selectionvia the lasso”. In: Journal of the Royal Statistical Society Series B (Methodological), pp. 267–288
16. Zhou, Yunhong, Dennis Wilkinson, Robert Schreiber, and Rong Pan (2008). “Large-scale parallel collaborative filtering for the netflix prize”. In: Lecture Notes in Computer Science 5034, pp. 337–348.
17. Burke, Robin (2002). “Hybrid recommender systems: Survey and experiments”. In: User modeling and user-adapted interaction 12.4, pp. 331–370
18. Sill, Joseph, Gabor Tak ´ acs, Lester Mackey, and David ´ Lin (2009). “Feature-weighted linear stacking”. In: arXiv preprint arXiv:0911.0460
19. Roda, Fabio, Alberto Costa, and Leo Liberti (2011). “Optimal recommender systems blending”. In: Proceedings of the International Conference on Web Intelligence, Mining and Semantics. ACM, p. 60
20. Jahrer, Michael, Andreas Toscher, and Robert Legenstein (2010). “Combining predictions for accurate recommender systems”. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp. 693–702
21. Koren, Yehuda (2009). “The bellkor solution to the Netflix grand prize”. In: Netflix prize documentation 81, pp. 1–10

A hybrid convolutional and recurrent network approach for spoken language understanding

Moath Al Ali
Institute of Computer Science and Technology
Peter the Great St.Petersburg Polytechnic University (SPbPU)
Saint Petersburg, Russia
moath.alali.94@gmail.com

***Abstract*—In this paper, we are focusing on one of the most famous NLP problems which is slot filling to approach the state-of-the-art results on the ticketing problem to make the Spoken Dialogue systems work more efficiently. We propose a hybrid architecture, as a combination of a Recurrent Neural Network and a Convolutional Neural Network models, for Slot Filling in Spoken Language Understanding. Our network model is built from stacked units of 1-dimensional CNN across the temporal domain, which are used to train an RNN layer to model dependencies in the temporal domain. Experimental tests show extensive comparisons between different models for NER (Named Entities Recognition). Results demonstrate the effectiveness of hybrid models that combine benefits from both RNN and CNN architecture compared over distinct RNN and CNN models and also compared with other traditional models. Experimental results show that our model achieves F1-score of 95.11 on benchmark ATIS dataset.**

Keywords—SLU, slot-filling, Hybrid CNN and RNN, Deep learning

I. INTRODUCTION

The methodological revolution in spoken language research had been started about 20 years ago when the machine learning algorithms started to take place in the programmer society. However, the last five years brought the real change after the new deep learning architectures, which led to a new level of solutions and the Spoken Dialogue Systems (SDS) is one of the fields which had really improved recently.

Spoken language understanding (SLU) is a key component of SDS pipeline. SLU is responsible for extracting the semantic of users' utterances or queries. SLU can be divided into three main tasks: Domain classification, intent detection and slot filing [1]. These three tasks aim to capture the semantic in users' utterances, where domain classification task is to identify the domain of the utterance (e.g., movies, flights, etc.), while intent detection task is to figure out the user goal of the utterance (e.g., find movie, find flight, etc.) and the slot filling task is to label each word of the sentence as some parameter of the intent, for example, if the intents is "find flight", then the parameters can be (e.g., origin, destination, date, etc.). An example semantic frame for a movie-related utterance, "find recent comedies by James Cameron", is shown in Fig. 1.

The main three tasks of SLU usually are treated separately, so slot filling can be considered as a sequence labeling problem, which aims to map the input sequence $X = (x_1, \dots, x_T)$, to the corresponding slot labels sequence $Y = (y_1^S, \dots, y_T^S)$, while domain classification and intent detection can be treated as classification problems where the goal is to determine the intent and domain labels y^I *and* y^D. Popular

W	find	recent	comedies	by	james	cameron
	↓	↓	↓	↓	↓	↓
S	O	B-date	B-genre	O	B-dir	I-dir
D	movies					
I	find_movie					

Fig. 1. An example utterance with annotations of semantic slots in IOB format (S), domain (D), and intent (I), B-dir and I-dir denote the director name

approaches for domain classification and intent detection are classifiers like support vector machines (SVMs) [2] and deep belief neural network methods [3]. Approaches for slot filling include maximum entropy Markov models (MEMMs) [4], conditional random fields (CRF) [5] and recurrent neural network (RNN) [6] [7] [8]. But joint models for slot filling and intent detection are used in [9] [10] [11] [12][13]. Joint models simplify the problem of SLU, because on model needs to be trained and fine-tuned and if they are well designed, they can figure out the relationship between intents and slots.

This work focuses on the slot-filling part by building a model that extracts information from text in a reliable way. Before the era of deep learning the task of Named Entity Recognition (NER) was solved using grammars-based models and rule-based approaches, these models have proven to achieve good results in terms of precision but fail to capture all human-text varieties and thus the recall will be bad. Probabilistic approaches came with models built on hidden Markov model (HMM), which were state-of-art for many years and achieved an impressive achievement. With the recent revolution, many deep learning methods has replaced traditional previous ones and pushed state-of-art for these tasks in. Recurrent Neural Networks (RNN) models have replaced models based on HMM, that is RNN achieved the same task in a simpler way and deep RNNs are able to capture complex representations for the input. The problem with such models was that they need to handle the input token-by-token in sequence. Therefore, such structures could not be parallelized and the models will be slow to train and inference if the neural network structure is deep. Convolution Neural Networks (CNN) added a way to extract relations between tokens by mixing them in a way similar to extracting n-grams in the traditional NLP tasks. Such architectures that contain CNN could be optimized by parallelization so adding a convolutional layer could reduce the complexity and control the size of the neural network. In this paper, we discuss different approaches to solve slot filling for ticketing task as a NER problem, and showed different architectures that contain distinct RNN, CNN or hybrid architectures ones. We conducted many experiments with different values of the hyper-parameters and different optimization methods.

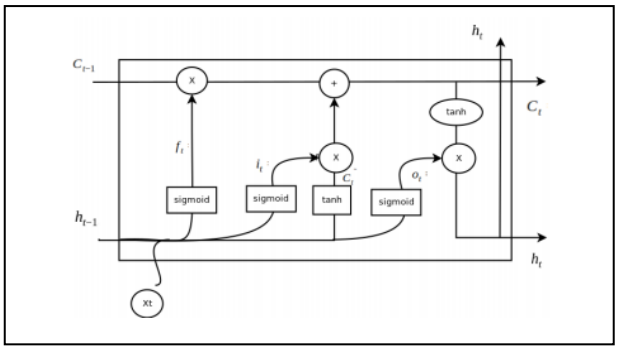


Fig. 2. LSTM unit.

II. RELATED WORK

Rule-based approaches are done manually, at first you all needed roles should be written need to achieve the goal, this operation is time-consuming and therefore not so efficient, it will be notable that the recall does not have not a good value, because it is so difficult to write all the varieties, but the positives of ruled-based approaches the precision will be quite high [14]. The most widely used formal system for modeling constituent structure in English and other natural languages is the Context-Free Grammar or CFG. A context-free grammar consists of a set of rules or productions, each of which expresses the ways that symbols of the language can be grouped and ordered together, and a lexicon of words and symbols [15]. In machine learning methods, we need a dataset of text with markup, in this dataset, each word should be assigned to a tag, this problem is known as slots filling problem. The first which we should do is making some Feature engineering, for example, see whether the word is capitalized or it is a name of a city, some cities consists of two words, maybe you check the previous or the next words (context). Probabilistic modeling and Conditional Random Field not only assume that features are dependent on each other but also considers the future observations while learning a pattern [16]. This combines the best of both HMM and MEMM. In terms of performance, it is considered previously to be the best method for entity recognition problem. Another paper studied the comprehensive investigations of RNNs for the task of slot filling in SLU. They implemented and compared several RNN architectures, including the Elman-type and Jordan-type networks with their variants [17].

III. DEEP LEARNING METHODS

A. *Recurrent Neural Network RNN*

Recurrent Neural Networks Fig. 2 are used for sequence modeling, it accepts input x_t at time step t and a hidden state h_t and use this hidden state to produce output y_t, and this hidden state will be passed to the next time step. So, we can think of the hidden state as a summary of the previous inputs to the neural network, we use activation function such as tanh or ReLU to calculate hidden state. Output y_t is the prediction of the next tag, it would be a vector of probabilities across our vocabulary, the following formulas explain the general form of RNN:

$$h_t = f(Ux_t + W\,h_{t-1}) \quad (1)$$

$$y_t = softmax(V\,h_t) \quad (2)$$

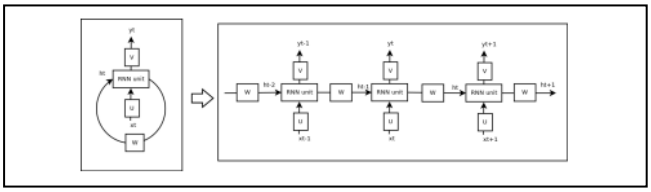


Fig. 3. General form of Recurrent Neural Network.

Long Short Term Memory (LSTM) and Gated Recurrent Units (GRU) are used as RNN units, these units can capture long term dependency. The LSTM does have the ability to remove or add information to the cell state, carefully regulated by structures called gates. Gates are a way to optionally let information through. They are composed out of a sigmoid neural net layer and a pointwise multiplication operation [18]. The sigmoid layer outputs numbers between zero and one, describing how much of each component should be let through. A value of zero means let nothing through, while a value of one means let everything through! An LSTM has three of these gates Fig. 3, to protect and control the cell state. The following formulas explain how does LSTM cell work:

$$f_t = \sigma(W_f\,[h_{h-1}, x_t] + b_f\,) \quad (3)$$

$$i_t = \sigma(W_i\,[h_{t-1}, x_t] + b_i) \quad (4)$$

$$C_t = tanh(W_c[h_{t-1}, x_t] + b_c) \quad (5)$$

$$C_t = f_t.C_{t-1} + i_t.C_t \quad (6)$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad (7)$$

$$h_t = o_t.tanh(C_t) \quad (8)$$

GRU has a simpler design Fig. 4 it was introduced by Cho, et al. (2014) [19], The key difference between a GRU and an LSTM is that a GRU has two gates (reset and update gates) whereas an LSTM has three gates (namely input, output and forget gates) [20]. The GRU unit controls the flow of information like the LSTM unit, but without having to use a memory unit. It just exposes the full hidden content without any control. GRU is relatively new but computationally more efficient. The following formulas describe the GRU mechanism:

$$z_t = \sigma(W_{z}.[h_{t-1}, x_t]) \quad (9)$$

$$r_t = \sigma(W_{r}.[h_{t-1}, x_t]) \quad (10)$$

$$\ddot{h}_t = tanh(W.[r.h_{t-1}, x_t]) \quad (11)$$

$$h_t = (1 - z_t).h_{t-1} + z_t * h_t \quad (12)$$

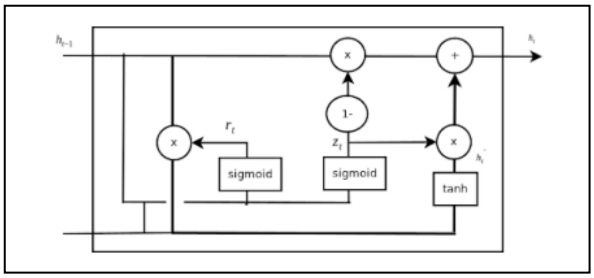


Fig. 4. GRU unit.

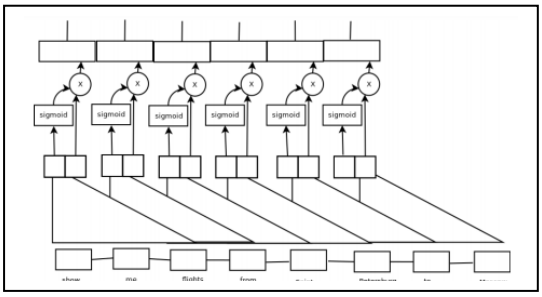


Fig. 5. Convolution Neural Network for sequences.

In our experiments, we used both GRU and LSTM units and compared between them. Other sequence architectures like Encoder-decoder architecture could be used to solve this task, at first the whole input will be encoded into hidden representation (encoder), and then this hidden representation is used to produce sequence of tags (decode). Some architectures use attention mechanism to give attention to parts of the input sequence and use these information to produce the output token.

B. Convolution Neural Network for sequences

RNNs operate sequentially, the output for the second input depends on the first one and so we can cannot parallelize an RNN. Convolutions have no such problem, each patch a convolutional kernel operates on is independent of the other, meaning that we can go over the entire input layer concurrently. Convolutions grow a larger receptive bled as we stack more and more layers. That means that by default, each step in the convolutions representation views all of the input in its receptive field, from before and after it Fig. 5. In our experiments we used 1D convolution to mix the tokens and extract relations between the consequence tokens, it is equivalent to n-gram relation where n is the size of the used filter, for example: if we care about the last 3 tokens we use filter size 3. Using CNN will result in some benefits, it runs faster than RNN and beats RNN in some tasks. If we divide convolution output into two parts one through sigmoid and linear and join them (element-wise multiplication), We get gated linear unit. Here we increased receptive field as it is shown in the following formula:

$$h_t(x) = (X.W + b) \oplus (X.V + c) \qquad (13)$$

C. Hybrid model CNN RNN

This model combines the benefits of both CNN and RNN, where RNN helps to capture the dependencies between tokens in the users query, using LSTM or GRU units will have resulted in a model that captures long-range dependencies between tokens using memory cell in their architecture. CNN will help with mixing the consequence tokens and extract relations between them [21]. In the task of slot filling, the hybrid architecture contains several convolution layers stacked with the same padding and the output of these layers will be the input for RNN layers Fig. 6, we can also stack several RNN layers. After these RNN layers, there will be a dense layer with SoftMax activations, this layer represents the output of the network.

IV. EXPERIMENTS

A. Dataset

ATIS (Airline Travel Information System) corpus (Tur et al., 2010) is one of the main data resources used in many studies over the past two decades for SLU research in spoken dialog systems e.g. [22] [5] [23]. Two primary tasks in SLU are intent determination (ID) and slot filling (SF). The dataset contains audio recordings of people making flight reservations. The training set contains 4,478 utterances and the test set contains 893 utterances. We use another 500 utterances for development set. There are 120 slot labels and 21 intent types in the training set [11]. The IOB format (inside, outside, beginning) is a common tagging format for tagging tokens in a chunking task in computational linguistics, The B- prefix before a tag indicates that the tag is the beginning of a chunk, and an I- prefix before a tag indicates that the tag is inside a chunk. The B- tag is used only when a tag is followed by a tag of the same type without O tokens between them. An O tag indicates that a token belongs to no chunk.

