Министерство образования Республики Беларусь
Учреждение образования
Белорусский государственный университет
информатики и радиоэлектроники

УДК 004.457

Пасюкевич
Дмитрий Юрьевич

Автомасштабирование веб-сервиса с применением машинного обучения

АВТОРЕФЕРАТ
на соискание академической степени
магистра технических наук

по специальности 1-40 81 04 – Обработка больших объемов информации

подпись магистранта

Научный руководитель
Анисимов В.Я.
к.ф.м.н., доцент

подпись научного руководителя

Минск 2019

КРАТКОЕ ВВЕДЕНИЕ

Онлайн-сервисы, как правило, испытывают значительные колебания спроса на рабочую нагрузку. Статическое выделение ресурсов для удовлетворения пикового спроса может минимизировать снижение производительности во время высоких нагрузок, но может привести к излишнему избыточному распределению ресурсов до 40-50%, что приводит к значительным затратам на ресурсы и расходу энергии.

Недооценка предоставления ресурсов может привести к дорогостоящим нарушениям соглашения об уровне обслуживания (SLA); переоценка резервирования может привести к неустойчивым эксплуатационным расходам и плохому использованию ресурсов. Хотя автоматическое масштабирование можно использовать как в физических кластерах (для экономии затрат на электроэнергию), так и в виртуальных кластерах (для экономии расходов на аренду), успешное автоматическое масштабирование системы является сложной задачей. В частности, не очевидно, как много узлов (машин или виртуальных машин) должны быть добавлены или удалены для достижения требуемой производительности.

Также, в случае значительных затрат по времени для запуска нового объекта приложения (например дополнительная реплика базы данных, где данные из сохраненного бекапа приводятся в консистентное состояние с работающей системой), используя масштабирование основанное на машинном обучении, можно заранее знать о росте потребления и подготовить нужное количество требуемых объектов.

Имея точные данные и количестве поступающих запросов в ближайшем будущем, можно заранее подготовить вычислительные мощности для обработки поступающих запросов. Иначе, клиенты, пользующиеся сервером начнут получать ошибки, возникающие в следствие загруженности ресурсов, что может повлиять на впечатление пользователя и может негативно повлиять на развитие сервиса в целом.

Еще одним важным фактором динамического автомасштабирования вычислительных ресурсов является снижение затрат на работающие ресурсы. Большинство облачных провайдеров в качестве метрики для выставления цены используют время пользования. И в случае простаивающих серверов, владелец сервиса будет вынужден выделять из бюджета средства на оплату серверов, используемых не на полную мощность.

Типичные подходы к автоматическому масштабированию основаны на глубоком понимании приложения, базовой инфраструктуры и их динамики для точного масштабирования ресурсов. В отсутствие такой информации для тщательного изучения системы требуется исчерпывающее инструментарий и экспериментирование. Учитывая разнообразие приложений, работающих сегодня в центре обработки данных (ЦОД) и в облаке, можно ли разработать подход автоматического масштабирования, не зависящий от приложений? Хотя прогнозы будущей нагрузки могут помочь в автоматическом масштабировании, они все еще требуют понимания того, как нагрузка связана с производительно-

стью, для успешного масштабирования системы; Кроме того, точные прогнозы часто доступны не для всех приложений.

Методы моделирования по принципу «черного ящика» (black box), основанные на машинном обучении, стали подходящим решением для автоматического масштабирования. Типичные ML-подходы к автоматическому масштабированию основаны на обучении с подкреплением (reinforcement learning), цель которого состоит в том, чтобы определить наилучшее масштабирующее действие для данного состояния системы, используя исторические данные и метод проб и ошибок. Однако такие подходы связаны с большими накладными расходами из-за необходимости большого количества состояний для обучения. Другие подходы включают применение методов, основанных на линейной регрессии, для выявления взаимосвязи между состоянием системы и производительностью. Однако производительность часто нелинейно связана с использованием ресурсов.

