УДК 681.3 DOI: 10.26102/2310-6018/2021.34.3.004

Оптимизация процесса распределения работ при управлении командной деятельностью в IT-компаниях с использованием машинного обучения

С.Г. Корчагин, Я.Е. Львович

ФГБОУ «Воронежский государственный технический университет», Воронеж, Российская Федерация

Резюме. Целью данной работы является оптимизация процесса распределения работ при управлении командной деятельностью в IT-компаниях с использованием алгоритмов машинного обучения, позволяющая снизить нагрузку на ключевого члена команды – менеджера, который является scrum-мастером. Рассматривается управление командой в scrum по модели Такмана с выделением 5 стадий развития: формирование, конфликтная, нормирующая, исполнительная, расставание. Для увеличения времени на микро-менеджмент внутри команды предложено автоматизировать рутинные процессы с использованием оптимизационного подхода. Показано, что наиболее времязатратной частью данного процесса является определение типа задач. Для этого необходимо детализировать техническое задание, отнести его компоненты (задачи) к определенному типу и поручить выполнение разработчику, который за самое короткое время достигнет требуемого результата. Рассмотрена структура оптимизационной модели распределения задач между разработчиками. Обосновано, что для формирования параметров критерия оптимизации требуется предварительная классификация этих задач по категориям с учетом возможностей членов команды. Обоснован выбор сверточной нейронной сети и применения машинного обучения для решения задачи классификации. В качестве исходных данных при обучении сети на начальных стадиях развития команды предлагается использовать тексты тестовых заданий и их распределение по категориям проектных задач.

Ключевые слова: agile-методы, оптимизация, сверточная нейронная сеть, классификация, кодирование слов, рекуррентные нейронные сети.

Для цитирования: Корчагин С.Г., Львович Я.Е. Оптимизация процесса распределения работ при управлении командной деятельностью в IT-компаниях с использованием глубокого машинного обучения. *Моделирование, оптимизация и информационные технологии.* 2021;9(3). Доступно по: https://moitvivt.ru/ru/journal/pdf?id=932 DOI: 10.26102/2310-6018/2021.34.3.004

Optimization of the work distribution process for managing activities in IT companies using machine learning

S.G. Korchagin, Y.E. Lvovich

Voronezh State Technical University, Voronezh, Russian Federation

Abstract: This work aims to optimize the task distribution process when working in an IT company using machine learning algorithms, which intends to reduce the load on the key team member - the manager, who is the Scrum Master. We consider a team in scrum according to the Takman model, highlighting five stages of development: formation, conflict, normalizing, executive, separation. To increase the time for micro-management within the team, it is proposed to automate routine processes using an optimization approach. It is shown that the most time-consuming part of this definition is the definition of the type of tasks. It is necessary to define the terms of reference, assign its components - tasks - to a specific type and entrust the implementation to the developer who will achieve the required result in the shortest possible time to do this. The structure of the optimization model for the distribution

of tasks between developers is considered. Preliminary classification of tasks by category is required, taking into account the capabilities of the team members. The choice of a convolutional neural network and the use of deep machine learning for solving the classification problem has been substantiated. As the initial data when training the network at the initial stages of team development, it is proposed to use the texts of test tasks and their distribution by categories of project tasks.

Keywords: agile-methods, optimization, convolutional neural network, classification task, word encoding, recurrent neural networks

For citation: Korchagin S.G., Lvovich Y.E. Optimization of the work distribution process for managing activities in IT companies using deep machine learning. *Modeling, Optimization and Information Technology*. 2021;9(3). Available from: https://moitvivt.ru/ru/journal/pdf?id=932 DOI: 10.26102/2310-6018/2021.34.3.004 (In Russ).

