УДК 004.05+005:004.8
DOI: http://dx.doi.org/10.20535/0203-3771332017100728

А. А. Чернюк[1], *бакалавр*

ДОСЛІДЖЕННЯ ВПЛИВУ ЯКОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ АВТОЗАПРАВНИХ КОМПЛЕКСІВ НА ЇХ ЕФЕКТИВНІСТЬ

En This project is devoted to the impact of the software filling stations quality on their performance. Software quality models were analyzed and their principles used for the development of software quality model of filling stations are presented. The evaluation system of (KPI) station efficiency is considered and the choice of key performance indicators for the task is justified. The proposed forecasting system based on neural network shows how to design, develop, and train the neural network, namely the implementation of learning algorithms.

Ru Данная работа посвящена исследованию влияния качества программного обеспечения (ПО) автозаправочных комплексов на их эффективность. Проанализированы модели качества ПО и приводятся их принципы построения, которые применены для разработки модели качества ПО автозаправочных комплексов. Рассматривается система оценки эффективности (KPI) станции и обосновывается выбор ключевых показателей эффективности для поставленной задачи. Предложенная система прогнозирования на основе нейронной сети, показывается, как проектировать, разрабатывать, обучать нейронную сеть, а именно реализаций алгоритмов обучения.

Вступ

Компанії, які займаються роздрібною торгівлею, завжди зацікавленні у ефективності свого бізнесу, виконанні поставлених цілей та прогнозуванні прибутковості підприємства. Автозаправні комплекси не є виключенням. Оскільки на кожному етапі бізнес-процесу для їх обслуговування вбудовується програмне забезпечення(ПЗ), починаючи від низькорівневого для контролю наливних систем АЗС і закінчуючи центральним офісом для моніторингу статистичних даних, якість ПЗ є актуальним питанням. Вихо-

[1] *НТТУ «Київський політехнічний інститут ім. Ігоря Сікорського», кафедра автоматизації експериментальних досліджень*

дячи із цього, можна припустити, що якість ПЗ впливає на ефективність АЗС. Але як отримати кількісні значення цього впливу? Саме це питання є основою дослідження.

Постановка завдання

Метою дослідження є отримання показників якості програмного забезпечення АЗС та, використовуючи їх, прогнозування основних показників діяльності АЗС. Об'єктом дослідження є ключові показники ефективності АЗС. Предметом дослідження є система прогнозування показників ефективності АЗС за значеннями показників її програмного забезпечення. Завдання дослідження: розробити модель якості ПЗ для АЗС; вибрати основні показники ефективності АЗС; розробити систему прогнозування показників KPI, використовуючи показники, що визначені моделлю якості (рис. 1).

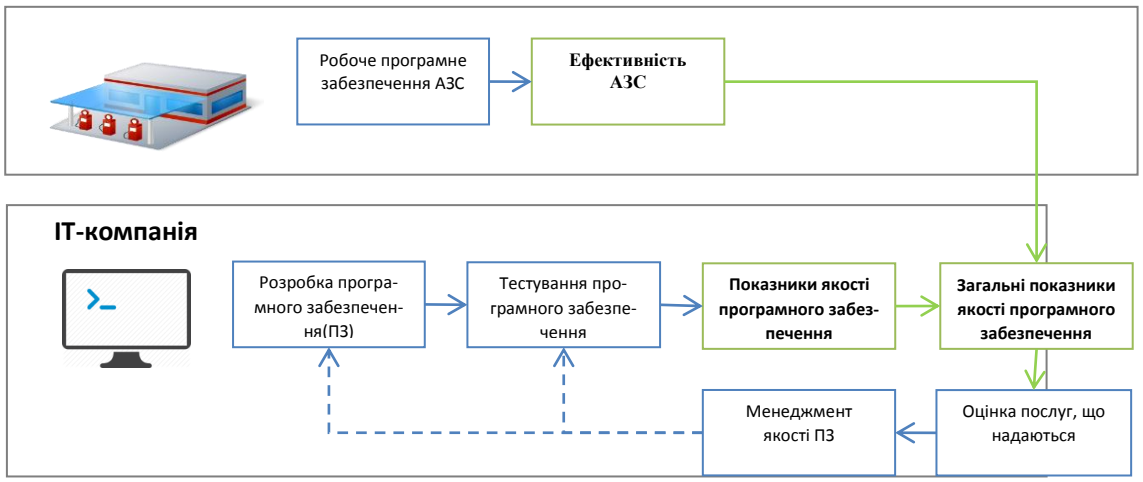

Рис. 1. Відношення між показниками ефективності АЗС і показниками якості її програмного забезпечення