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Цель и задачи исследования

Целью диссертационной работы является обучение нейронной сети для решения задач масштабирования вычислительных ресурсов сервиса, предоставляющего услуги по сети Интернет, а также интеграция с API выбранного провайдера облачных услуг.

Для достижения поставленной цели необходимо решить следующие задачи:

1.Провести анализ существующих архитектур нейронных сетей.

2.Провести анализ технологий по автоматизации запуска приложений и выбрать

3.Подготовить данные и обучить модель нейронной сети, выполняющей предсказание потребления вычислительных ресурсов системы.

4.Реализовать ПО, осуществляющее интеграцию разработанной модели с API систем оркестрации вычислительных ресурсов.

5.Провести экспериментальные исследования разработанной системы.

Объектом исследования являются системы оркестрации и автомасштабирования вычислительных ресурсов.

Предметом исследования является улучшение качества работы систем автомасштабирования для уменьшения затрат на работу вычислительных ресурсов.

Основной *гипотезой*, положенной в основу диссертационной работы, является возможность более точного предсказания требуемых вычислительных ресурсов априори на основании уже имеющихся данных использования веб-сервиса.

Личный вклад соискателя

Результаты, приведенные в диссертации, получены соискателем лично. Вклад научного руководителя В. Я. Анисимова, заключается в формулировке целей и задач исследования.

Структура и объем диссертации

Диссертация состоит из введения, общей характеристики работы, четырех глав, заключения, списка использованных источников и приложений. В первой главе представлен анализ существующих архитектур нейронных сетей, был обоснован выбор используемой архитектуры модели. Вторая глава посвящена анализу и выбору инструментов для управления и конфигурацией инфраструктуры. Были рассмотрены варианты запуска приложения с помощью виртуальной машины, а также с помощью технологий контейнеризации. В третьей главе предложены алгоритмы проведения автомасштабирования веб-сервиса, были сгенерированы данные для проведения исследования. Было разработано программное обеспечение, реализующее предложенные алгоритмы. В четвертой главе было проведено исследование производительности и эффективности работы алгоритмов на тестируемом проекте, а также произведено сравнение производительности по сравнению с уже предложенными алгоритмами. В приложении приведены предложенные алгоритмы, а также программный код реализации.

Общий объем работы составляет 61 страниц, из которых основного текста – 42 страниц, 17 рисунков на 4 страницах, 4 таблиц на 4 страницах, список использованных источников из 34 наименований на 3 страницах и 4 приложения на 8 страницах.

ОСНОВНОЕ СОДЕРЖАНИЕ

Во **введении** определена область и указаны основные направления исследования, показана актуальность темы диссертационной работы, дана краткая характеристика исследуемых вопросов, обозначена практическая ценность работы.

В **первой главе** проведен анализ и сравнение существующих архитектур нейронных сетей. Приведено краткое описание структуры сверточных и рекуррентных сетей. Указаны достоинства, недостатки и наиболее подходящие области примения описываемых архитектур сетей. Была выбрана сеть LSTM-RNN и обоснована целесообразность использования данной топологии для решения

поставленной задачи автомасштабирования веб-сервиса в зависимости от нагрузки.

Вторая глава посвящена анализу и проектированию ПО для управления ресурсами в распределенных системах. Были описаны виды и способы изоляции процессов приложения в системе для уменьшения влияние внешних факторов на стабильность работы сервиса. Для этого применяются технологии виртуализации и контейнеризации.

Облачные сервисы используют методы виртуализации для достижения эластичности крупномасштабных общих ресурсов. Виртуальные машины являются основой инфраструктуры, обеспечивая виртуализированные операционные системы. Контейнеры - это похожая, но более легкая концепция виртуализации; они требуют меньше ресурсов и времени для запуска, поэтому их предложили в качестве решения для более эффективной энкапсуляции приложения для размещения в облаке.

Контейнеризация обеспечивает:

1.Легковесную переносимую среду выполнения.