Введение

На современном этапе увеличение скорости разработок в IT-компаниях привело к увеличению рисков и потере качества разрабатываемого продукта. Для избежания проблемы с потерей качества были предложены гибкие методологии разработки в соответствии с концепцией Agille, которые являются альтернативой традиционным методам таким, как: каскадная модель, спиральная модель, V-модель и им подобные [1]. Основным отличием является то, что гибкие методологии подразумевают получение в конце каждого этапа работающего продукта. Эффективность Agile-методологий [2] достигается за счет снижения рисков, возникающие при разработке и получении результатов, соответствующих требованиям заказчика.

Одним из популярных методов гибкой разработки семейства Agile является Scrum. В данной методологии требования клиента разбивают на микрокомпоненты, которые максимально независимы друг от друга. Все этапы работ подразумевают постоянную коммуникацию между участниками команды и клиентом.

Управление командой в scrum осуществляется по модели Такмана и включающей в себя следующие стадии развития команды [3,4]:

1) Формирование. В данном этапе команда только формируется, члены команды еще не сработались, только знакомятся, выполняют рутинные задачи. Во время данного процесса важно накапливать информацию и приступать к реализации стратегии и целей. Менеджер в этот период определяет кто и как работает, сильные и слабые стороны каждого члена команды;

2) Конфликтная. После формирования в единую группу, участники предлагают свои идеи. В большинстве случаев самые опытные члены команды определяют вектор развития. Некоторые избегают серьезные вопросы и фокусируются на мелочах. В данный период может происходить «болезненный» рост команды, в котором менеджер команды должен разрешать спорные ситуации, помогать участникам и подавать пример профессиональным поведением;

3) Нормирующая. В данном этапе команда приступает к согласованию общего плана работ. Далеко не все идеи будут реализованы, следовательно, членам команды придется отказываться от своих инициатив для командной работы;

4) Исполнительная. Команда уже полностью функционирует и постепенно начинает добиваться результатов, может саморегулироваться для выполнения работы гладко и без неуместных споров;

5) Расставание. Совместная деятельность команды прекращается, проект завершен, все цели и задачи достигнуты.

Scrum использует поэтапный и итеративный подход для оптимизации

предсказуемости и контроля рисков. Оптимизация процесса создания программных продуктов идет путем оптимизации процесса командной работы, которая строится согласно определенному алгоритму [5]:

1) Увеличение прозрачности процесса командной работы, это необходимо для того, чтобы каждый член команды понимал как его работы влияет на весь процесс;
2) Упрощение применения принципов бережливого производства;
3) Адаптация и проверка новых методов работы команды;
4) Сосредоточенность на результате;
5) Локальная оптимизация даст меньше преимуществ, чем оптимизация всего процесса, следовательно, нужно анализировать весь процесс и ориентироваться на общие проблемы.

Менеджер является ключевым лицом при выполнении проекта, он отвечает за подбор членов команды и общий настрой при работе, он же отвечает за соблюдение принципов scrum. Также менеджер ведет общение с заказчиком, отвечает за обратную связь с клиентом, ведет ежедневные собрания.

Хоть scrum и направлен на оптимизацию процесса работы команды, сам процесс разработки, который занимает большую часть времени, является творческим. Это в свою очередь означает, что рабочая рутина приведет к выгоранию разработчиков. Чтобы этого не допустить, необходимо больше времени уделять самой команде и настроениям внутри нее.

Чтобы было больше времени на микро-менеджмент внутри команды, необходимо провести оптимизации рутинного процесса.

При слаженной работе команды самым рутинным процессом является процесс распределения задач. Оптимизация данного процесса позволит выделить освободившееся время команде.

Постановка задачи

При использовании гибких методологий разработки, в частности Scrum, фронт работ определяется и изменяется во время собраний в начале спринта [6]. Затем задачи, обеспечивающие выполнение формализованных требований заказчика, распределяются среди членов команды.