TABLE V. ATIS DATASET EXAMPLE

Sentence	**show**	**me**	**flights**	**from**	**Moscow**	**to**	**London**	**Today**
Slots/Concepts	O	O	O	O	B-fromLoc	O	I-toLoc	B-departDate
Named Entity	O	O	O	O	S-city	O	I-city	O
Intent	Find Flight							
Domain	Airline Travel							

The Table. I shows an example in the ATIS dataset, with the annotation of slot/concept, named entity, intent as well as domain. The latter two annotations are for the other two tasks in SLU: domain detection and intent determination. We can see that the slot filling is quite similar to the NER task, following the IOB tagging representation, except for a more specific granularity.

B. Training Details:

In training, we compared between different models for NER (Named Entities Recognition) system, all the models were trained using 100 epochs. We tuned our models using different dropout values (0.1, 0.25, 0.5) and we used different optimization methods (ADAM, RMSProb, SGD). For the embedding layer, we represent each token by a vector of size 100, and for our choice for the convolution layer we used 64 filters of size 5 and used ReLU as an activation function. The hidden size of the GRU/LSTM unit is 100 Fig. 7. Our architecture will go as following, input layer which is a sequence of tokens represented by indices using bag of words, embedding layer will represent each token with a vector, the vector size is a hyperparameter for the network, this embedding layer is followed by one of the main choices of the layers discussed above, recurrent neural network, convolutional neural network or a hybrid model which contains layer of CNN followed by layer of RNN.

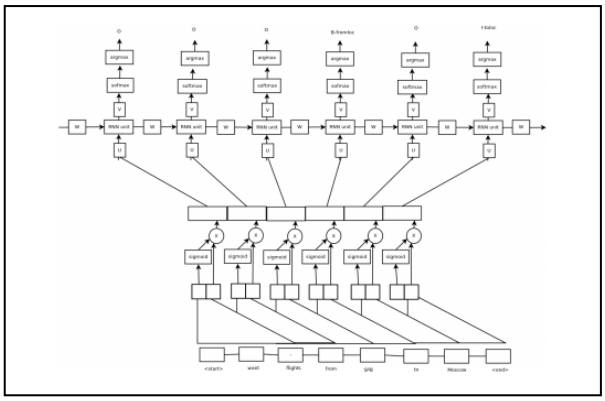


Fig. 6. Hybrid model CNN/RNN

C. Evaluation Metrics:

For evaluation, we computed precision, recall and F1 score for training and validation sets, and we picked the model with the best value of the F1 score. For Slot filling, the error rate can be computed in two ways: The more common metric is the F-measure using the slots as units. This metric is similar to what is being used for other sequence classification tasks in the natural language processing community, such as parsing and named entity extraction. In this technique, usually the IOB schema is adopted, where each of the words is tagged with their position in the slot: beginning (B), in (I) or other (O). Then, recall and precision values are computed for each of the slots. A slot is considered to be correct if its range and type are correct.

TABLE VI. COMPARISON BETWEEN DIFFERENT DEEP LEARNING STRUCTURES.

Structure description	Precision	Recall	F1-score	Average
Hybrid structure Convolution1D and RNN/GRU with dropout 0.25	94.47	95.61	95.04	94.89 0.15
Convolution1D structure with dropout 0.25	91.75	90.57	91.16	91.01 0.1
RNN/GRU structure with dropout 0.25	93.02	93.12	93.07	92.48 0.42

TABLE VII. COMPARISON BETWEEN HYBRID STRUCTURES BASED ON OPTIMIZATION MODEL.

Structure description	Precision	Recall	F1-score	Average
Hybrid structure Convolution1D and RNN/GRU with dropout 0.25; RMSProp	94.47	95.61	95.04	94.89 0.15
Hybrid structure Convolution1D and RNN/GRU with dropout 0.25; ADAM	94.71	94.95	94.83	94.67 0.23
Hybrid structure Convolution1D and RNN/GRU with dropout 0.25; SGD	94.22	94.65	94.44	94.23 0.16

D. Results

During evaluation process we focused on the difference between the use of different architectures of neural networks, we compared also between different optimization methods for the best neural network structure and at the end we included a comparison based on the type of recurrent unit used in the model.

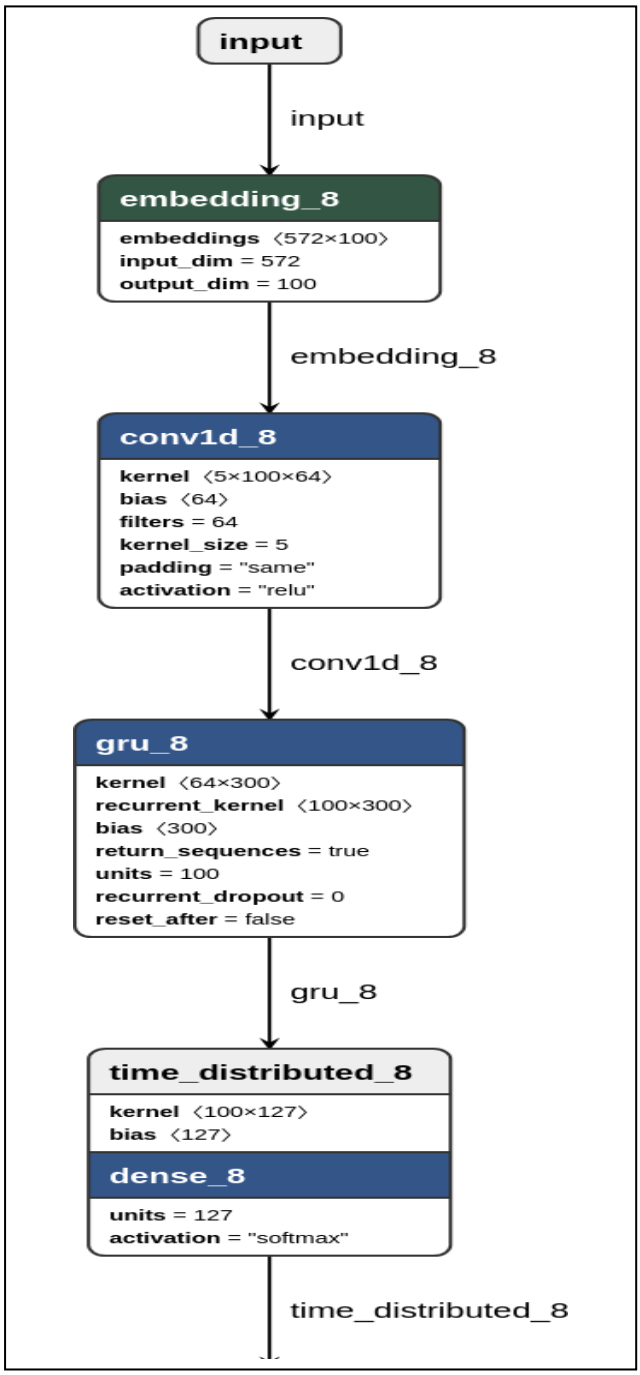


Fig. 7. Best Model Architecture, convolution layer with RNN layer of GRU units without dropout.

Our results show that hybrid architectures perform better than other pure RNN or pure CNN models Table. II, when we used dropout 0.25 and RMSProb optimization method , we got F1-score 95.04 for hybrid model compared with 91.16 for convolution model and 93.07 for recurrent model.

Our results show also that the use of RMSProb resulted in the best models according to f1-score metrics Table. III, under the same dropout 0.25 and hybrid model, we got F1-score equals to 95.04 for RMSProb compared with 94.83 when we used ADAM optimization model, and 94.44 when

we used SGD. Result show that the effect the Hybrid structure Convolution1D and RNN/GRU without dropout using RMSProp optimizer is giving the best F1-score 95.11 comparing with different levels of dropout on the same architecture Table. IV Based on the recurrent unit used in our experiments, GRU based

TABLE VIII. COMPARISON BETWEEN HYBRID STRUCTURES BASED ON THE USED RECURRENT UNIT.

Structure description	Precision	Recall	F1-score	Average
Hybrid structure Convolution1D and RNN/GRU with dropout 0.25; RMSProp	94.47	95.61	95.04	94.89 0.15
Hybrid structure Convolution1D and RNN/LSTM with dropout 0.25; RMSProp	94.36	95.17	94.76	94.40 .35

TABLE IX. COMPARISON BETWEEN DIFFERENT DEEP LEARNING STRUCTURES BASED ON DROPOUT VALUE.

Structure description	Precision	Recall	F1-score	Average
Hybrid structure Convolution1D and RNN/GRU without dropout; RMSProp	94.98	95.47	95.11	94.69 0.47
Hybrid structure Convolution1D and RNN/GRU with dropout 0.1; RMSProb	94.29	95.31	94.82	94.42 0.28
Hybrid structure Convolution1D and RNN/GRU with dropout 0.25; RMSProp	94.47	95.61	95.04	94.89 0.15
Hybrid structure Convolution1D and RNN/GRU with dropout 0.5; RMSProp	93.3	94.6	93.95	93.25 0.43

hybrid methods with f1-score 95.04 compared with LSTM based hybrid models with f1-score 94.67, GRU units improved the score by 0.37% Table. V. Our results show that the hybrid CNN/RNN-based models outperform Bi-dir. Jordan- RNN baseline by 1.13% on the ATIS benchmark Table. VI.

TABLE X. COMPARISON BETWEEN HYBRID STRUCTURES BASED ON THE USED RECURRENT UNIT.

Models	Precision	Recall	F1-score
Jordan-RNN [17]	92.76	93.87	93.31
Bi-dir. Jordan-RNN [17]	93.82	94.15	93.98
Hybrid structure (Our)	**94.98**	**95.47**	**95.11**

V. CONCLUSION

This paper addresses the problem of slot filling in Spoken Language Understanding. In particular, we focused on slot tagging without paying attention to the other intent classification part. We formulated our learning architecture as a hierarchy of spatial CNN features followed by the RNNs to model dependencies in the temporal domain. Experimental results on the ATIS dataset consistently demonstrated the effectiveness of the proposed approach. It is good to mention that combined models that solve the two tasks at the same time could be implemented and these models had proven to lead to better performance. But still, in the way to implement a full chatbot, we will need to generate human-like text in response to users input. In future work, we intend to explore the incorporation of attentional mechanism in our model, which could provide additional information to the slot label prediction.

REFERENCES

[1] G. Tur and R. De Mori, Spoken Language Understanding: Systems for Extracting Semantic Information from Speech. 2011.

[2] P. Haffner, G. Tur, and J. H. Wright, “Optimizing SVMs for complex call classification,” in ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, 2003.

[3] R. Sarikaya, G. E. Hinton, and B. Ramabhadran, “Deep belief nets for natural language call-routing,” in ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, 2011.

[4] A. McCallum, D. Freitag, and F. C. N. Pereira, “Maximum Entropy Markov Models for Information Extraction and Segmentation,” in Proceedings of the Seventeenth International Conference on Machine Learning, 2000.

[5] C. Raymond and G. Riccardi, “Generative and discriminative algorithms for Spoken Language Understanding,” in Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH, 2007.

[6] K. Yao, B. Peng, Y. Zhang, D. Yu, G. Zweig, and Y. Shi, “Spoken language understanding using long short-term memory neural networks,” in 2014 IEEE Workshop on Spoken Language Technology, SLT 2014 - Proceedings, 2014.

[7] G. Mesnil et al., “Using recurrent neural networks for slot filling in spoken language understanding,” IEEE Trans. Audio, Speech Lang. Process., 2015.

[8] B. Liu and I. Lane, “Recurrent Neural Network Structured Output Prediction for Spoken Language Understanding,” Proc. NIPS Work. Mach. Learn. Spok. Lang. Underst. Interact., 2015.

[9] P. Xu and R. Sarikaya, “Convolutional neural network based triangular CRF for joint intent detection and slot filling,” in 2013 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2013 - Proceedings, 2013.

[10] D. Z. Guo, G. Tur, W. T. Yih, and G. Zweig, “Joint semantic utterance classification and slot filling with recursive neural networks,” in 2014 IEEE Workshop on Spoken Language Technology, SLT 2014 - Proceedings, 2014.

[11] C.-W. Goo et al., “Slot-Gated Modeling for Joint Slot Filling and Intent Prediction,” 2018.

[12] D. Hakkani-Tür et al., “Multi-domain joint semantic frame parsing using bi-directional RNN-LSTM,” in Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH, 2016.

[13] B. Liu and I. Lane, “Attention-based recurrent neural network models for joint intent detection and slot filling,” in Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH, 2016.

[14] M. Surdeanu and H. Ji, “Overview of the English Slot Filling Track at the TAC2014 Knowledge Base Population Evaluation,” in Proceedings ofthe TAC- KBP 2014 Workshop, 2014.

[15] U. Schade and M. R. Hieb, “Formalizing Battle Management Language: A grammar for specifying orders,” in Spring Simulation Interoperability Workshop 2006, 2006.

[16] C. Sutton, A. McCallum, and K. Rohanimanesh, “Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data,” J. Mach. Learn. Res., 2007.

[17] G. Mesnil, X. He, L. Deng, and Y. Bengio, “Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding,” in Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH, 2013.

[18] B. Qu, X. Li, D. Tao, and X. Lu, “Deep semantic understanding of high resolution remote sensing image,” in IEEE CITS 2016 - 2016 International Conference on Computer, Information and Telecommunication Systems, 2016.

[19] K. P. Zhu et al., “Gated Recurrent Units Based Neural Network For Tool Condition Monitoring,” Int. J. Mach. Tools Manuf., 2017.

[20] G. Kurata, B. Zhou, B. Xiang, and M. Yu, “Leveraging sentence-level information with encoder LSTM for semantic slot filling,” in EMNLP 2016 - Conference on Empirical Methods in Natural Language Processing, Proceedings, 2016.

[21] S. T. Hsu, C. Moon, P. Jones, and N. F. Samatova, “A Hybrid CNN-RNN alignment model for phrase-aware sentence classification,” in 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017 - Proceedings of Conference, 2017.

[22] Y. He and S. Young, “A data-driven spoken language understanding system,” in 2003 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2003, 2003.

[23] G. Tur, D. Hakkani-Tür, and L. Heck, “What is left to be understood in atis?,” in 2010 IEEE Workshop on Spoken Language Technology, SLT 2010 - Proceedings, 2010.