Особливості побудови моделі якості ПЗ

Користувачі програмного забезпечення відчувають потреби у створенні моделей якості програмного забезпечення для оцінки якості як якісно, так і кількісно. Моделі якості, які є на даний час, у більшості випадків є ієрархічними моделями на основі критеріїв якості і пов'язаних із ними показників (метрик). Всі моделі якості можуть бути розділені на три категорії відповідно до методів, на основі яких вони були створені. До першого виду можна віднести теоретичні моделі, засновані на гіпотезі відносин між змінними якості. До другого виду відносяться моделі «управління даними», що базуються на статистичному аналізі. І нарешті комбінована модель, у якій інтуїція дослідника використовується для визначення потріб-

ного виду моделі, а аналіз даних використовується для визначення констант моделі якості [4].

Основні вимоги і принципи побудови моделі якості ПЗ

Основним завданням побудови моделі якості є формування, із урахуванням призначення, особливостей, умов експлуатації, ступеня важливості окремих вимог тощо.

Модель якості для ПЗ, як правило, будується на базі міжнародних стандартів, які регламентують показники якості. Однак для оцінки якості конкретного ПЗ регламентовані стандартами характеристики не завжди підходять у повній мірі. Для конкретного розроблюваного ПЗ необхідно, виходячи із його функ-ціонального призначення, особливостей, ступеня важливості окремих вимог, сформувати повний перелік актуальних характеристик якості.

Аналіз показує, що практично всі існуючі на сьогоднішній день підходи і методи побудови моделі якості ПЗ, засновані на формуванні ієрархічної структури характеристик. На верхньому рівні розташовані характеристики якості, які деталізуються показниками більш низьких рівнів до тих пір, поки декомпозиція не призводить до атомарних і вимірних атрибутів. Відмінності ж полягають у пропонованому числі рівнів ієрархії (від 2 і більше), а також у самих характеристиках верхнього рівня ієрархії, багато з яких все ж збігаються.

Доцільність ієрархічної структури для моделі якості ПЗ пояснюється, по-перше, тим, що багаторівнева структура показників якості дозволяє системно описати вимоги до програмного засобу, дозволяючи зацікавленим сторонам задати властивості (характеристики) програмного продукту, які вони хочуть бачити. По-друге, більшість базових показників якості, а саме функціональна відповідність, надійність, безпека тощо, не можуть бути безпосередньо виміряні й оцінені. Для отримання оцінок цих показників вони можуть бути представлені сукупністю більш вузьких характеристик нижнього рівня (оціночні показники), які в свою чергу також можуть бути деталізовані. Така деталізація здійснюється доти, доки характеристики нижнього рівня ієрархії будуть легко оцінювані і забезпечать отримання об’єктивних кількісних оцінок. Характеристики нижнього рівня, виражені у фізичних чи відносних одиницях, називають одиничними показниками якості.

Побудова моделі якості ПЗ проводиться шляхом детального і послідовного опису зверху вниз багаторівневої структури показників від характеристик верхнього рівня ієрархії до оцінних елементів (одиничних показників). У такому разі оцінювальний елемент повинен забезпечити безпосереднє визначення наявності тієї чи іншої властивості в ПЗ.

Побудова моделі якості йде зверху вниз від характеристик до оцінювальних елементів, а оцінка досягнутих показників якості йде у зворотному напрямку: від оцінки одиничних оцінювальних елементів до агрегованої оцінки вищеназваних показників якості у моделі якості ПЗ.

На сьогоднішній день не існує загальноприйнятої методики побудови моделі якості ПЗ, що дозволяє зводити чинники якості до кінцевого набору кількісних оціночних показників, значення яких були б легко і об'єктивно оцінювані [5].

Методика побудови моделі якості ПЗ

Побудову моделі якості ПЗ слід проводити з урахуванням його призначення, типу (класу), стадії життєвого циклу, на якому вона буде застосовуватися.

На першому етапі за основу слід брати всю базову номенклатуру характеристик, підхарактеристик і атрибутів якості програмного продукту по *ISO* 25010. Їх описи бажано попередньо впорядкувати за пріоритетами з урахуванням призначення і сфери застосування конкретного проекту програмного засобу. Далі необхідно виділити і ранжувати за пріоритетами споживачів, яким необхідні певні показники якості проекту програмного засобу з урахуванням їх професійних інтересів. Підготовка вихідних даних завершується виділенням номенклатури базових, пріоритетних показників якості, що визначають функціональну придатність програмного засобу для певних споживачів.