В настоящее время похожей задачей занимаются немногие компании. Самым ярким примером является компания Microsoft. В 2020 году данная организация решила задействовать алгоритмы машинного обучения в совместной работе с экспертами в области безопасности при выявлении ошибок и уязвимостей в программном обеспечении. Основной задачей разработанной системы является отсеивание ложных срабатываний, тем самым исключить пропуск инженером критической уязвимость в релизную версию разрабатываемого продукта, будь то обновление операционной системы или программы [7]. Система занимается анализом поступающих данных и отчетов разработчиков. Однако, в рамках данной статьи рассматривается процесс распределения задач в команде, в некоторых ситуациях, отделе компании.

Процесс распределения задач менеджером на начальных стадиях развития команды, предлагается ускорить за счет применения формализованных методов оптимизации.

При постановке оптимизационной задачи самым времязатратным этапом является определение типа задач. Необходимо повторно изучить составленное техническое задание членом команды, отнести его к определенному типу задач и отдать разработчику, который за самое короткое время выдаст самый приемлемый результат.

Сокращение времени выполнения этого этапа предлагается реализовать с использованием классификатора текстов технических заданий, обученном на массиве тестовых заданий различных компаний, которые применяются для принятия решения управленческим персоналом при формировании команды по созданию IT-продуктов.

Таким образом, для оптимизации процесса распределения задач при управлении командной деятельностью в IT-компаниях необходимо рассмотреть подходы к решению следующих задач:

– построение оптимизационной модели;

– формирование исходных данных для формализованного описания экстремальных и граничных требований в оптимизационной модели;

– выбор метода машинного обучения классификатора типов задач по текстовой информации, используемого при определении параметров формализованного описания.

Для решения первой задачи предлагается использовать методы многоальтернативной оптимизации [8], второй – экспертного оценивания [9], третьей – методы машинного обучения нейронных сетей.

Выбор нейронной сети проведем среди следующих: сверточная нейронная сеть (CNN), рекуррентная нейронная сеть с долгой краткосрочной памятью (LSTM), рекуррентные нейронные сети с управляемыми рекуррентными блоками (GRU), двунаправленные рекуррентные нейронные сети (BRNN) [10].

Результаты

Принята следующая этапность применения оптимизационного подхода.

1. Введение альтернативных переменных, характеризующих распределение управляющим персоналом задач между членами команды

$$x_{mn} = \begin{cases} 1, \text{если } m - \text{я задача выполняется } n - \text{м членом команды}, \\ 0, \text{в противном случае}, m = \overline{1,M}, n = \overline{1,N}, \end{cases} \quad (1)$$

где $m = \overline{1,M}$ – нумерационное множество задач, входящих в проект;

$n = \overline{1,N}$ – нумерационное множество членом команды – исполнителей проекта.

2. Формализованное описание экстремального требования.

Необходимо максимизировать интегральную оценку эффективности командной деятельности по всем задачам проекта

$$\sum_{m=1}^{M}\sum_{n=1}^{N} a_{mn}x_{mn} \to \max_{x_{mn}}, \quad (2)$$

где $0 \le a_{mn} \le 1$ – коэффициент эффективности выполнения m-й задачи n-м членом команды.

3. Формализованное описание граничных требований.

Эти требования определяют выполнение одним работником одной задачи

$$\sum_{m=1}^{M} x_{mn} = 1, n = \overline{1,N}, \quad (3)$$

$$\sum_{n=1}^{N} x_{mn} = 1, m = \overline{1,M}. \quad (4)$$

4. Объединение (1) – (4) в единую модель многоальтернативной оптимизации [8]

$$\begin{gathered}\sum_{m=1}^{M}\sum_{n=1}^{N} a_{mn}x_{mn} \to \max_{x_{mn}},\\ \sum_{m=1}^{M} x_{mn} = 1, n = \overline{1,N},\\ \sum_{n=1}^{N} x_{mn} = 1, m = \overline{1,M},\\ x_{mn} = \begin{cases}1,\\0\end{cases}, \; m = \overline{1,M}, n = \overline{1,N}.\end{gathered} \tag{5}$$

5. Формирование информации, необходимой для определения коэффициентов эффективности a_{mn}.