Построение филогенетического дерева эволюции для задач, требующих больших объемов данных

Roman D. Chusov, Nikita V. Voinov
Peter the Great St.Petersburg Polytechnic University
St. Petersburg, Russia
Churomann@gmail.com, voinov@ics2.ecd.spbstu.ru

***Аннотация* — Рассматривается проблема построения филогенетического дерева для случаев, когда требуется обработать большой объем данных. Приводится описание уже существующих инструментов и алгоритмов, лежащих в их основе, а также их сравнение между собой. Модифицирован один из базовых алгоритмов, для поддержки распараллеленных кластерных вычислений. Реализован веб-интерфейс для взаимодействия с приложением. Приложение загружено на облачный хостинг и доступно в сети Интернет. При помощи данного приложения было вычислено дерево расстояний для двух сотен геномных цепочек.**

Ключевые слова* — *ФИЛОГЕНЕТИКА, BIG DATA, БИОИНФОРМАТИКА, ДЕРЕВО ЭВОЛЮЦИИ, CLOUD COMPUTING, NEIGHBOR JOINING, КЛАСТЕРНЫЕ ВЫЧИСЛЕНИЯ

I. Введение

У каждого живого организма есть набор генов или геномная последовательность. Данные, содержащиеся в геномной последовательности, содержат информацию об этом организме. В процессе эволюции организмы изменялись, изменялись их геномные последовательности. Было обнаружено, что у близких организмов могут быть схожие геномные последовательности [1]. Если научиться считать расстояния между таким последовательностями, то можно получить инструмент, позволяющий анализировать ход эволюции живых организмов по их геномам.

Филогения [2] - раздел биологии, изучающий родственные взаимоотношения разных групп живых организмов. Филогению отображается обычно в виде "эволюционных древ" [3] или систематических названий. Филогенетика или молекулярная филогенетика – это те же взаимоотношения, но на уровне отдельных белковых семейств. Автоматизация метода построения филогенетических деревьев позволяет проводить сравнение нескольких биологических объектов, а также реконструировать дерево эволюции для этих объектов.

В последние годы в повсеместно возрос интерес к быстрым облачным вычислениям, которые позволяют проводить распределенные манипуляции над большими объемами данных, в том числе и биологии [4]. В связи с этим было решено построить собственную платформу.

В данной статье рассмотрены существующие способы построения филогенетических деревьев, проведено краткое сравнение существующих алгоритмов построения филогенетических деревьев. В статье будет представлено описание реализованного и модифицированного для кластерных вычислений алгоритма Neighbor Joining [5], который лежит в основе также представленного в данной статье приложения. Основные требования к приложению – это должна быть облачная PaaS платформа, предоставляющая пользователю возможность проводить вычисления филогенетического дерева из браузера.

II. Обзор алгоритма

Филогенетическое дерево — дерево, отражающее эволюционные взаимосвязи между различными видами или другими сущностями, имеющими общего предка. Идея «дерева» появилась в ранних взглядах на жизнь, как на процесс развития от простых форм к сложным. Современные эволюционные биологи продолжают использовать деревья для иллюстрации эволюции, так как они наглядно показывают взаимосвязи между живыми организмами. Пример такого дерева представлен на Рисунке 1.

Рис. 1. Филогенетическое дерево

Данные об организмах как правило хранятся в файлах в формате “fasta", где символом «>» помечается информация о последовательности, а за ней следует последовательность букв, кодирующая белок или аминокислоту, как показано на Рисунке 2.

```
>gi|532319|pir|TVFV2E|TVFV2E envelope protein
ELRLRYCAPAGFALLKCNDADYDGFKTNCSNVSVVHCTNLMNTTVTTGLLLNGSYSENRT
QIWQKHRTSNDSALILLNKHYNLTVTCKRPGNKTVLPVTIMAGLVFHSQKYNLRLRQAWC
HFPSNWKGAWKEVKEEIVNLPKERYRGTNDPKRIFFQRQWGDPETANLWFNCHGEFFYCK
MDWFLNYLNNLTVDADHNECKNTSGTKSGNKRAPGPCVQRTYVACHIRSVIIWLETISKK
TYAPPREGHLECTSTVTGMTVELNYIPKNRTNVTLSPQIESIWAAELDRYKLVEITPIGF
APTEVRRYTGGHERQKRVPFVXXXXXXXXXXXXXXXXXXXXXXXVQSQHLLAGILQQQKNL
LAAVEAQQQMLKLTIWGVK
```

Рис. 2. Запись в fasta-файле

В задачах филогенетики как правило приходится сталкиваться с решением проблем, связанных с эволюцией символьных цепочек. Пример такой эволюции в цепочке представлен ниже на Рисунке 3.

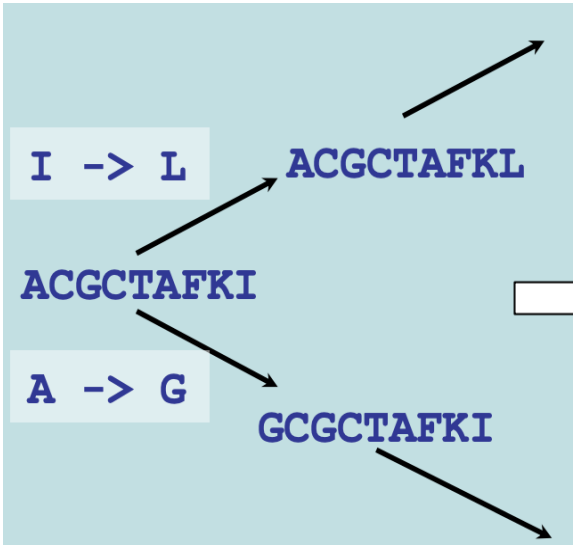


Рис. 3. Эволюция символьных цепочек

Эволюция цепочек может быть вычислена на методах, основывающихся на вычислении всех дистанций между цепочками, либо на символьных методах, основывающихся на анализе цепочек посимвольно. Покажем работу методов наглядно для набора геномов, представленного на Рисунке 4.

Taxa	Characters
Species A	ATGGCTATTCTTATAGTACG
Species B	ATCGCTAGTCTTATATTACA
Species C	TTCACTAGACCTGTGGTCCA
Species D	TTGACCAGACCTGTGGTCCG
Species E	TTGACCAGTTCTCTAGTTCG

Рис. 4. Тестовый набор геномов

Символьный метод подразумевает использование уже выровненных цепочек во время построения дерева. В качестве критерия оптимальности используется критерий «максимальной жадности» - использование минимального количества эволюционных событий (разветвлений). Метод представлен на Рисунках 5, 6. Объяснение данного подхода выходит за рамки данной статьи, подробный алгоритм работы данного метода можно найти в публикации “Constructing Phylogenetic Trees” (Regina Krisztina Bíró) [6].

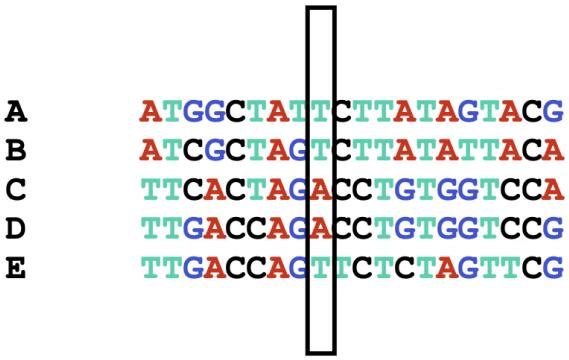


Рис. 5. Выровненные геномные цепочки

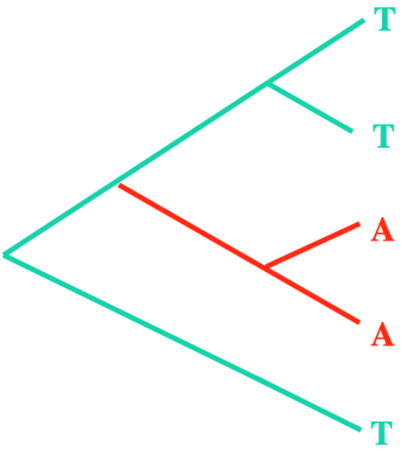


Рис. 6. Символьный метод построения дерева

Дистанционный метод требует построения новой матрицы, представленной, в которой попарно сравниваются все рассматриваемые последовательности, также представленные на Рисунке 7.

Таблица получается путем вычисления метрики расстояния попарно между каждым из геномов, представленных на входе. Этой метрикой может быть, например, расстояние Левенштейна или любое другое, показывающее корреляцию между символьными цепочками [7]. В данном примере для простоты ограничимся метрикой Левенштейна.

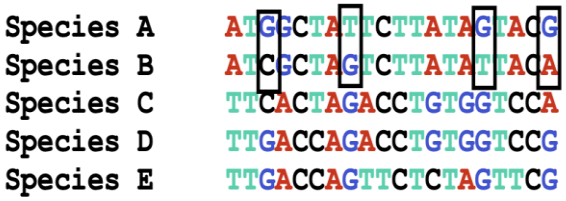


Рис. 7. Геномные цепочки для дистанционного метода

	A	B	C	D	E
Species A	----	0.20	0.50	0.45	0.40
Species B	0.23	----	0.40	0.55	0.50
Species C	0.87	0.59	----	0.15	0.40
Species D	0.73	1.12	0.17	----	0.25
Species E	0.59	0.89	0.61	0.31	----

Рис. 8. Матрица попарных расстояний

Полученная матрица представлена на Рисунке 8. В верхней части находятся результаты сравнения последовательностей при помощи вычисления расстояния Левенштейна, в нижней части находится скорректированное по формуле Джукеса-Кантора (1) это же расстояние.

$$K(A,B) = -\frac{3}{4}\ln\left(1-\frac{4}{3}D(A,B)\right) \quad (1)$$

После этого происходит объединение узлов в дерево при помощи метода ближайших соседей, как показано на Рисунке 9.

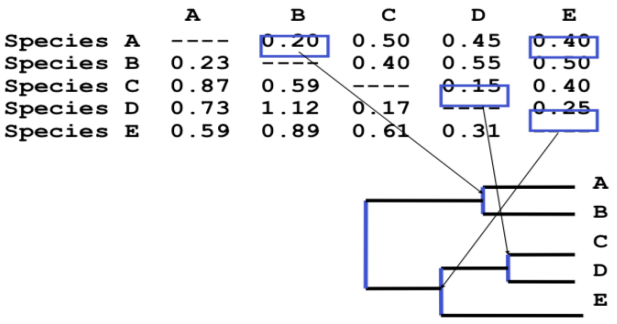


Рис. 9. Объединение узлов дерева

III. Обзор существующих решений

В ходе данной работы были изучены принципы работы уже существующих инструментов для задач построения филогенетических деревьев.

На рисунке 10 представлена сводная таблица для основных существующих инструментов.

Одним из самых точных является инструмент под названием FastTree [8]. Этот инструмент опубликован в открытом доступе и насчитывает порядка пяти тысяч строк кода. Реализован на языке программирования C с использованием библиотеки OpenMP [9] для распараллеливания вычислений. Если сравнивать с остальными инструментами из таблицы, то на нем можно вычислять филогенетическое дерево, состоящие из примерно семидесяти девяти тысяч входных последовательностей геномов с высокой точностью. Этот алгоритм, как и все остальные, построен на базе уже рассмотренного выше алгоритма Neighbor Joining.

Одной из самых слабых частей алгоритма Neighbor Joining является потребление памяти (O(N^2)), поскольку он требует вычисления попарных расстояний между каждым геномом.

Topological accuracy for simulated alignments with varying numbers of sequences

#Sequences	250	1,250	5,000	78,132
Type	a.a.	a.a	a.a.	nt
RAxML 7 (JTTCAT + SPRs)	90.5%	88.4%	88.4%	--
PhyML 3.0 (Γ_4 + SPRs)	89.9%	--	--	--
FastTree 2.0.0 (JTT+CAT or JC+CAT)	86.9%	83.7%	84.3%	92.1%
PhyML 3.0 (Γ_4, no SPR)	86.0%	--	--	--
PhyML 3.0 (no gamma, no SPR)	81.7%	80.1%	--	--
FastME 1.1 (log-corrected distances)	79.6%	77.7%	75.3%	--
BIONJ (max-lik. distances)	77.7%	73.7%	73.1%	--
Parsimony (RAxML)	76.8%	76.5%	69.4%	--
BIONJ (log-corrected distances)	76.6%	73.0%	72.3%	--
Neighbor-joining (log-corrected distances)	76.0%	72.6%	71.6%	66.1%
Clearcut 1.0.8 (log-corrected distances)	75.5%	72.3%	71.5%	58.1%

Рис. 10. Список основных инструментов для построения филогенетических деревьев

Этим и объясняется, что большинство инструментов не справляется с таким большим количеством вычислений в памяти.

В случае FastTree используется модифицированная версия алгоритма [10, 11], потребляющая O(N*La+N*sqrt(N)) памяти.

С точки зрения пользователя FastTree представляет собой консольную утилиту, которая принимает на вход файл в формате fasta, а на выходе отдает филогенетическое дерево в Newick-формате [12]. Для отрисовки дерева необходим отдельный инструмент, для этого часто используется MEGA [13].

IV. Реализация собственного инструмента

В ходе данной работы было решено разработать собственный инструмент, способный строить филогенетическое дерево на больших объемах данных. В разрабатываемом инструменте в настоящий момент для вычисления дерева используется наивная реализация алгоритма Neighbor Joining, представленная во второй главе. Однако, для преодоления ограничений по памяти, алгоритм реализован для запуска на кластере, что позволяет при необходимость горизонтально масштабировать вычислительную мощность – увеличивать количество вычислительных узлов, а не увеличивать вычислительную мощность, а, следовательно, и стоимость каждого узла, как это реализовано в остальных алгоритмах.