На другому етапі, після фіксування вихідних даних, необхідно провести ранжування характеристик і підхарактеристик для конкретного проекту. Далі цими фахівцями для кожного з відібраних показників повинна бути встановлена і погоджена метрика і шкала оцінок підхарактеристик і їх атрибутів. Вибрані значення характеристик якості і їх атрибутів повинні бути попередньо перевірені розробниками на їх реалізованість з урахуванням доступних ресурсів конкретного проекту і при необхідності відкориговані.

У номенклатурі показників якості слід вказувати пріоритетність кожного із показників. Найвищий пріоритет слід інтерпретувати як обов'язкове виконання розробником відповідної вимоги до зазначеної властивості або атрибуту якості [5].

Алгоритм розв'язування задачі за допомогою багатошарового персептрона

Побудова багатошарового персептрону пов'язана із необхідністю вибору його параметрів. Найчастіше вибір значень ваг і порогів вимагає навчання, тобто покрокових змін вагових коефіцієнтів і порогових рівнів.

Загальний алгоритм розв'язання:

1. Визначити зміст, що вкладається у компоненти вхідного вектора *x*. Вхідний вектор повинен містити формалізовану умову задачі, тобто всю інформацію, необхідну для отримання відповіді.
2. Вибрати вихідний вектор таким чином, щоб його компоненти мали повну відповідь поставленого завдання.
3. Вибрати вид нелінійності в нейронах (функцію активації). У такому разі бажано врахувати специфіку завдання, тому що вдалий вибір скоротить час навчання.
4. Вибрати число шарів і нейронів.
5. Задати діапазон зміни входів, виходів, ваг і порогових рівнів, враховуючи безліч значень обраної функції активації.
6. Присвоїти початкові значення ваговим коефіцієнтам, граничним рівням і додатковим параметрам (наприклад, крутизні функції активації, якщо вона буде налаштовуватися за навчання). Початкові значення не повинні бути великими, щоб нейрони не опинилися у насиченні (на горизонтальній ділянці функції активації), інакше навчання буде дуже повільним. Початкові значення не повинні бути і занадто малими, щоб виходи більшої частини нейронів були рівні нулю, інакше навчання також сповільниться.
7. Провести навчання, тобто підібрати параметри мережі так, щоб задача вирішувалася найкращим чином. Після закінчення навчання мережа готова вирішити завдання того типу, яким вона навчена.
8. Подати на вхід мережі умови задачі у вигляді вектора *x*. Розрахувати вихідний вектор *y*, який і дасть формалізований розв'язок задачі [6].

Вибір кількості нейронів і шарів

Немає чітко визначеної процедури для вибору кількості нейронів і кількості шарів у мережі. Чим більша кількість нейронів і шарів, тим ширші можливості мережі, тим повільніше вона навчається і працює, і тим більше нелінійною може бути залежність вхід вихід [6].

Кількість нейронів і шарів пов'язано:

- зі складністю завдання;
- із кількістю даних для навчання;
- із необхідною кількістю входів і виходів мережі;
- із наявними ресурсами: пам'яттю і швидкодією машини, на якій моделюється мережа.

Були спроби записати емпіричні формули для числа шарів і нейронів, але застосовність формул виявилася дуже обмеженою. Якщо у мережі занадто мало нейронів або шарів:

- мережа не навчиться і помилка під час роботи мережі залишиться великою;
- на виході мережі не будуть передаватися різкі коливання функції, що апроксимується $y(x)$.

Перевищення необхідної кількості нейронів теж заважає роботі мережі. Якщо нейронів або шарів занадто багато [11]:

- швидкодія буде низька, а пам'яті буде потрібно багато;
- мережа перевчиться: вихідний вектор буде передавати незначні і несуттєві деталі у досліджувану залежність $y(x)$, наприклад, шум або помилкові дані;
- залежність виходу від входу виявиться різко нелінійною: вихідний вектор буде суттєво, але і непередбачувано змінюватися за малу зміну вхідного вектора x;
- мережа буде нездатна до узагальнення: у галузі, де немає або мало відомих точок функції $y(x)$, вихідний вектор буде випадковий і непередбачуваний, що не буде адекватним вирішенням задачі.

Правила навчання

Існують три парадигми навчання: за допомогою вчителя, без вчителя (самонавчання) і змішана. У першому випадку нейронна мережа має правильні відповіді (виходи мережі) на кожен вхідний приклад. Ваги налаштовуються так, щоб мережа виробляла відповіді як можна більш близькі до відомих правильних відповідей. Посилений варіант навчання за допомогою вчителя припускає, що відома тільки критична оцінка правильності виходу нейронної мережі, але не самі правильні установки виходів. Навчання без вчителя не вимагає знання правильних відповідей на кожний приклад навчальної вибірки.