Источником такой информации являются результаты выполнения тестовых заданий работниками при их приеме на работу в IT-компанию. Для привязки тестовых заданий к типам $m = \overline{1,M}$ задач, которые выполняются членами команды, построен классификатор текстов тестовых заданий различных компаний, имеющихся в открытом доступе, по принадлежности к нумерационному множеству $m = \overline{1,M}$. Проведено сравнение перечисленных выше методов машинного обучения нейронных сетей классификации, реализованных на языке программирования Python 3.8.

Перед началом обучения и работы с нейронными сетями данные подготавливались в несколько шагов:

1. Составлялся общий массив из всех слов в нормальной форме (единственное число, именительный падеж, союзы, предлоги и частицы удаляются). Также во время этой операции определяется размер самого длинного условия задачи;

2. Создавалось два массива, первый с задачами, второй с названием категорий (индекс списка задач категории совпадает с индексом названия категории во втором массиве);

3. Первый массив является многомерным. При этом первый элемент заполняется массивами условий задач первой категории. Массив условия задачи состоит из индексов слов из словаря, если длинна условия меньше определенной максимальной длинны, то до конца она заполняется нулями, по такому принципу формируются все данные.

После всех преобразований в библиотеку Tensorflow отправляется два массива: первый с задачами, второй с названиями категорий. Процент проверочных данных устанавливается на уровне 20%. В рамках поставленной задачи на выход от нейронной сети идет информация о потерях и точности обучения, в последствии, при усовершенствовании модели на выходе будет массив равный размеру массива категорий с оценкой от 0 до 1 по всем категориям и тот элемент, который будет ближе всех к 1 является типом поданной на вход для определения задачи.

Сравнение алгоритмов (таблица 1) идет по-разному числу эпох – 5, 10, 15, функции активации рассматривались следующие – гиперболический тангенс (Tanh), сигмоидная (Sigmoid), ReLu. Реализованы алгоритмы с помощью библиотеки Tensorflow. Количество слоев – 2.

Таблица 1. Эффективность обучения приведенных типов сетей

Тип нейронной сети	Время обучения (секунды)	Количество эпох	Количество нейронов по слоям		Точность (%)
			1	2	
CNN	5.92 / 5.8 / 5.63	5	10	5	70.5 / 0 / 67.2
	6.2 / 6.1 / 5.6		64	32	68.5 / 70.65 / 68.8
	11 / 10.8 / 10.72	10	10	5	69.5 / 0 / 70.1
	10.9 / 10.82 / 10.7		64	32	70.65 / 70.34 / 71.48
	18.6 / 17.8 / 18.56	**15**	10	5	0 / 71.1 / 71.17
	19.1 / 15.6 / **15.7**		**64**	**32**	70.23 / 71.59 / **73.36**
LSTM	19 / 19 / 18	5	10	5	0 / 0 / 0
	24 / 24 / 25		64	32	70.96 / 72.2 / 70.96
	29 / 28 / 34	10	10	5	0 / 0 / 0
	44 / 44 / 44		64	32	70.86 / 72.84 / 0
	41 / 48 / 38	15	10	5	70.1 / 0 / 70.03
	64 / 64 / 64		64	32	70.75 / 71.8 / 0
GRU	14 / 14 / 14	5	10	5	68.033 / 68.033 / 0
	18 / 20 / 20		64	32	69.61 / 70.55 / 68.78
	24 / 34 / 24	10	10	5	71.4 / 70.3 / 0
	36 / 33 / 34		64	32	67.84 / 72.73 / 72.6
	34 / 34 / **34**	**15**	**10**	**5**	70.75 / 71.9 / **74.82**
	50 / 48 / 49		64	32	71.8 / 71.8 / 71.9
BRNN	7 / 7 / 7	5	10	5	68.5 / 73.4 / 70.1
	8 / 8 / 8		64	32	71.28 / 72.42 / 72.63
	13 / 17 / 12	10	10	5	65.5 / 71.5 / 70.8
	13 / 17 / 12		64	32	72 / 70.23 / 70.65
	17 / 17 / **17**	**15**	**10**	**5**	71.6 / 71.2 / **73.8**
	17 / 17 / 17		64	32	72.11 / 71.07 / 72.6