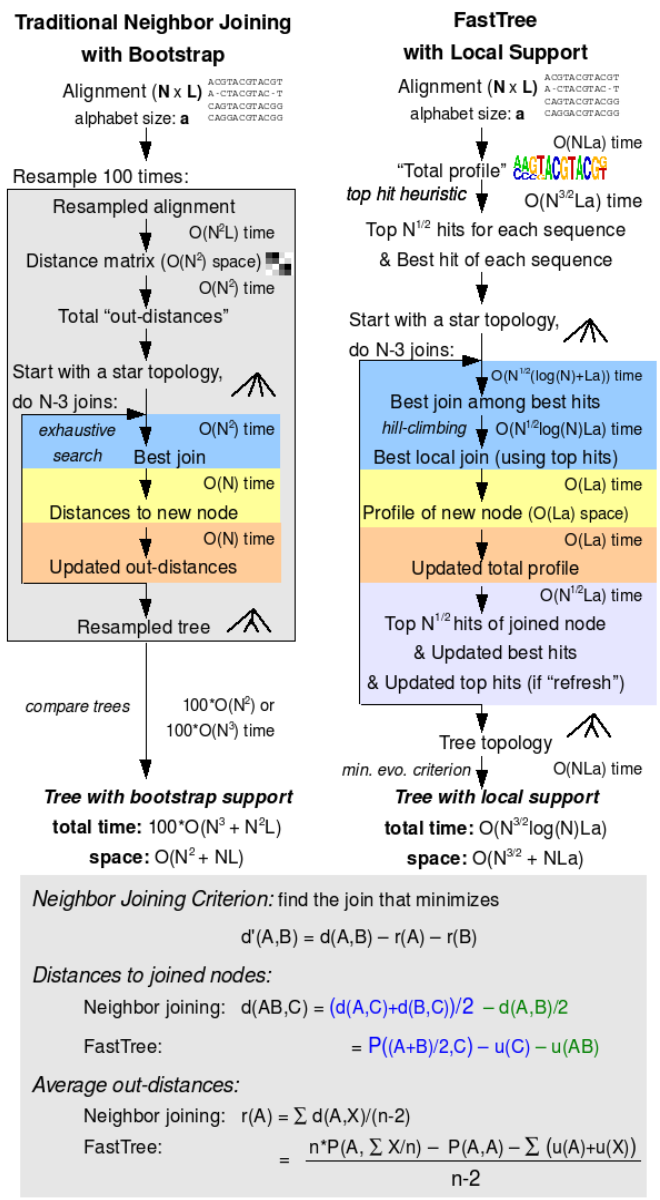


Рис. 11.Сравнение алгоритма FastTree и наивного Neighbor Joining

Инструмент на данный момент находится в разработке в стадии "Proof of Concept". К настоящему моменту был развернут кластер Apache Spark [14], реализован алгоритм построения матрицы расстояний, а также Neighbor Joining алгоритм. Также реализована простая веб-форма, позволяющая загружать данные для анализа. Приложение запущено на облачной платформе Heroku и доступно в сети Интернет.

На Рисунке 12 представлена посчитанная матрица расстояний для тестового примера алгоритма Neighbor Joining из второй главы.

taxon1	data1	taxon2	data2	distance	JC-disrance
A	ATGGCTATTCTTATAGTACG	B	ATCGCTAGTCTTATATTACA	0.20000000298023224	0.23261620029183264
A	ATGGCTATTCTTATAGTACG	C	TTCACTAGACCTGTGGTCCA	0.5	0.8239592165010822
A	ATGGCTATTCTTATAGTACG	D	TTGACCAGACCTGTGGTCCG	0.44999998807907104	0.6872180191032944
A	ATGGCTATTCTTATAGTACG	E	TTGACCAGTTCTCTAGTTCG	0.3499999940395355	0.47145648333909097
B	ATCGCTAGTCTTATATTACA	A	ATGGCTATTCTTATAGTACG	0.20000000298023224	0.23261620029183264
B	ATCGCTAGTCTTATATTACA	C	TTCACTAGACCTGTGGTCCA	0.40000000596046445	0.5716050518075965
B	ATCGCTAGTCTTATATTACA	D	TTGACCAGACCTGTGGTCCG	0.5500000119209929	0.9913169246902245
B	ATCGCTAGTCTTATATTACA	E	TTGACCAGTTCTCTAGTTCG	0.5	0.8239592165010822
C	TTCACTAGACCTGTGGTCCA	A	ATGGCTATTCTTATAGTACG	0.5	0.8239592165010822
C	TTCACTAGACCTGTGGTCCA	B	ATCGCTAGTCTTATATTACA	0.40000000596046445	0.5716050518075965
C	TTCACTAGACCTGTGGTCCA	D	TTGACCAGACCTGTGGTCCG	0.15000000596046448	0.16735767093623788
C	TTCACTAGACCTGTGGTCCA	E	TTGACCAGTTCTCTAGTTCG	0.40000000596046445	0.5716050518075965
D	TTGACCAGACCTGTGGTCCG	A	ATGGCTATTCTTATAGTACG	0.44999998807907104	0.6872180191032944
D	TTGACCAGACCTGTGGTCCG	B	ATCGCTAGTCTTATATTACA	0.5500000119209929	0.9913169246902245
D	TTGACCAGACCTGTGGTCCG	C	TTCACTAGACCTGTGGTCCA	0.15000000596046448	0.16735767093623788
D	TTGACCAGACCTGTGGTCCG	E	TTGACCAGTTCTCTAGTTCG	0.25	0.30409883108112323
E	TTGACCAGTTCTCTAGTTCG	A	ATGGCTATTCTTATAGTACG	0.3499999940395355	0.47145648333909097
E	TTGACCAGTTCTCTAGTTCG	B	ATCGCTAGTCTTATATTACA	0.5	0.8239592165010822
E	TTGACCAGTTCTCTAGTTCG	C	TTCACTAGACCTGTGGTCCA	0.40000000596046445	0.5716050518075965
E	TTGACCAGTTCTCTAGTTCG	D	TTGACCAGACCTGTGGTCCG	0.25	0.30409883108112323

Рис. 12.Матрица расстояний

Рис. 13.Веб-приложение

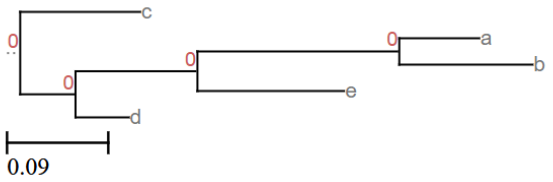


Рис. 14.Визуализация тестового дерева

На Рисунке 13 представлен веб интерфейс, доступный для пользователя в сети Интернет по ссылке [https://blooming-wave-47898.herokuapp.com].
Необходимо выбрать файл в формате fasta и загрузить его на сайт, после чего нажать "Submit". Приложение проведет расчеты и вернет дерево в формате Newick. Для визуализации воспользуемся веб-сервисом etetoolkit [15].

Стоит отметить, что входным параметром ошибки для каждого разработчика является средняя относительная погрешность оценки разработчиком заданий на прошлой итерации расчета модели. Чем ниже классифицируется точность оценки конкретного разработчика, тем ниже весовой коэффициент его оценки

в совокупной оценке стоимости следующей итерации проектной разработки.

V. ЗАКЛЮЧЕНИЕ

В данной работе была рассмотрена задача построения филогенетического дерева для случаев. Было приведено описание популярных инструментов, а также была приведена таблица сравнения этих инструментов между собой. Был рассмотрен базовый алгоритм объединения соседей, лежащий в основе большинства инструментов этой области. Была приведена собственная реализация этого алгоритма для кластерной инфраструктуры. Было реализовано веб-приложение, позволяющее пользователям взаимодействовать с программой в сети Интернет

В дальнейшем планируется модифицировать алгоритм программы – опробовать разные метрики расстояния между последовательностями, адаптировать алгоритм FastTree для распределенной работы в кластере. Также планируется провести сравнительный анализ собственной программы с уже существующими на одинаковых наборах больших данных.

СПИСОК ИСТОЧНИКОВ

1. Eduard Kellenberger (2003). “The evolution of molecular biology”. DOI: 10.1038/sj.embor.7400180 URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1299088/ - (дата ообращения 25.11.2019)
2. Douglas E. Soltis; Pamela S. Soltis (2003). “The Role of Phylogenetics in Comparative Genetics”. DOI: 10.1104/pp.103.022509 URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC526274/ - (дата обращения 29.11.2019)
3. T. Ryan Gregory (2008). “Understanding Evolutionary Trees”. URL: https://link.springer.com/article/10.1007/s12052-008-0035-x - (дата обращения 29.11.2019)
4. Lin Dai, Xin Gao, Yan Guo, Jingfa Xiao & Zhang Zhang (2012). “Bioinformatics clouds for big data manipulation”. URL: https://biologydirect.biomedcentral.com/articles/10.1186/1745-6150-7-43 - (дата обращения 29.11.2019)
5. Saitou N, Nei M. (1987). “The neighbor-joining method: a new method for reconstructing phylogenetic trees”. URL: https://www.ncbi.nlm.nih.gov/pubmed/3447015 - (дата обращения 29.11.2019)
6. Regina Krisztina Biro (2015). “Constructing Phylogenetic Trees”. URL: https://web.cs.elte.hu/blobs/diplomamunkak/bsc_alkmat/2015/biro_regina_krisztina.pdf - (дата обращения 01.12.2019)
7. Chunchun Zhao and Sartaj Sahni (2019). “String correction using the Damerau-Levenshtein distance”. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6551241/ - (дата обращения 01.12.2019)
8. Morgan N Price, Paramvir S Dehal and Adam P Arkin (2009). “FastTree: Neighbor-Joining with Profiles instead of a Distance Matrix”. URL: http://www.microbesonline.org/fasttree/FastTree-preprint.pdf - (дата обращения 01.12.2019)
9. Ms. Ashwini M. Bhugul (2017). “Parallel Computing using OpenMP”. URL: https://www.ijcsmc.com/docs/papers/February2017/V6I2201713.pdf - (дата обращения 01.12.2019)
10. Morgan N. Price, Paramvir S. Dehal, Adam P. Arkin (2009). “FastTree: Computing Large Minimum Evolution Trees with Profiles instead of a Distance Matrix”. URL: https://academic.oup.com/mbe/article/26/7/1641/1128976 - (дата обращения 01.12.2019)
11. Morgan N. Price, Paramvir S. Dehal, Adam P. Arkin (2010). “FastTree 2 – Approximately Maximum-Likelihood Trees for Large Alignments”. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0009490 - (дата обращения 03.12.2019)
12. “The Newick tree format”. // Официальный сайт Вашингтонского университета. URL: http://evolution.genetics.washington.edu/phylip/newicktree.html - (дата обращения 03.12.2019)
13. Kumar S, Stecher G, Tamura K (2016). “MEGA7: Molecular Evolutionary Genetics Analysis Version 7.0 for Bigger Datasets”. URL: https://www.ncbi.nlm.nih.gov/pubmed/27004904 - (дата обращения 12.12.2019)
14. Zhan Ye, Ahmad P. Tafti, Karen Y. He, Kai Wang and Max M. He (2016). “SparkText: Biomedical Text Mining on Big Data Framework”. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5042555/ - (дата обращения 12.12.2019)
15. Jaime Huerta-Cepas, François Serra and Peer Bork (2016). “ETE 3: Reconstruction, Analysis, and Visualization of Phylogenomic Data”. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4868116/

Wildfire detection using CNN

Diaby Oumar
Software development department
Innovation-Conception-ConseilsI2C-Mali
Bamako, Republic of Mali
oumardiaby@i2c-mali.com

***Abstract*—Indonesia, by the location of its geographic and geologic, it have more potential encounters for natural disasters. This nation is traversed by three tectonic plates, namely: Indo-Australian, the Eurasian and the Pacific plates. One of the tools employed to detect danger and send an early disaster warning is sensor device for ocean waves, but it has drawbacks related to the very limited time gap between information/warnings obtained and the real disaster event, which is only less than 30 minutes. Natural disaster early detection information system is essential to prevent potential danger. The system can make use of the pattern recognition of images that take place during the natural disaster.**

Keywords—natural disasters; CNN; images ; videos; wildfire; flooding, earthquakes.

I. INTRODUCTION

Natural disasters like hurricanes, wildfire, earthquake, flood, tsunamis, and volcanic eruption often have catastrophic effects on human lives and cause significant infrastructure harm. It is challenging to detect these disasters as quickly as possible before it can largely affect human and animal lives. Remote sensing applications are used for this purpose. Images captured from the satellite are assessed to identify any abnormal change in the atmosphere, which may lead to a natural disaster. However, these systems are still not fully automatic and require human effort to analyze satellite images properly. Use of human resources for this purpose makes this process time-taking and also human error-prone. Recently, a wildfire in Amazon rainforest got the attention of the world when NASA released a satellite image of the burning forest. The forest could be saved if we can detect wildfire at an early stage and take appropriate action to minimize losses.

Current advances in remote sensing applications yield significant success in the management of natural disasters. It still requires human effort and time to analyze images, which causes a delay in prediction. This challenge leads to a substantial loss of infrastructure and lives. Therefore, a fully automated system is the need of the hour, which can identify these natural disasters with minimum time delay. This system can prevent the loss of precious human lives and other resources.

In this study, we developed an automated calamity detection system using deep learning, which can predict disasters .

II. APPROACH

A. Methodology

This section first provides a preview of the CNN algorithm, and then elaborates the procedure of the susceptibility model development through a series of processes including data preprocessing and sample library generation, model architectures design and parameter adjustment, and performance evaluation. Finally, the detailed methodological flowchart is described.

B. Preview of Convolutional Neural Network

Convolutional neural network (CNN) is one of the most notable DL approaches and has exhibited robust performance in feature learning for image classification and recognition. It is a feed-forward neural network whose parameters are trained by using the classic stochastic gradient descent based on the backpropagation algorithm (Hu et al. 2015).

Generally, the CNN consists of several building blocks—convolutional, pooling, and fully connected layers (Yamashita et al. 2018). The different types of processing layers play different roles. The convolutional layers, which perform linear convolution operations between the input tensor and a set of filters, output the feature maps. Typically, each feature map is then followed by a nonlinear activation function. The rectified linear unit (ReLU), which performs the nonlinear transformation of the feature map generated by the convolution layer and introduces nonlinearity into the system, is the most commonly used activation function.

The purpose of the convolution operation is to extract different input layer features and achieve weight sharing. The input and output of each stage are sets of arrays called feature maps. For example, if the input is a 2-dimensional image □, the input is first decomposed into a sequential array □ = {□1, □2, …, □n}. The convolutional layer is defined as:

$$y_j = f\left(b_j + \sum_i k_{ij} * x_i\right)$$

where □□ denotes the □th output for the convolutional layer and □□ denotes each input feature map.

The pooling layers perform a subsampling operation to reduce the dimensions of the feature maps. According to the maximum and average functions, the pooling layer can be divided into the max-pooling and average-pooling layers. The fully connected layers, which are the flat feed-forward neural network layers, provide high-level abstraction features. They are often used at the end of the network architecture and create the final nonlinear combinations of features for making

the predictions by the network. The activation function for the last fully connected layer needs to be selected reasonably based on given tasks. The softmax or sigmoid function can be used to compute the posterior probability for each grid cell.