Відомі 4 основних типи правил навчання [7]:

- корекція за помилкою;
- машина Больцмана;
- правило Хебба;
- навчання методом змагання.

Мережі, які реалізують парадигму самонавчання (без вчителя), призначені, як правило, для аналізу внутрішньої латентної структури вхідної інформації і вирішують завдання автоматичної класифікації, кластеризації, факторного аналізу, компресії даних. Навчальний алгоритм підлаштовує ваги мережі так, щоб пред'явлення досить близьких вхідних векторів давало однакові виходи. Процес навчання виділяє статистичні властивості навчальної множини і групує подібні вектори у класи [7].

Функція активації (активаційна функція, функція збудження, характеристична функція) – нелінійна функція, що обчислює вихідний сигнал

формального нейрона (*OUT)*. Зазвичай приймає у якості аргументу сигнал, що отримуємо на виході вхідного суматора (*NET*). Як активаційна часто використовуються сигмоїдальна функція (рис. 2).

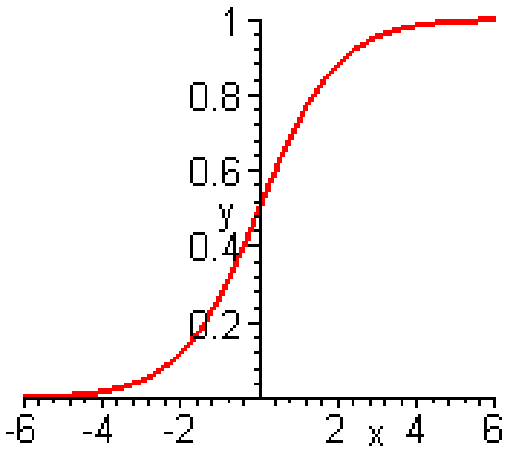

Рис. 2. Сигмоїдальна функція

Сигмоїдальна функція (сигмоїд, сигмоїда) – монотонно зростаюча всюди, диференційована, *S*-образна нелінійна функція із насиченням, яку дуже зручно використовувати у формальному нейроні як функцію активації. Сигмоїд дозволяє підсилювати слабкі сигнали і не насичуватися потужним сигналом.

Серед однопараметричних найбільш поширені сигмоїдальні функції наступних видів:

– логістична функція;
– гіперболічний тангенс;
– раціональна сигмоїда.

Застосовується дуже часто для багатошарових перцептронів та інших мереж із безперервними сигналами. Гладкість, безперервність функції – важливі позитивні якості. Безперервність першої похідної дозволяє навчати мережу градієнтними методами (наприклад, метод зворотного поширення помилки).

Значення похідної легко виражається через саму функцію. Швидкий розрахунок похідної прискорює навчання [7].

Висновок

Запропонована система прогнозування допоможе оцінювати ефективність роботи автозаправних станцій у контексті якості придбаного ПЗ. Це дозволить оцінювати ризики, слідкувати за виконанням поставлених цілей та співпрацювати із постачальником ПЗ для підвищення якості отриманого продукту, що у результаті призведе до підвищення ефективності АЗС.

Список використаної літератури

1. Firesmith D. G. Common concepts underlying safety, security, and survivability engineering, Technical Note CMU/SEI-2003-TN-033, Carnegie Mellon Software Engineering Institute. 2003
2. Рассел, Норвиг: Искусственный интеллект. Современный подход. Вильямс, 2015 г.–1408 с.
3. Boehm B.W., Brown J.R., Kaspar H., Lipow M., MacLeod G.J., Merritt M.J.. Characteristics of Software Quality, TRW Series of Software Technology, Amsterdam, North Holland, 1978. 166 p.
4. Заенцев І. В. Нейронні мережі: основні моделі І. В. Заенцев.- Воронеж: Изд-во Воронезького госуд. ун-ту, 1999. - 76 с.
5. Круглов В. В., Борисов В. В. Искусственные нейронные сети. Теория и практика.М.: Горячая линия-Телеком, 2002. - 382 с.
6. Рутковська Д. Нейронні мережі, генетичні алгоритми і нечіткі системи Д. Рутковська, М. Піліньскій, Л. Рутковський.- М .: Гаряча лінія Телеком, 2006. - 452 с.
7. Воронівка Г. К. Генетичні алгоритми, штучні нейронні мережі і проблеми віртуальної реальності Г. К. Вороновський, К. В. Махота, С. Н. шів: ОСНОВА, 1997. - 112 с.