2.Возможность разрабатывать, тестировать и запускать приложения на большом количестве серверов.

3.Возможность соединения контейнеров между собой по сетевому интерфейсу.

Также в этой главе были описаны существующие технологии контейнеризации приложения, такие как Docker, Linux LXC и приведена таблица сравнения этих технологий.

Для управления большим количеством контейнеров в кластере серверов, используется дополнительное программное обеспечение (Kubernetes, Ansible, Chef). Было приведено описание каждого инструмента, дана характеристика использования, а также обоснован выбор используемого набора инструментов.

В **третьей главе** были сгенерированы тестовые данные запросов к веб-сервису, содержащие описываемый Slashdot-эффект и была приведена визуализация сгенерированных данных. Пример Slashdot-эффекта, при котором наблюдается значительный рост запросов к серверу показан на рисунке 1.

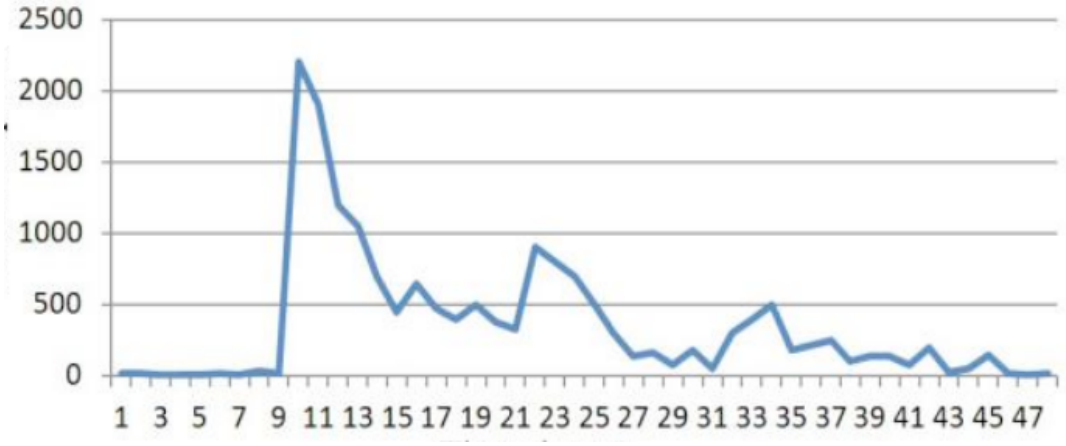

Рисунок 1 – Пример slashdot-эффекта

Также был проведен анализ уже предложенных алгоритмов, направленных на корректное устранение и предупреждение данной проблемы. Для этого было выбрано два наиболее эффективных алгоритма, предложенный A.Gandhi и второй алгоритм, предложенный Kanagala и Sekaran.

В этой главе были приведены разработанные алгоритмы, нивелирующий Slashdot-эффект, а также была дана реализация данного алгоритма на языке программирования Golang.

Первый алгоритм автоматического масштабирования использует две разных LSTM-RNN для прогнозирования будущего спроса. Первая LSTM-RNN обучается с нормальной рабочей нагрузкой без Slashdot-эффекта, а вторая сеть LSTM-RNN обучается только с рабочей нагрузкой Slashdot. и постоянно обновляются с использованием прогнозируемого и наблюдаемого потребления CPU. Требуемое CPU после шага итерации прогнозируется с использованием LSTM-RNN с самым низким показателем ошибки. Предсказанное потребление CPU отправляется в качестве входного параметра алгоритма принятия решения о соответствующем действии масштабирования. Количество виртуальных машин для увеличения или уменьшения задается в соответствии с разницей между прогнозируемыми и предоставленными ресурсами после шага итерации.

Алгоритм 2 показывает итерацию второго алгоритма автоматического масштабирования, который использует только одну сеть LSTM-RNN для прогнозирования требуемого CPU с нормальным соотношением Slashdot-нагрузок и обычных рабочих нагрузок.