III. Data Preprocessing

First, appropriate forest fire influencing factors were selected, and variables raster datasets (VRDs) and ignition raster datasets (IRDs) were constructed through preprocessing. Then, the sample libraries were collected from the established IRDs and VRDs using an appropriate sampling method.

Finally, complete content and organizational editing before formatting. Please take note of the following items when proofreading spelling and grammar:

A. Forest Fire Influencing Factors

Four categories of forest fire influencing factors were considered, including topography-related, climate-related, vegetation-related, and human activities-related variables. ArcGIS 10.5 was employed for handling geographic data. The effect of the topography has been considered as a significant feature in forest fire assessment (Renard et al. 2012; Adab et al. 2013). Three topography-related influencing factors—elevation, slope, and aspect—were retrieved from the digital elevation model (DEM). The DEM was produced by the Advanced Spaceborne Thermal Emission and Reflection Radiometer (ASTER) GDEM version 2 (with a 30 m pixel size). Surface roughness was obtained from the Climate Forecast System Reanalysis (CFSR). The climatic characteristics of an area affect the occurrence and intensity of forest fires (Moritz et al. 2012).

B. Sample Library Generation

Since the CNN conducts its effective training in a fully supervised manner, the training and validation sample libraries must be constructed first. A binary classification method was adopted for the susceptibility analysis of forest fires, and samples were classified to either the forest fire class or the nonfire class, with 1 representing the fire class and 0 representing the nonfire class.

The CNN has many learnable parameters to estimate; thus, this predictor requires more data to achieve sufficient training. Table 2 shows that there is a large annual variation in the actual number of fire points in 2002–2009 (data from 2010 were used as the test dataset and therefore not included in the training and validation process). A simple random sampling method may lead to sample quantity imbalances, while small samples would lead to overfitting the model. Therefore, to generate more samples and keep the sample quantity balanced, proportional stratified sampling was adopted. The total number of forest fires was multiplied by the sampling rate (0.8) to determine the number of fire samples in every year. The same number of nonfire samples was randomly selected, which constitutes a total of 4800 training samples.

C. Architecture

The architecture of the proposed convolutional neural network (CNN) model was completed by referring to the AlexNet model (Krizhevsky et al. 2013) and the architecture and hyperparameters were tuned based on our datasets. As mentioned above, each input patch was a 3-dimensional data representation of size $n \times n \times c$, taking $224 \times 224 \times 11$ as an example. Figure 3 shows the main architecture of the prediction model of the CNN for forest fire susceptibility. There were a total of three convolution layers, two pooling layers, and three fully connected layers. The first three consecutive convolution layers had 64, 128, and 256 kernels, with uniform kernel sizes of 3×3. Each convolution layer was followed by an activation function (ReLU) and a pooling layer. Zero padding was employed to retain in-plane dimensions. All of the pooling layers perform max-pooling and summarize a 2×2 neighborhood with a stride of 2 pixels. At the end, the next three weight layers were fully connected layers with 128, 64, and 32 neurons each. Finally, the output of the last fully connected layer was fed into a 2-way classifier with an activation function named softmax, which computes the probabilities for the two classes' labels.

The parameters in CNN, which are automatically learned during the training process, refer to kernels in the convolution layers and weights in the fully connected layers. Training the CNN network is a process to find appropriate parameters to minimize the error between the predicted results and the ground truth labels on a training dataset. The CNN converted each input patch from the original pixel values to the final probability classification results, and the parameters were calculated by a loss function through feed-forward propagation. The learnable parameters were updated according to the loss value by using the stochastic gradient descent based on the backpropagation algorithm.

The hyperparameters are the variables that need to be set before the training process begins. Dropout is a recently introduced regularization technique and the fully connected layers are followed by dropout rates of 0.5 to mitigate overfitting. Bergstra and Bengio (2012) argued that random searches are more efficient for hyperparameter optimization than grid searches and manual searches. A random search was used to optimize the hyperparameters and improve the accuracy and speed of the model. Adam, an efficient stochastic optimization algorithm based on the gradient (Kingma and Ba 2014), was selected as the optimizer.

D. Performance Evaluation

$$\text{Overall accuracy} = \frac{TP + TN}{TP + TN + FP + FN}; \quad \text{Specificity} = \frac{TN}{FP + TN}$$

$$\text{Sensitivity} = \frac{TP}{TP + FN}; \quad \text{PPV} = \frac{TP}{FP + TP}; \quad \text{NPV} = \frac{TN}{FN + TN}$$

The evaluation criteria are a key factor in assessing the classification performance and guiding the classifier modeling (Sokolova and Lapalme 2009). In this article, a two-class classification method is modeled to predict forest fire susceptibility. Thus, five statistical measures including overall accuracy, specificity, sensitivity, positive predictive value (PPV), and negative predictive value (NPV) are employed to appraise the classification capability (Tien Bui

et al. 2017). The five statistical measures are computed in the following manner:

where TP (true positive) and TN (true negative) are the number of samples that are correctly classified as positive (fire class) and negative (nonfire class) observations, respectively. FP (false positive) and FN (false negative) are the number of samples that are misclassified. Sensitivity is the percentage of positive (fire class) observations that are correctly classified whereas specificity is the percentage of negative (nonfire class) observations that are correctly identified.

The ROC curve has been increasingly utilized to evaluate and validate the global performance assessment of the prediction models in ML and data mining research (Pourtaghi et al. 2016; Satir et al. 2016). It depicts the trade-offs between the TPs and FPs rather than arbitrarily selecting a particular threshold (Freeman and Moisen 2008). A ROC plot is constructed by plotting the TP rate (TPrate, sensitivity) on the Y-axis against the FP rate (FPrate, 1.0—specificity) for all possible thresholds from 0 to 1 on the X-axis. The ROC plot for a good classifier tends to rise sharply at the origin and then level off near the maximum value of 1. A trivial classifier will result in a ROC graph near the diagonal where the TP rate is equal to the FP rate for all thresholds. The AUC is generally considered to be an important index to quantitatively assess the overall accuracy of the classifier's performance. An AUC value near 0.5 means that the predictive ability of the model is completely random and a value of 1.0 represents a perfect prediction without misclassification. The closer the AUC value is to 1, the better the performance of the forest fire prediction model. The AUC measure is computed by obtaining only the area of the graphic:

$$AUC = \frac{1 + TP_{rate} - FP_{rate}}{2}$$

IV. Technical Process for Predicting Forest Fire Susceptibility

The technical process can be summarized into the following five steps. The detailed procedure of this study is depicted in:

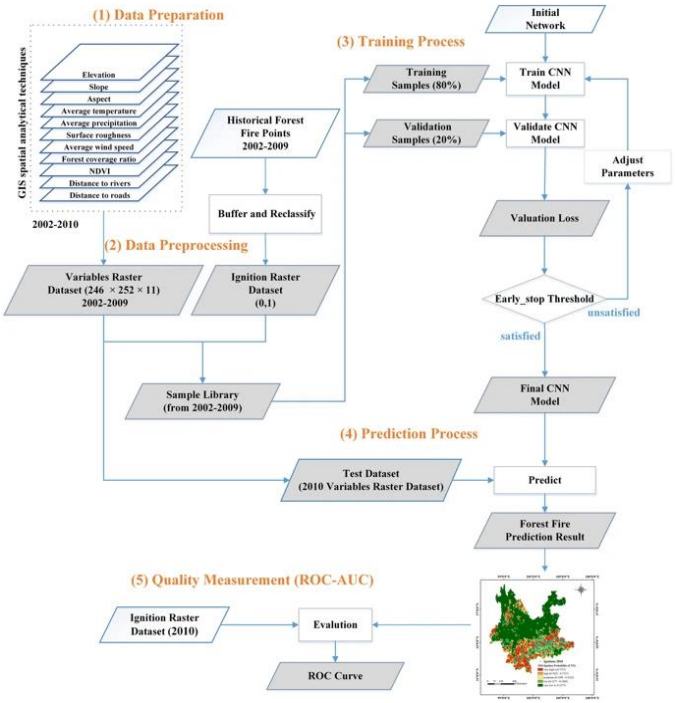
Step 1 Construct the fire and nonfire inventory maps and create maps of the influencing factors that can potentially influence the ignition susceptibility.

Step 2 Preprocess all datasets and generate the training/validation samples.

Step 3. Design the architecture of the CNN model, optimize the CNN hyperparameters, and train the CNN classifier.

Step 4 Predict forest fire susceptibility using the VRD of 2010.

Step 5 Evaluate the performance of the proposed model.



The CNN predication model was constructed under a graphics processing unit (GPU) acceleration environment using the Keras DL framework that uses TensorFlow as a backend, which is a Python-based DL library. The system configuration used in the lab environment is as follows: Intel Core i7 CPU, 16 GB RAM, Windows10 OS, and an NVIDIA GeForce GTX 1070 with 12 GB of onboard memory.

A. *Results*

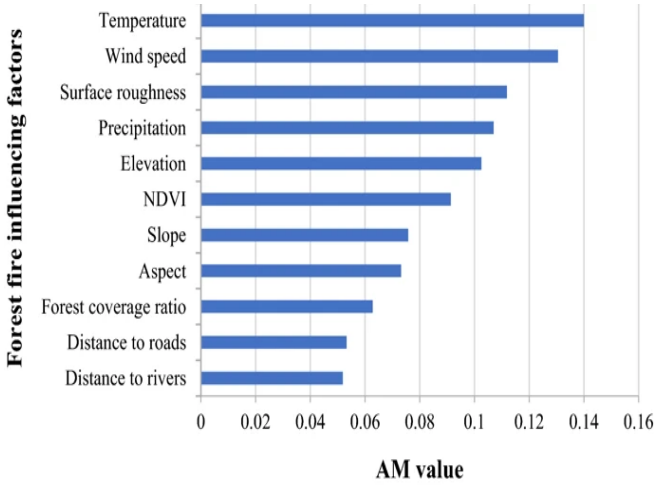
This section first shows the results of multicollinearity analysis and an information gain ratio (IGR) for the selection of forest fire influencing factors. The loss and the accuracy in the training/validation phases were tracked. Then, the test dataset was fed into the trained model and the prediction map of ignition probabilities was constructed by the CNN model. Finally, the performance of the proposed model was compared with benchmark methods.

B. *Relative Importance Analysis of Influencing Factors*

According to the results of a multicollinearity analysis of the 14 forest fire influencing factors , three factors—precipitation rate, specific humidity, and maximum temperature—did not satisfy the critical values, suggesting the existence of multicollinearity and should be excluded from further analyses.

For the IGR method, the factors with a higher value of average merit (AM) indicate a stronger prediction ability of

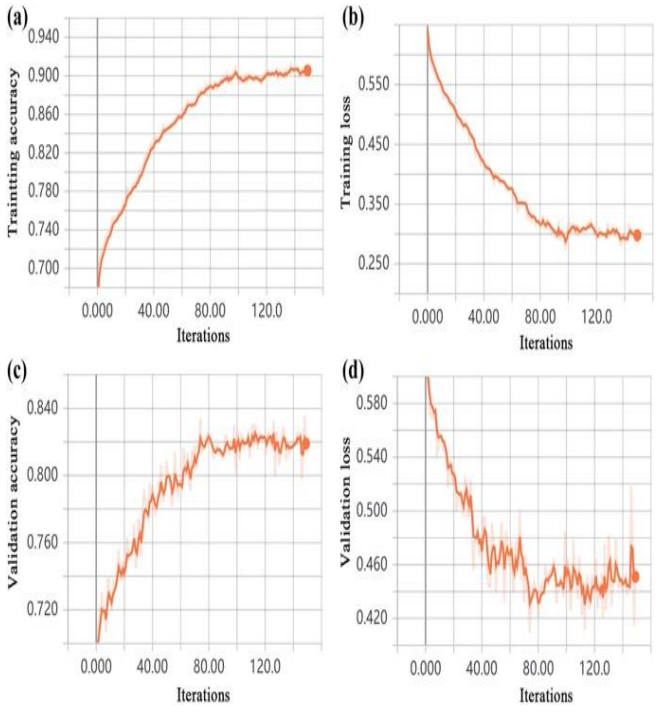
the model. However, factors with AM values equal to or less than 0 indicate a "null" contribution to the forest fire susceptibility model and should be excluded from further analysis (Bui, Tuan, et al. 2016). The results listed show that the AM values of all remaining 11 influencing factors are greater than 0, indicating that all these influencing factors contribute to the model and should be retained in the following prediction process. Temperature is the most important factor for forest fire susceptibility, with the highest AM value of (0.139). It is followed by wind speed (0.131), surface roughness (0.112), precipitation (0.107), and elevation (0.102).



C. Model Accuracy

The training process was divided into training and validation phases. The validation dataset was used to monitor the model classification performance after the end of each epoch, and the validation results were used as the basis for whether the training process should be terminated earlier or the hyperparameters should be fine-tuned. The loss and accuracy were two important indicators for evaluating the effect of model training. Callback functions were applied to adjust the training state and statistics during model training, including "EarlyStopping," "ReduceLROnPlateau," and "ModelCheckpoint."

Figure 6 shows the loss and the training/validation accuracy using TensorBoard for visualization. After the training phase, the training accuracy was close to 91%, the training loss no longer decreased after the 100th epoch and tended to fit, and the minimum loss value was 0.3. After the 80th epoch, the validation loss no longer decreased and tended to converge. However, the validation loss rose slightly after the 120th epoch, suggesting overfitting may have occurred. Immediately, the EarlyStopping function ended the training process to decrease the phenomenon of overfitting. The validation accuracy of the model reached 82%, and the minimum validation loss was 0.45.



D. Discussion

This section first describes the advantages of CNN compared with benchmark methods, then discusses the selection of the model architecture and window size, and finally discusses the differences between the CNN and traditional ML algorithms.

E. Advantage of the Convolutional Neural Network Method

Compared with the benchmark methods, the CNN has the following advantages. First, because the CNN can consider the correlation of adjacent spatial information, it has advantages in the study of problems with spatial and geographical correlation characteristics. Second, the CNN preserves the spatial relationships between pixels by learning the internal feature representations from factor vectors. The process of DL reveals the deep features and can distinguish the differences between different geographical units. The CNN was used to conduct multiple convolution and pooling operations to extract the characteristics. As the convolutions and pooling increased, these features became more advanced and more abstract. These abstract features depicted the degree of forest fire susceptibility, which was the decisive factor for determining forest fire susceptibility. Third, the CNN reduces the number of weights that need to be trained and the computational complexity of the network through weight sharing.