Из Таблицы 1, следует вывод, что самыми эффективными алгоритмами машинного обучения на малых объемах данных (140 документов на 6 категорий поставленных в команде задач с максимальным размером задачи в 122 слова) являются: сверточная нейронная сеть (CNN) [11, 12], рекуррентные нейронные сети с механизмом вентилей (GRU) и двунаправленная рекуррентная нейронная сеть (BRNN). Самым стабильным алгоритмом оказался BRNN, CNN и GRU в некоторых ситуациях не могли обучиться из-за нехватки памяти. У всех представленных алгоритмов функцией активации являются ReLu.

Лучший результат, в соотношении точности и времени обучения получился с использованием сверточной нейронной сети, но в силу того, что она отработала

нестабильно в различных условиях, нельзя быть уверенным, что она не подведет в важный момент после внедрения. Следовательно, самым лучшим вариантом алгоритма для классификации задач является двунаправленная рекуррентная нейронная сеть.

Для повышения точности определения необходима увеличить базу обучающего материала и осуществить подбор материала так, чтобы минимальный размер условия задачи после приведения к нужному виду был как минимум 120 слов.

6. Вычисление коэффициентов эффективности.

С использованием обученного классификатора для каждого *n*-го члена команды по текстам выполненных им тестовых заданий определяются соответствующие типы задач и подсчитывается величина

$$a_{mn} = \frac{r_{mn}}{r_n},$$

где r_{mn} −количество тестовых заданий *n*-го члена команды, соответствующих по результатам классификации и экспертной оценки [8] *m*-му типу задачи;

r_n − общее количество тестовых заданий, выполненных *n*-м членом команды.

7. Определение оптимального распределения задач между членами команды в соответствии с оптимизационной моделью (5) на основе алгоритма многоальтернативной оптимизации [8]

$$x^*_{mn}, m = \overline{1,M}, n = \overline{1,N}.$$

Значение $x^*_{mn} = 1$ означает, что менеджеру следует поручить выполнение *n*-й задачи *m*-му члену команды.

Заключение

Повышение эффективности управления командной деятельностью в IT-компаниях на основе концепции Agile в большей мере нацелено на совершенствовании механизмов взаимодействия с заказчиком. Однако до сих пор не предложено автоматизировать механизмы выполнения рутинных процессов путем использования оптимизационного подхода и алгоритмов машинного обучения.

Автоматизация процесса распределения задач между членами команды позволяет снизить нагрузку на менеджера. Тем самым у него появляется возможность сфокусироваться на взаимодействии внутри команды в период каждого спринта и более точном формировании технических заданий членам команды в соответствии с требованиями заказчика.

Таким образом, оптимизация внутренних рутинных процессов приводит к улучшению работы команды, повышения качества разработки и увеличению скорости выполнения запланированных работ для достижения поставленных целей.

ЛИТЕРАТУРА

1. *5 моделей эффективного командного взаимодействия.* Доступно по: https://habr.com/ru/company/hygger/blog/418001/ (дата обращения 12.01.2021).
2. *Гибкая процессная методология Agile.* Доступно по: https://intuit.ru/studies/courses/3590/832/info (дата обращения 20.01.2021).
3. Tuckman, B.: Developmental Sequence in Small Groups. Psychological Bulletin, 1965:384-399.
4. Tuckman, B., Jensen, M.: *Stages of Small Group Development. Group and Organizational Studies*, 1977:419-427.
5. Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D.