Пользователь запускает процесс, который делает запрос по TCP к сети предсказания использования CPU и в зависимости от полученного значения запускает процесс chef client который осуществляет автомасштабирование виртуальных машин вверх или вниз в зависимости от предсказания. Были приведены формулы расчета пороговых значений нагрузки, которые позволяют принимать решение о масштабировании вычислительных ресурсов вверх или вниз. На вход алгоритма подаются данные о загрузке CPU за весь доступный период времени, предсказанное потребление CPU в определенный момент времени. А также время задержки на запуск новых виртуальных машин.

Также были предложены методы, позволяющие регулировать чувствительность системы к росту количества запросов. Пороги инициализируются одинаковыми значениями для всех приложений. Формулы вычисления максимального порогового значения; порогового значения, чуть меньшего, чем максимальное пороговое значение; минимального порогового значения приведено в формулах (1), (2), (3) соответственно. Однако из-за различий в характере рабочих нагрузок установка одинаковых значений для всех приложений увеличивает вероятность нарушения соглашений об уровне обслуживания (SLA). Поэтому все пороговые значения периодически и автоматически адаптируются с использованием медианного абсолютного отклонения требуемой истории потребления процессорного времени для каждого приложения соответственно.

$$ThrU := 1 - c_1{*}D \quad (1)$$
$$ThrbU := 1 - c_2{*}D \quad (2)$$
$$ThrL := 1 - c_3{*}D \quad (3)$$

где $c_1, c_2, c_3 \in R,\ c_1 < c_2 < c_3,$
D - среднеквадратичное отклонение от CPU_h

Используя, c_1, c_2, c_3 мы можем адаптировать безопасность предложенного алгоритма. Например, более низкие значения c_1и c_2 уменьшают стоимость, но увеличивают вероятность нарушения соглашений об уровне обслуживания (SLA).

В **четвертой главе** приведен анализ производительности разработанных алгоритмов в сравнении с выбранными алгоритмами в главе 3. Было проведено сравнение количества запущенных виртуальных машин, среднее время отклика, а также количество обработанных запросов в зависимости от времени. Сравнительные данные представлены в виде таблиц, а также визуализированы. График сравнения количества запущенных виртуальных машин в зависимости от времени и изменяющегося количества запросов к серверу показано на рисунке 2.