F. Model Sensitivity

The architecture of the CNN model should be selected in accordance with the quantity of sample data and the problem complexity. For those with small quantities of sample data and simple classification problems, the complex structure was prone to model overfitting. For those with large quantities of sample data and complex classification

problems, the simple structure was prone to model nonconvergence. Both of these problems should be avoided as much as possible in the training of the CNN model.

The selection of the window size should be consistent with the maximum geospatial area that affects the forest fire susceptibility of the center window pixel. A larger window size means that the pixels of a larger geographical unit impact the center window pixel. In fact, the pixels that affect the fire susceptibility of the center window pixel have a certain geographical spatial range. Therefore, the selection of the window size must be appropriate. After the experiments, the 25×25 window size was most reasonable. First, this size corresponds to the maximum geographical spatial range that affects the fire susceptibility of a pixel. Second, the size of this window is smaller than that of the commonly used windows (for example, 224×224) in image processing because forest fire probability prediction and image classification are completely different applications, and the CNN model that is used for image processing cannot be completely duplicated.

Conclusion

In this article, we investigated a CNN with deep architectures for the spatial prediction of forest fire. Past forest fire locations from 2002 to 2010 were extracted and a set of 14 forest fire influencing factors were optimized using multicollinearity analysis and the IGR technique. We explored the preprocessing methods for forest fire influencing factors and the methods for generating effective training/validation sample libraries. The CNN architecture suitable for the prediction of forest fire susceptibility in the study area was designed, and hyperparameters were optimized to improve the prediction accuracy. Several common methods, such as more training samples, regularization (dropout), batch normalization, and reduced architecture complexity, were used in the CNN model to mitigate overfitting. Then, the test dataset was fed into the trained model and the prediction map of ignition probabilities was constructed by the CNN model. Finally, the performance of the proposed model was compared with traditional ML methods using several statistical measures, including WSRT, ROC, and AUC.

Through this research, we found that the CNN model performs better than the benchmark methods. The CNN model ($AUC = 0.86$) has higher predictive power than the benchmark methods according to the ROC–AUC. The probability result obtained by the CNN can clearly distinguish the very high and very low susceptible zones, and the susceptibility spatial pattern was very distinct. The CNN model shows a strong generalization ability and the prediction time of the CNN was relatively short when using GPU-accelerated computing technology. In conclusion, the CNN has the advantages of considering neighborhood information, extracting deep features, sharing weights, and pooling operations, which allow the CNN to obtain better prediction results. The CNN model will have important practical application value for forest fire prevention planning and forest management.d by the original foreign-language citation [6].

There are still some limitations in the research. For example, the influence of different CNN architectures—such as VGG-net (Visual Geometry Group Network), RES-net Residential Energy Services Network), and GoogLeNet—on forest fire prediction results have not been studied in depth. In addition, more actual data are needed for the experimental verification of the method. In recent years, the application of CNNs has become increasingly extensive. Many different variants of the architecture have been derived and many ensemble classifiers have been proposed. Comparing various classifiers and exploring the most suitable models to improve forest fire prediction should be investigated in the future.

References

[1] Albertus Joko Santoso ; Findra Kartika Sari Dewi ; Thomas Adi Purnomo Sidhi
Natural disaster detection using wavelet and artificial neural network
(*https://ieeexplore.ieee.org/document/7237228*)

[2] Ananya Gupta, Elisabeth Welburn, Simon WatsonHujun Yin
CNN-Based Semantic Change Detection in Satellite Imagery
https://link.springer.com/chapter/10.1007/978-3-030-30493-5_61

[3] Aich, S., Van Der Kamp, W., Stavness, I.: Semantic binary segmentation using convolutional networks without decoders. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 182–186, 2018 June (2018). https://doi.org/10.1109/CVPRW.2018.00032

[4] Amit, S.N.K.B., Aoki, Y.: Disaster detection from aerial imagery with convolutional neural network. In: International Electronics Symposium on Knowledge Creation and Intelligent Computing, pp. 239–245. IEEE, september 2017. https://doi.org/10.1109/KCIC.2017.8228593

[5] Barzohar, M., Cooper, D.B.: Automatic finding of main roads in aerial images by using geometricstochastic models and estimation. IEEE Trans. Pattern Anal. Mach. Intell. 18(7), 707–721 (1996). https://doi.org/10.1109/34.506793

[6] Bhattacharjee, S., Roy, S., Das Bit, S.: Post-disaster map builder: crowdsensed digital pedestrian map construction of the disaster affected areas through smartphone based DTN. Comput. Commun. 134, 96–113 (2019). https://doi.org/10.1016/j.comcom.2018.11.010

[7] Boccardo, P., Giulio Tonolo, F.: Remote sensing role in emergency mapping for disaster response. In: Lollino, G., Manconi, A., Guzzetti, F., Luino, F., Culshaw, M., Bobrowsky, P. (eds.) Engineering Geology for Society and Territory - Volume 5, pp. 17–24. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-09048-1_3

[8] Chaurasia, A., Culurciello, E.: LinkNet: exploiting encoder representations for efficient semantic segmentation. In: 2017 IEEE Visual Communications and Image Processing, VCIP Description 2017, January 2018, pp. 1–4, June 2018. https://doi.org/10.1109/VCIP.2017.8305148

[9] Demir, I., et al.: DeepGlobe 2018: a challenge to parse the earth through satellite images. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, June 2018, pp. 172–181, May 2018. https://doi.org/10.1109/CVPRW.2018.00031

[10] Kethavath Srinivas, Mohit Dua
Fog Computing and Deep CNN Based Efficient Approach to Early Forest Fire Detection with Unmanned Aerial Vehicles
https://link.springer.com/chapter/10.1007/978-3-030-33846-6_69

[11] Lu, R., Heung, K., Lashkari, A.H., Ghorbani, A.A.: A lightweight privacy-preserving data aggregation scheme for fog computing-enhanced IoT. IEEE Access 5, 3302–3312 (2017)
https://ieeexplore.ieee.org/document/7869305

[12] Muhammad, K., Ahmad, J., Mehmood, I., Rho, S., Baik, S.W.: Convolutional neural networks based fire detection in surveillance videos. IEEE Access 6, 18174–18183 (2018)

https://ieeexplore.ieee.org/document/8307064

[13] Muhammad, K., Ahmad, J., Baik, S.W.: Early fire detection using convolutional neural networks during surveillance for effective disaster management. Neurocomputing 288, 30–421 (2018)

https://www.sciencedirect.com/science/article/abs/pii/S0925231217319203?via%3Dihub

[14] Guha-Sapir, D., Vos, F., Below, R., Penserre, S.: Annual disaster statistical review 2015: the numbers and trends, 2015. http://www.cred.be/sites/default/files/ADSR_2015.pdf

[15] Yuan, C., Zhang, Y., Liu, Z.: A survey on technologies for automatic forest fire monitoring, detection, and fighting using unmanned aerial vehicles and remote sensing techniques. Can. J. For. Res. 45(7), 783–792 (2015)

https://www.nrcresearchpress.com/doi/10.1139/cjfr-2014-0347#.XgASyHUzZuQ

Разработка метода фальсификации данных для упрощения разработки и тестирования

Semen D. Sheremetov
Peter the Great St.Petersburg Polytechnic University
St. Petersburg, Russia
s.sheremetov@yandex.ru

***Аннотация* — На сегодняшний день при разработке программного обеспечения востребованной задачей является задача подменна данных. Решение данной задача помогает при тестировании и разработке. Проведен сравнительный анализ подходов и обоснование выбора разработки собственного подхода. Предложена концептуальная схема с описанием модулей. Описаны функциональные и нефункциональные требования к системе. На основе требований и концептуальной схемы была предложена архитектурная схема программного средства с описанием модулей и их взаимодействием между собой и с внешним миром.**

Ключевые слова—инструментирование, программный, метод, архитектура сервера, клиент-серверное приложение, тестирование, распределенная, взаимодействие, kafka.

VI. Введение

На сегодняшний день при разработке программного обеспечения востребованной задачей является задача подменна данных. С подменной данных тесно связаны такие процессы разработки программного обеспечения, как разработка и тестирование. Так же данная тема актуальна во всех сферах информационных систем, например, web-приложения, распознавание образа и базы данных. Решение данной задача помогает при тестировании и разработке.

Например, описанную ниже ситуацию тяжело решить без подменных данных. REST API может являться бэк-энд стороной любого мобильного или веб приложения. Зачастую во время разработки, REST API попросту не готов. Чтобы увидеть мобильное или веб приложений в действии вам потребуется сервер, которые будет возвращать данные в нужном нам формате.

В работе была поставлена следующая цель: разработать подход для подменных данных на всех этапах разработки.

Для выполнения цели были поставлены следующие задачи:

- Исследование численных методов машинного обучения и программных средств для предсказания;
- Исследование существующих программных средств и подходов;
- Провести сравнительный анализ программных средств;
- Разработать клиент-серверную архитектуру приложения;
- Разработать подход для разработки программное средство на основе разработанной архитектуры.

Для выбора подхода в реализации необходимо было провести сравнительный анализ существующих подходов, который представлен ниже.

VII. Обзор существующих методов

Существуют следующие подходы, для подменны данных: mock, stub, generated db, json server. Рассмотри каждый из подходов более подробно.

Mock [9]. Объекты, которые настраиваются (например, специфично каждому тесту) и позволяют задать ожидания в виде своего рода спецификации вызовов, которые мы планируем получить. Проверки соответствия ожиданиям проводятся через вызовы к Mock-объекту.

Преимущества:

- Позволяет верифицировать систему

Недостатки:

- Занимают много места
- Требуют хранение радом с тестами

Stub [9]. обеспечивают жестко зашитый ответ на вызовы во время тестирования. Применяются для замены тех объектов, которые обеспечивают system under test (SUT) [7] входными данными. Также они могут сохранять в себе информацию о вызове (например параметры или количество этих вызовов) - такие иногда называют своим термином Test Spy. Такая "запись" позволяет оценить работу SUT, если состояние самого SUT не меняется.

Преимущества:

- Простота использования

- Позволяет проверить состояние

Недостатки:

- Отсутствие связей между различными вызовами
- Требуют хранение радом с тестами

Generated DB [2]. Данный подход заключается в том чтобы сгенерировать базу данных, реализационную или не реализационную [3], на основе моделей data transfer object (DTO) [4]

Преимущества:

- Предоставляет возможность сгенерировать базу данных на основе моделей
- Нет привязки к типу БД

Недостатки:

- Требует написание отдельного сервиса для работы с БД

JSON server [6]. Генерация полноценного программного продукта для работы с json-объектами.

Преимущества:

- Предоставляет CRUD [1] методы по работе с данными
- Предоставляет возможность сгенерировать сервис на основе моделей

Недостатки:

- Нету верификации типов данных
- Отсутствие согласованности данных

Рассмотренные подходы имею свои недостатки, поэтому было решено разработать собственный подход.

VIII. Разработка архитектуры метода подменны данных

Для разработки подхода необходимо было описать требования к нем. Требования к системе разделены на функциональные и нефункциональные [10]. В табл. 1 приведены нефункциональные требования.

Таблица 1
Нефункциональные требования к системе

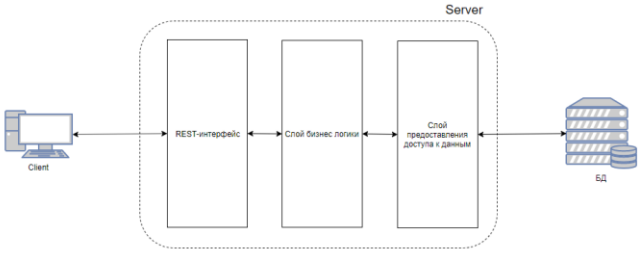
№	Требование
1	Система
1.1	Система должна иметь доступ к глобальной сети
2	Сервер
2.1	Сервер должен иметь доступ к глобальной сети
2.2	Сервер должен быть платформонезависимым
2.3	Сервер должен поддерживать работу с базой данных

В табл. 2 описаны функциональные требования.

Таблица IV
Функциональные требования к системе

№	Требование
1	Система
1.1	Система должна иметь клиент-серверную архитектуру
2	Сервер
2.1	Сервер должен иметь распределенную структуру
2.2	Сервер должен иметь модуль контроля данных для работы с базой данных (БД)
2.2.1	Сервер должен поддерживать методы сохранения и получения данных из БД
2.3	Сервер должен иметь RESTful-интерфейс
2.3.1	Получение данных должно осуществляться методами POST [17]
2.3.2	Отправка данных должна осуществляться методами GET [17]
2.4	Сервер должен быть маштабируемым

Из ходя из выше перечисленных требований требований, была разработана следующая архитектура.



Состоящая из клиента, сервера и базы данных. Далее опишем детально каждый из компонентов.

Клиент – компьютер, связанный с сервером по сети [11] и работающий с ним по принципу запрос-ответ.

Сервер – приложение, находящееся в постоянном доступе и имеющее статичный ip-адрес, способное обрабатывать запросы от нескольких клиентов [12].

База данных – сгенерированное на основе модели данных хранилище объектов.

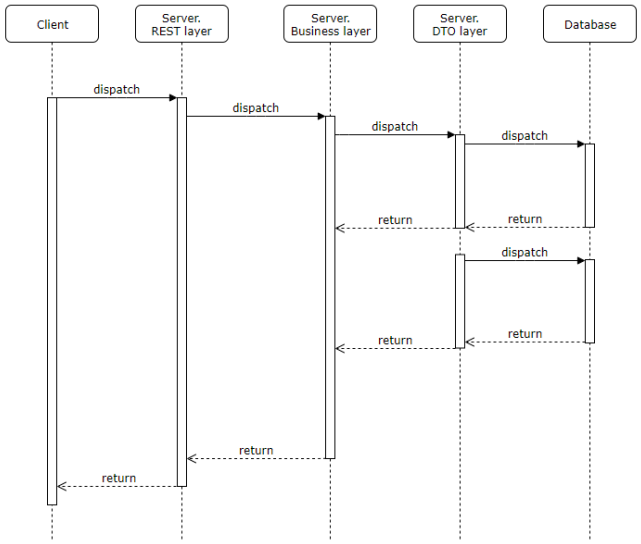
В свою очередь сервер состоит из следующих функциональных блоков.

REST-интерфейс – функциональный блок отвечающий за соответствие запросов пришедших на сервер и вызов функции из бизнес-логики. Так же данный слой отвечает за определение формата ответа, его преобразование и унификацию запроса к бизнес-логике.

Блок бизнес логики – данный блок отвечает за добавление дополнительной логики, например, агрегация нескольких моделей данных в одну, при работе с данными.