Jackel: Backpropagation Applied to Handwritten Zip Code Recognition, *Neural Computation*, 1989;1(4):541-551.
6. Scrum Mastery: *5 Steps to Improve Team Process*. Доступно по: https://www.agilesocks.com/scrum-mastery-5-steps-improve-team-process/ (дата обращения 20.01.2021).
7. *Secure the software development lifecycle with machine learning.* Доступно по: https://www.microsoft.com/security/blog/2020/04/16/secure-software-development-lifecycle-machine-learning/ (дата обращения 10.01.2021).
8. Львович Я.Е. *Многоальтернативная оптимизация: теория и приложения.* – Воронеж: Издательский дом «Кварта». 2006.
9. Львович Я.Е. *Принятие решений в экспертно-виртуальной среде.* Я.Е.Львович, И.Я.Львович. – Воронеж: ИПЦ «Научная книга», 2010.
10. Hastie T., Tibshirani R., Friedman J. *The Elements of Statistical Learning.* Springer, 2001.
11. Zhang, X. Character-level convolutional networks for text classification. Xiang Zhang, Junbo Zhao, Yann LeCun. *In Advances in Neural Information Processing Systems*. 2015:649-657.
12. Kim, Y. *Convolutional neural networks for sentence classification*. Yoon Kim. IEMNLP. – 2014:1746-1751.

REFERENCES

1. *5 models of effective team interaction.* Available from: https://habr.com/ru/company/hygger/blog/418001/ (accessed 12.01.2021).
2. *Flexible Agile Process Methodology*. Available from: https://intuit.ru/studies/courses/3590/832/info (accessed 20.01.2021).
3. Tuckman, B.: Developmental Sequence in Small Groups. *Psychological Bulletin*, 1965:384-399.
4. Tuckman, B., Jensen, M.: *Stages of Small Group Development*. *Group and Organizational Studies*, 1977:419-427.
5. Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel. Backpropagation Applied to Handwritten Zip Code Recognition, *Neural Computation*, 1(4)1989:541-551.
6. Scrum Mastery: 5 Steps to Improve Team Process. Available at: https://www.agilesocks.com/scrum-mastery-5-steps-improve-team-process/ (accessed 20.01.2021).
7. Secure the software development lifecycle with machine learning. Available at: https://www.microsoft.com/security/blog/2020/04/16/secure-software-development-lifecycle-machine-learning/ (accessed 10.01.2021).
8. Lvovich Y.E. Multiple Optimization: Theory and Applications. Voronezh: Publishing house «Kvarta». 2006.
9. Lvovich Y.E. *Decision making in expert-virtual environment*. Y.E. Lvovich, I.Y. Lvovich, Voronezh: PPC «Scientific book», 2010.
10. Hastie T., Tibshirani R., Friedman J. *The Elements of Statistical Learning*. Springer, 2001.
11. Zhang, X. Character-level convolutional networks for text classification. Xiang Zhang, Junbo Zhao, Yann LeCun. *In Advances in Neural Information Processing Systems.* 2015:649-657.
12. Kim, Y. *Convolutional neural networks for sentence classification.* Yoon Kim. IEMNLP. – 2014:1746-1751.

ИНФОРМАЦИЯ ОБ АВТОРАХ / INFORMATION ABOUT THE AUTHORS

Корчагин Сергей Геннадьевич, аспирант, кафедра систем автоматизированного проектирования и информационных систем, Федеральное государственное бюджетное образовательное учреждение высшего образования "Воронежский государственный технический университет", Воронеж, Российская федерация.
email: pwkgv138@yandex.ru

Sergey G. Korchagin, Phd Student, Computer-Aided Design And Information Systems Department, Federal State Budgetary Educational Institution Of Higher Education "Voronezh State Technical University", Voronezh, Russian Federation.

Львович Яков Евсеевич, заведующий кафедрой, Федеральное государственное бюджетное образовательное учреждение высшего образования "Воронежский государственный технический университет", Воронеж, Российская Федерация.
e-mail: office@vivt.ru

Yakov E. Lvovich, Head Of The Chair, Federal State Budgetary Educational Institution Of Higher Education "Voronezh State Technical University", Voronezh, Russian Federation.