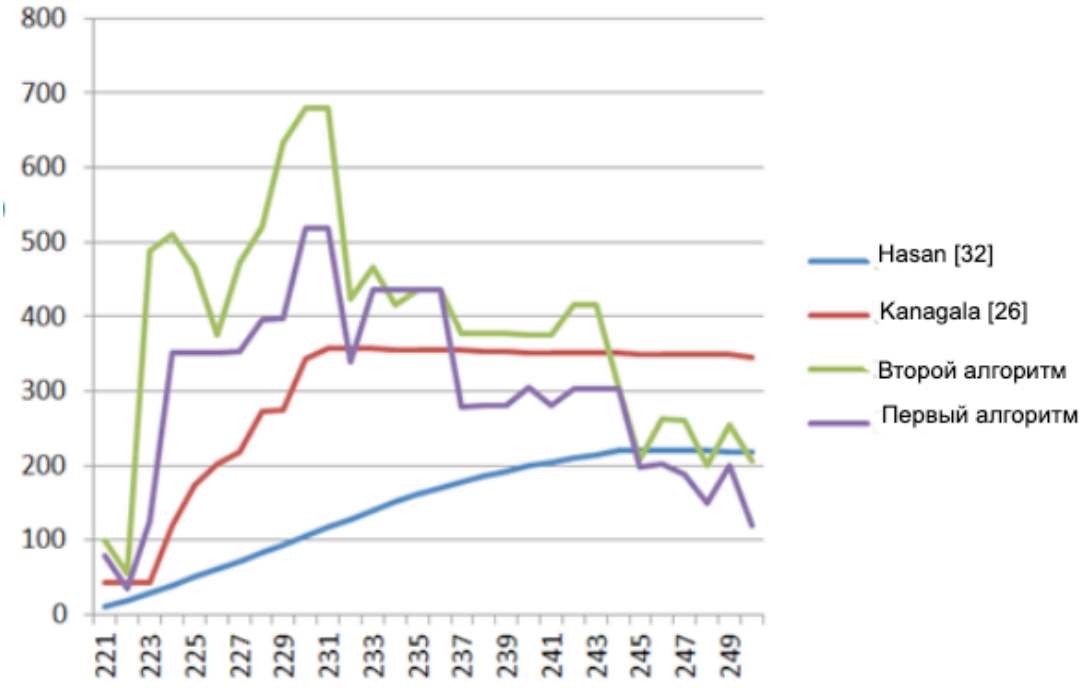

Рисунок 2 – Количество запущенных виртуальных машин в зависимости от времени

ЗАКЛЮЧЕНИЕ

Предложенные алгоритмы были оценены эмпирически в сравнении с некоторыми из уже существующих подходов. Результаты эксперимента показали, что предложенные алгоритмы превосходят другие как по экономии стоимости, так и по уровню обслуживания. На основании этих результатов можно сделать вывод о том, что использование рекуррентной нейронной сети с кратковременной памятью для распознавания и обработки Slashdot-эффекта может минимизировать ее последствия. В будущем можно будет использовать глубокую рекуррентную нейронная сеть с кратковременной памятью для распознавания поведения Slashdot. Глубокая LSTM-RNN эффективно применяется во многих областях и доказала свою эффективность на протяжении многих лет. Глубокие LSTM-RNN предлагают больше преимуществ по сравнению со стандартными LSTM RNN, благодаря наличию нескольких скрытых слоев. Каждый слой обрабатывает некоторую часть данных перед отправкой на следующий слой.

Количество клиентов облачных провайдеров растет с каждым годом и большая часть компаний, использующих вычислительные ресурсы для осуществления бизнеса, осуществляют или уже провели миграцию в облачные системы. Ожидается, что рынок облачных услуг продолжит интенсивный рост и в ближайшее время. Внедрение предложенного алгоритма позволит предоставлять более выгодные условия и более качественный набор услуг для клиентов, что может дать значительную прибыль за счет увеличения количества клиентов. Также это выгодно и компаниям-пользователям облачными услугами, так как они смогут тратить значительно меньше финансовых средств на инфраструктуру проекта, за счет более эффективного использования вычислительных ресурсов.

Основные научные результаты диссертации

1. Разработаны алгоритмы анализа запросов к веб-сервису и принятие решения для масштабирования вверх/вниз, либо бездействие. Алгоритм разработан с учетом возможного Slashdot-эффекта, при котором может наблюдаться значительный рост количества запросов в единицу времени. Разработанные алгоритмы автомасштабирования позволяют значительно снизить затраты на содержание элементов кластера, в результате адаптивного реагирования на рост количества запросов.
2. Разработано программное обеспечение, реализующее данные алгоритмы масштабирования с использованием LSTM-RNN.
3. Проведена оценка эффективности работы и точности реагирования на рост количества запросов, по сравнению с уже предложенными вариантами алгоритмов. Был произведен учет и сбор статистических данных об эффективности работы и реагирования на изменения количества запросов к кластеру приложения.

Рекомендации по практическому использованию результатов

1.Полученные результаты формируют теоретическую и практическую базу для разработки ПО компьютерных систем для решения задач автомасштабирования с применением LSTM RNN. Они могут быть использованы для модернизации и дальнейшего развития существующих систем.

2.Разработанные алгоритмы анализа запросов могут применяться и интегрироваться в набор программного обеспечения провайдеров облачных услуг для повышения качества сервиса, предоставляемого клиенту.

3.Также, разработанные алгоритмы анализа запросов могут быть применены на стороне пользователя облачными услугами для более эффективного использования вычислительных ресурсов и для уменьшения затрат за пользование облачными ресурсами.