Блок предоставления доступа к данным – блок, отвечающий за обработку запросов к базе данных и обработку ответов от неё.

Ниже представлена диаграмма последовательности сообщений.



На диаграмме видно, что клиент отправляет запрос на сервер, который попадает на уровень REST интерфейса, где он перенаправляет запрос на уровень бизнес логики. Далее уровень бизнес логики через уровень DTO получает данные и агрегирует их, после чего данные возвращаются к клиенту.

IX. Заключение

В ходе выполнения работы были выполнены следующие задачи:

- Исследованы существующие программные средства и подходы;
- Проведен сравнительный анализ программных средств;
- Разработана клиент-серверная архитектура приложения;
- Разработан подход для разработки программное средство на основе разработанной архитектуры.

В процессе выполнения вышеописанных задач был произведен обзор существующих подходов машинного обучения и проведен их сравнительный анализ. На основе сравнительно анализа было принято решение разработать свой подход в разработке средства по подмене данных и разработка программного средства на основе данного подхода. Далее была предложена архитектурная схема и требования на основе которых разрабатывалось программное средство.

Источники

18. Raul Sidnei Wazlawick: Object-Oriented Analysis and Design for Information Systems – 2014 – p. 85
19. Jian Zhang, Chen Xu, S.-C. Cheung: Automatic generation of database instances for white-box testing, URL: https://ieeexplore.ieee.org/abstract/document/960612
20. Michael J. Hernandez: Database Design for Mere Mortals – 2014 – p. 119
21. Microsoft, URL: https://docs.microsoft.com/en-us/previous-versions/msp-n-p/ff649585(v=pandp.10)?redirectedfrom=MSDN
22. Таннебаун Э., М. ван Стеен Распределённые системы. Принципы и парадигмы. — М.: Питер 2003. – p. 880
23. Typicode, URL: https://github.com/typicode/json-server
24. Justyna Zander, Ina Schieferdecker, Pieter J. Mosterman: Model-Based Testing for Embedded Systems – 2012 – p. 31
25. Milind G. Limaye: Software Testing – 2009 – p. 228
26. Paul Ammann, Jeff Offutt: Introduction to Software Testing – 2017 – p. 299
27. Вигерс К. Битти Д. Разработка требований к программному обеспечению. 3-е изд. — М.: БХВ-Петербург 2018. – 736 с.
28. Joe Habraken, Matt Hayden: Networking – 2007 – p. 104
29. James F. Kurose, Keith W. Ross: Computer networking – 2016 – p.
30. Шереметов С.Д., Дроздов И.Д., Никифоров И.В. Программное средство прогнозирования атрибутов модели за счет численных методов экстраполяции // Современные технологии в теории и практике программирования. 24 апреля 2018 г. СПб.: Изд-во Политехнического ун-та. – 2018. – p. 91
31. Кобышев К.С., Вишневская Т.А. Клиент-серверное приложение для анализа данных по ДТП // Современные технологии в теории и практике программирования. 24 апреля 2018 г. СПб.: Изд-во Политехнического ун-та. – 2018. – p. 91.
32. Вигерс К. Битти Д. Разработка требований к программному обеспечению. 3-е изд. — М.: БХВ-Петербург 2018. – p. 736
33. Masse M. REST API design rulebook. — М.: O'Reilly Media, 2011. – p. 116

Построения рекуррентных искусственных нейронных сетей для решения задач анализа тональности текстов в обработке естественных языков

Гнатюк Игорь Владимирович
Институт компьютерных наук и технологий
Санкт-Петербургский политехнический университет Петра Великого
Санкт-Петербург, Российская Федерация
demospro1996@gmail.com

***Аннотация* - обратная связь клиентов имеет ключевую роль в маркетинге и бизнесе. Соответственно, первостепенно важно учитывать и налаживать обратную связь с людьми, потребляющими услуги или же товар и получать от них отзывы. Иногда таких отзывов могут быть тысячи и сотни тысяч, что затрудняет ручную обработку для такого массива данных. И приходится прибегать к методам машинного обучения и методикам обработки естественного языка. Для решение подобных задач существует много подходов, но наиболее точными и эффективными себя показали решения, использующие искусственные нейронные сети. В работе рассмотрен подход, способный улучшить анализ тональности текста, определить его эмоциональный окрас и извлечь информацию о том, какой характер носит отзыв: положительный, отрицательный или нейтральный. Для анализа результатов используется размеченный корпус комментариев разной длины, имеющий метки эмоциональной окраски каждого комментария, так как это принципиально важный фактор.**

Ключевые слова - большие данные, нейронные сети, обработка естественного языка, обработка текста.

I. ВВЕДЕНИЕ

Естественные языки имеют некоторые особенности. Первая особенность заключается в том, что эти языки никем специально не составлялись, а появились и эволюционировали в результате потребности людей в коммуникации. Никто полностью не знает правил естественных языков существует множество неписаных правил, которые меняются от региона к региону и со временем, особенно сейчас, в эпоху глобализации. Не обязательно соблюдать правила. Если человек скажет какую-то фразу неправильно, то, скорее всего, его всё равно поймут до определённого предела. А ещё, правила в естественном языке неоднозначны.

Соответственно, при рассмотрении текста человеком он сможет понять о чём в этом тексте говориться, несмотря на ошибки и какие-либо неточности, имея большое количество критериев выборки. Но обрабатывать большие объёмы текста для человека требует слишком много времени. Поэтому требуется прибегать к вычислительной мощности компьютера для решения подобного рода задач с использованием машинного обучения.

II. АКТУАЛЬНОСТЬ ТЕМЫ

С активным развитием технологий и накопившимися объемами данных, к тому же, ростом вычислительных мощностей, появилась возможность обрабатывать огромные объемы данных и обучать на них предсказательные модели. В частности, задачи анализа текста требуют больших вычислительных мощностей и современные кластеры, построенные на множестве графических процессоров (англ. graphics processing unit, GPU) предоставляют возможность для обучения подобных моделей.

На данный момент, с помощью моделей машинного обучения можно решать следующие задачи:

- распознавание речи и преобразование в цифровой сигнал;

- анализ текста (извлечение информации, анализ тональности текста, вопросно-ответные системы);

- генерирование текста;

- машинный перевод.

При решении подобных задач, модель должна учитывать множество зависимостей, которые далеко не всегда имеют нелинейный характер, между словами, предложениями и находить в них закономерности.

Выберем одну задачу: анализ текста и его тональности (sentiment analysis), так как подобного рода задачи можно наглядно оценить на правильность предсказания и выразить одним числом, к тому же, существует множество методов решения.

Этот тип задач хорош тем, что качество подобной модели можно оценить с помощью конкретных метрик. Некоторые из таких метрик строится на ковариационной матрице неточностей, рассмотренной на рисунке 1.

	Истинная метка - 0	Истинная метка - 1
Предсказание - 0	Истинно отрицательное предсказание True Negative, TN	Ложно-отрицательное предсказание False Negative, FN
Предсказание - 1	Ложно-положительное предсказание False Positive, FP	Истинно положительное предсказание True Positive, TP

Рисунок 1 - Матрица неточностей

Имея такую матрицу точность и полнота для каждого класса рассчитывается очень просто. Точность равняется отношению соответствующего диагонального элемента матрицы и суммы всей строки класса. Полнота – отношению диагонального элемента матрицы и суммы всего столбца класса. Расчет точности и полноты можно воспроизвести по формулам (1) и (2) соответственно:

$$Precision = \frac{TP}{TP+FP+\epsilon} \quad (1)$$

$$Recall = \frac{TP}{TP+FN+\epsilon} \quad (2)$$

Понятно что чем выше точность и полнота, тем лучше. Но в реальной жизни максимальная точность и полнота не достижимы одновременно и приходится искать некий баланс. Поэтому, хотелось бы иметь некую метрику которая объединяла бы в себе информацию о точности и полноте нашего алгоритма. Поэтому появилась такая метрика как F-мера[1], которую можно вычислить по формуле (3):

$$F_1 = \frac{2PrecisionRecall}{Precision+Recall+\epsilon} \quad (3)$$

где ϵ - малое по модулю положительное число (чтобы не было деления на 0).

В случае же нейронных сетей, подобного рода классификацию можно оценивать с помощью специальной формулы Binary Cross-Entropy / Log Loss [2], формула (4):

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1-y_i) \cdot log(1-p(y_i)) \quad (4)$$

где y_i - истинная метка для элемента выборки x_i, p(y_i) - предсказанная метка для элемента выборки x_i, N - количество элементов в выборке.

III. ОБЗОР СУЩЕСТВУЮЩИХ РЕШЕНИЙ

Перечислим некоторые методы для решения подобных задач:

- векторные разреженные модели [3];
- линейные модели классификации [4];
- построение деревьев решений и правил [5];
- нейросетевые подходы [6].

Рассмотрим данные подходы, выделим плюсы и минусы каждого, которые описаны в таблице 1.

Таблица 1. Методы решения задач обработки текста и их описание.

Подход	**Описание**	**Недостатки**	**Преимущества**
Векторные разреженные модели	Используется представление слов в векторном пространстве	Требует большого объема памяти для работы с большими разреженными матрицами	Хорошо работают на больших корпусах текста и показывают неплохие результаты
Линейные модели классификации	Построение корпуса текста как совокупность линейных зависимостей	На практике показывает не очень хорошие результаты для небольших предложений	Не требуют больших объемов ресурсов для обучения моделей
Построение деревьев решений и правил	Для корпуса текста строится огромный набор правил	Подобные модели страдают переобучением и плохо работают для текстов с другой семантикой	Могут выстраивать нелинейные зависимости и хорошо работают с большими предложениями
Нейросетевые подходы	Используется построение сложных нелинейных зависимостей	Требует огромных вычислительных мощностей	Показывают самую высокую точность

IV. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

A. Data science. Data mining

Наука о данных (англ. data science [7]) является разделом информатики, решающих задачи по анализу и обработки данных в условиях больших объемов. На сегодняшний день разделы науки о данных (англ. data science) активно используются в работе, т.к. за последнее время значительно увеличился объем данных, над которыми проводится работа.

Интеллектуальный анализ данных (англ. Data mining [8]) включает в себя методы обнаружения в данных знаний, полезных для принятия решений.

B. Machine Learning. Deep Learning

Машинное обучение (англ. machine learning, ML [9]) — класс методов искусственного интеллекта, характерной чертой которых является не прямое решение задачи, а обучение в процессе применения решений множества сходных задач. Для построения таких методов используются средства математической статистики, численных методов, методов оптимизации, теории вероятностей, теории графов, различные техники работы с данными в цифровой форме.

Глубокое обучение (глубинное обучение; англ. Deep learning[10]) — совокупность методов машинного обучения (с учителем, с частичным привлечением учителя, без учителя, с подкреплением), основанных на обучении представлениям (англ. feature/representation learning), а не специализированным алгоритмам под конкретные задачи.

Классикой нейронных сетей стал так называемый Многослойный Персептрон (англ. Multilayer Perceptron[11])., состоящий из множества слоев нейронов, среди которых каждый нейрон соединен с каждым нейроном предыдущего и следующего слоев. Принцип построения многослойной нейронной сети рассмотрен на рисунке 2.

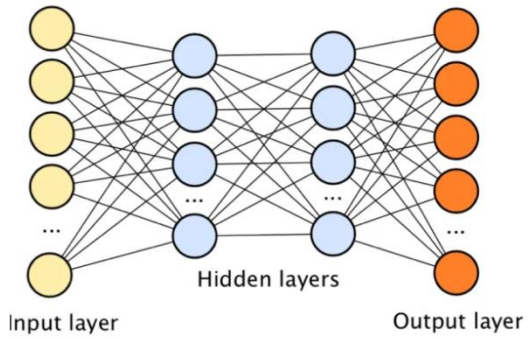


Рисунок 2 - Глубокая нейронная сеть

C. Natural language processing. RNN, LSTM

Обработка естественного языка (Natural Language Processing, NLP [12]) является перспективной областью информационных технологий и позиционируется как раздел искусственного интеллекта. С применением подходов NLP можно решать такие задачи как машинный перевод, генерация и анализ текста, извлечение информации и распознавание речи.

Естественным языкам часто противопоставляются формальные языки. Например, языки программирования или язык математических формул. Такие языки подчиняются строгим правилам, которые задаются, например, порождающими формальными грамматиками. Для таких языков процесс разбора и поиска ошибок детерминирован и есть эффективные алгоритмы, реализующие этот процесс.

Рекуррентные нейронные сети (Recurrent Neural Networks, RNNs[13]) — популярные модели, используемые в обработке естественного языка (NLP). Во-первых, они оценивают произвольные предложения на основе того, насколько часто они встречались в текстах. Это дает нам меру грамматической и семантической корректности.

Идея RNN заключается в последовательном использовании информации. В традиционных нейронных сетях подразумевается, что все входы и выходы независимы. Но для многих задач это не подходит. Еще одна интерпретация RNN: это сети, у которых есть «память», которая учитывает предшествующую информацию. Принцип работы классической рекурретной нейронной сети показан на рисунке 3.

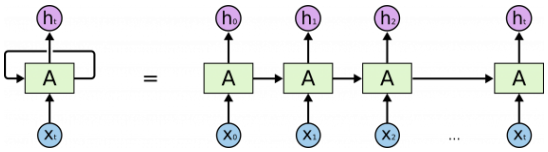


Рисунок 3 - Развертка цикла RNN

К сожалению, по мере увеличения этого разрыва между контекстом RNN теряют связь между информацией. На практике RNN не способны справиться с такими «долгосрочными зависимостями».

Долгая краткосрочная память (с англ. long short-term memory, LSTM[14]) — тип рекуррентной нейронной сети, способный обучаться долгосрочным зависимостям. Сети LSTM довольно популярны в наши дни, мы кратко говорили о них выше. LSTM не имеют принципиально отличающейся архитектуры от RNN, но они используют другую функцию для вычисления скрытого состояния. Память в LSTM называется ячейками, и вы можете рассматривать их как черные ящики, которые принимают в качестве входных данных предыдущее состояние. LSTM специально разработаны для устранения проблемы долгосрочной зависимости. Их специализация — запоминание информации в течение длительных периодов времени, поэтому их практически не нужно обучать. Несколько соединенных модулей LSTM показаны на рисунке 4.

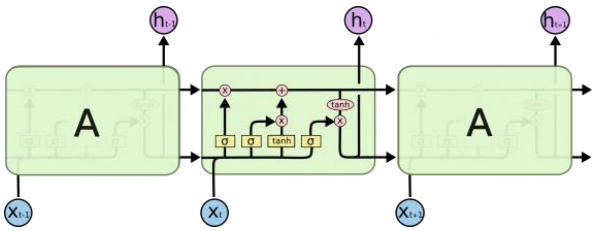


Рисунок 4 - Повторяющийся модуль LSTM

V. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ПРОДУКТА

A. Выбор набора данных и предобработка

Для обучения нейронной сети необходимо огромное количество данных, в случае sentiment analysis, большой корпус текста с определенный разметкой, т.е., для каждого предложения есть определенный набор таргет-значений, которые представляют собой ответы для той или иной задачи. К корпусу текста нужны метки дающие ответы на эмоциональный окрас каждого предложения.

Для анализа был выбран датасет Kaggle Toxic Comment Classification [15], который содержит в себе множество комментариев с метками, отображающими эмоциональный окрас. Размер датасета: около 500 000 примеров.

На рисунке 5 показана выборка из данного набора.

Для анализа нам потребуются два обработанных набора данных:

- очищенные токенизированные слова для каждого предложения;
- очищенные символьные n-граммы для каждого предложения.

Предобработка будет осуществляться следующим образом:

- очистка данных от пунктуации;
- переведение всех слов в нижний регистр;
- токенизация слов;
- создание n-грамм по символам.

N-грамма — последовательность из n элементов. С семантической точки зрения, это может быть последовательность звуков, слогов, слов или букв. Принцип разделения предложения на n-граммы по словам представлен на рисунке 6.

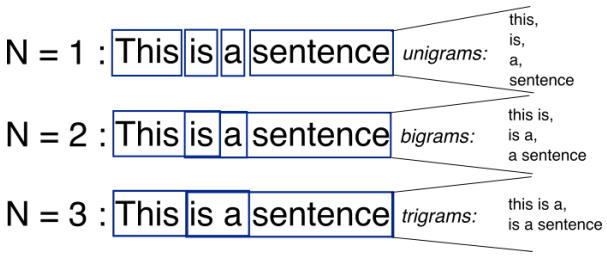


Рисунок 6 - Создание n-грамм

Оба датасета сохраняются в промежуточные файлы и загружаются уже непосредственно в нейронную сеть.

B. Построение нейронной сети

Реализация заключалась в том, чтобы взять конкретную архитектуру, которая хорошо решает поставленную задачу и улучшить ее при помощи добавления в нее элементов сиамской сети, которые помогают избегать ситуаций потери информации когда попавшееся слова, которое могло быть просто написано с ошибкой, отсутствует в словаре.

В качестве основы была построена архитектура на рисунке 7.

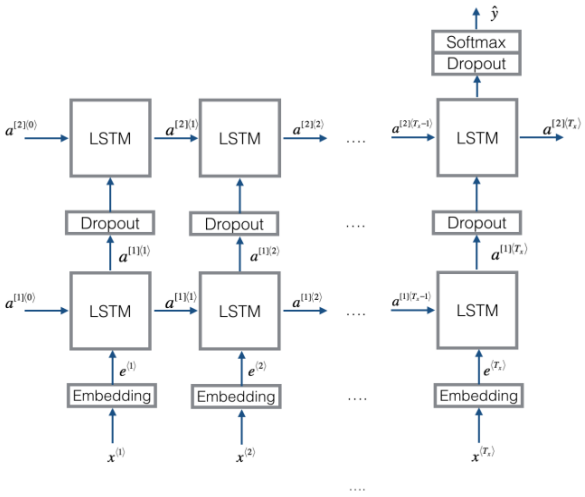


Рисунок 7 - Базовая архитектура

	id	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
0	0000997932d777bf	Explanation\nWhy the edits made under my usern...	0	0	0	0	0	0
1	000103f0d9cfb60f	D'aww! He matches this background colour I'm s...	0	0	0	0	0	0
2	000113f07ec002fd	Hey man, I'm really not trying to edit war. It...	0	0	0	0	0	0
3	0001b41b1c6bb37e	"\nMore\nI can't make any real suggestions on ...	0	0	0	0	0	0
4	0001d958c54c6e35	You, sir, are my hero. Any chance you remember...	0	0	0	0	0	0
5	00025465d4725e87	"\n\nCongratulations from me as well, use the ...	0	0	0	0	0	0
6	0002bcb3da6cb337	COCKSUCKER BEFORE YOU PISS AROUND ON MY WORK	1	1	1	0	1	0
7	00031b1e95af7921	Your vandalism to the Matt Shirvington article...	0	0	0	0	0	0
8	00037261f536c51d	Sorry if the word 'nonsense' was offensive to ...	0	0	0	0	0	0
9	00040093b2687caa	alignment on this subject and which are contra...	0	0	0	0	0	0

Рисунок 5 - Выборка из набора данных

В данной архитектуре применяются:

- Embedding слои, которые представляют слова в векторном пространстве, их тренировка представляет собой такую же задачу оптимизации как и нейронная сеть в целом;
- LSTM слои;
- регуляризационная техника Dropout, вставленная между слоями LSTM;
- в конце находится слой с Softmax,а перед ним еще один слой Dropout.

На ее основе была построена более продвинутая архитектура, которая представляет собой сиамскую сеть. Предлагаемая архитектура изображена на рисунке 8.

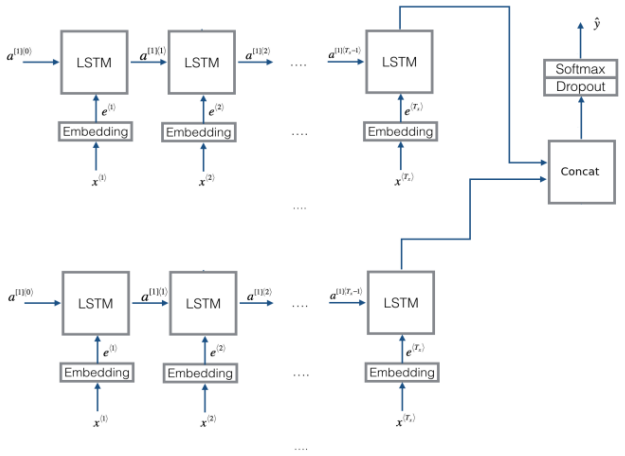


Рисунок 8 - Предлагаемая архитектура

Данная архитектура имеет 2 входные точки основанные на Embedding слоях, первая принимает закодированную последовательность из токенизированных слов, вторая последовательность из символьных n-грамм.

Обе входные точки тренируются как отдельные сети и в итоге усредняются.

VI. АНАЛИЗ РЕЗУЛЬТАТОВ

Каждая сеть обучалась по 20 эпох, т.е. датасет целиком проходил через сеть, по окончанию каждой эпохи фиксировалась средняя ошибка на тренировке и на валидационном наборе данных.

В итоге, нейросеть на базовой архитектуре достигала значения loss-функции, в среднем, около 0.162315, это хороший показатель точности.

Примеры графиков значений loss-функции показаны на рисунке 9.

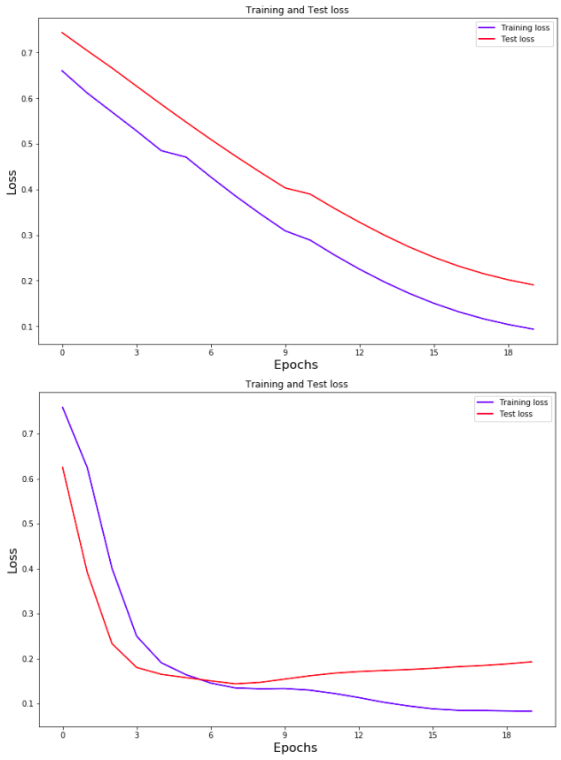


Рисунок 9 - Результаты обучения базовой архитектуры

Нейросеть на предлагаемой архитектуре обучалась ощутимо дольше, но показала явно более высокие результаты. Нейросеть достигала значения loss-функции, в среднем, около 0.09456.

Примеры графиков значений loss-функции показаны на рисунке 10.

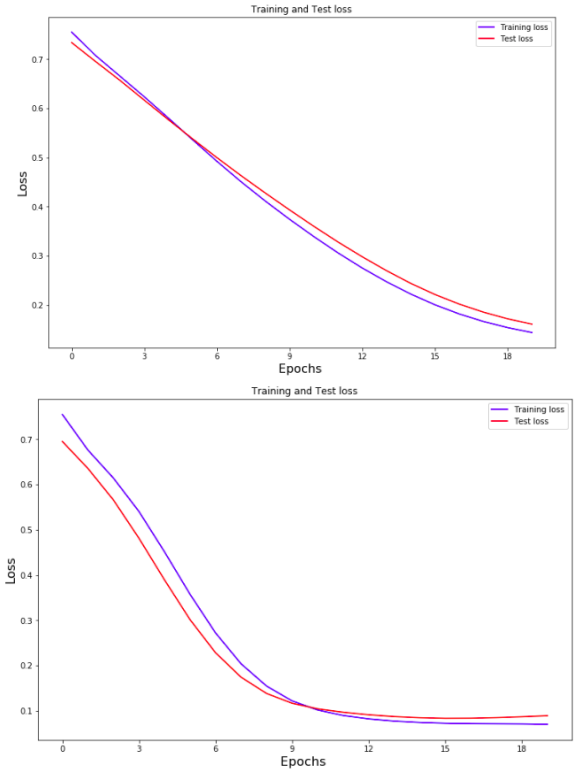


Рисунок 10 - Результат обучения предлагаемой архитектуры

VII. ВЫВОДЫ И ДАЛЬНЕЙШЕЕ УЛУЧШЕНИЕ

Проанализировав результаты работы программного продукта можно говорить о явном улучшении качества анализа со стороны нейронной сети, о рабочем методе избежании потери информации.

Опираясь на комплексные результаты нескольких алгоритмов можно говорить о более точной аналитике. Значимость проекта с экономической точки зрения позволяет развиваться бизнесу, что в свою очередь положительно влияет на обратную связь от клиентов независимо от объёмов информации.

С научной точки зрения проект показывает пример использования методов разных областей для получения результата, решающего задачи анализа текста.

Рекомендации по развитию проекта включают в себя использование современных техник построения нейронных сетей для улучшения качества и быстродействия обучения.

СПИСОК ИСТОЧНИКОВ

[1]Черноруцкий, Игорь Георгиевич. Практическая оптимизация и невыпуклые задачи / И. Г. Черноруцкий // Научно-технические ведомости Санкт- Петербургского государственного политехнического университета. Сер. : Информатика. Телекоммуникации. Управление = St. Petersburg state polytechnical university journal. Computer science. Telecommunications and control systems : научное издание. Санкт-Петербург. 2013. No 4 (176). С. 79-86. ISSN 1994-2354. ISSN 2304-9766.

[2] Machine Learning Book. Andriy Burkov. Издательство OReilly. ISSN 978-5-93286-159-2.

[3] Detection and classification of social media-based extremist affiliations using sentiment analysis techniques [Электронный ресурс] URL: https://link.springer.com/article/10.1186/s13673-019-0185-6 (дата обращения 15.09.2019).

[4] Optimization of sentiment analysis using machine learning classifiers [Электронный ресурс] URL: https://link.springer.com/article/10.1186/s13673-017-0116-3 (дата обращения 28.09.2019).

[5] A C4.5 algorithm for english emotional classification [Электронный ресурс] URL: https://link.springer.com/article/10.1007/s12530-017-9180-1 (дата обращения 6.10.2019).

[6] Big Data: Deep Learning for financial sentiment analysis [Электронный ресурс] URL: https://link.springer.com/article/10.1186/s40537-017-0111-6 (дата обращения 18.10.2019).

[7] Jake VanderPlas. Python Data Science Handbook. OReilly. ISBN 978-0-491-78785-8

[8] Гитис, Леонид Хаскелевич. Кластерный анализ в задачах классификации, оптимизации и прогнозирования / Л.Х. Гитис. Москва : МГГУ, 2001. - 104 с. : ил. - (Серия: Практическая статистика для горных инженеров ; вып. 2). Библиогр.: с. 101-102. ISBN 5-7418-0010-6.

[9] Steven Bird. Natural Language Processing with Python. OReilly. ISBN 978-0-596-51784-9

[10] Черноруцкий, Игорь Георгиевич. Практическая оптимизация и невыпуклые задачи / И. Г. Черноруцкий // Научно-технические ведомости Санкт- Петербургского государственного политехнического университета. Сер. : Информатика. Телекоммуникации. Управление = St. Petersburg state polytechnical university journal. Computer science. Telecommunications and control systems : научное издание. Санкт-Петербург. 2013. No 4 (176). С. 79-86. ISSN 1994-2354. ISSN 2304-9766.

[11] Никифоров, Игорь Валерьевич (1988-). Курсовое проектирование по учебной дисциплине "Наука о данных и аналитика больших объемов информации" : учебное пособие / И. В. Никифоров ; Санкт- Петербургский политехнический университет Петра Великого. Санкт-Петербург : Изд-во Политехн. ун-та, 2017. 62 с. : ил. ; 20 см. ISBN 978-5-7422-5638-0.

[12] Открытые данные Санкт-Петербурга [Электронный ресурс] URL: https://link.springer.com/article/10.1007/s41019-017-0040-6 (дата обращения 18.10.2019).

[13] Эшби, У. Р. Введение в кибернетику / У. Р. Эшби : пер. с англ. Москва : Издательство иностранной литературы, 1959. 432 с.

[14]Bharath Ramsundar, Reza Bosagh Zadeh. TensorFlow for Deep Learning. www.allitebooks.com. OReilly. 101-102. ISBN 978-1-491-0010-3.

[15] Нишант Шакла. Машинное обучение и TensorFlow. Издательство «Питер», 2019

ISBN 978-1617293870 англ. ISBN 978-5-4461-0826-8 рус.