програми мають до нас одну вимогу облікові записи вводити латинськими літерами. Всі ці процедури робляться задля того щоби уникнути серйозних проблем, що виникають при завантаженні ПК. Можуть усі змінити пароль за допомогою інших програм DebPloitta GetAdm2. Але якщо допустити видалення SAMце стане фатальною помилкою для користувачаоскільки може накритися уся операційна система і увійти в неї стане неможливо. Або остання річ така як MSV1_O.DLL (тобто заміна звичайної ліцензованої версії на патчінгову версію у якій відключений будь-який тип захисту у тому числі і відсутність працездатності паролю. Варто лише виправити код такої DLLбібліотеки з метою зміни переходів змістовних на беззмістовні [2, с. 98].

В якості висновку можна лише додати для того щоб захистити інформацію потрібно встановити потужний антивірус.

Список використаних джерел:

1. Храмцов П.Б., Брик С.А., Русак А.М., Сурин А.И.– Основы WEB-технологий. – М.: ИТУИТ.РУ, 2003. – 512 с.
2. Роджерс Д.– Программирование на Microsoft JScript.NET. «Вильямс», 2002. – 352 с.
3. Старыгин А. XML: разработка Web-приложений. – СПб. : «БХВ-Петербург», 2003. – 585 с.
4. Троелсен Э.. C# и платфома.NET. Библиотека программиста. – СПб.: «Питер», 2007. – 796 с.

Кулибаба П.О.
магістр;
Якименко Д.О.
магістр,
Черкаський державний технологічний університет

ПЕРЕВІРКА ПРАВОПИСУ ВВЕДЕНОГО ТЕКСТУ НА ОСНОВІ МОДЕЛІ NOISY CHANNEL

У наш час проблема орфографічної грамотності набуває все більшої актуальності. Адже без знання орфографічних правил неможливо написати лист, який-небудь документ, статтю або просто записку близьким. Для перевірки орфографії все більше використовують комп’ютери. Існує безліч онлайн сервісів та комп’ютерних програм для перевірки орфографії. Тому задача побудови простого і швидкого алгоритму пошуку помилок у тексті є актуальною.

Мета роботи. Проаналізувати роботу алгоритму для перевірки орфографії в тексті основаного на моделі Noisy Channel. Спростити та прискорити роботу алгоритму.

Ми хочемо написати деяке слово Z із m букв, а виходить, що ми написали слово X із n букв. Виявляється, що правильне слово Z викривилось до

неправильного слова X. Потрібно віднайти таке слово, яке ми найбільш ймовірно мали на увазі при написані слова X. Потрібно розробити алгоритм для перевірки орфографії в тексті, основаного на моделі Noisy Channel, яка дозволяє автоматично перевіряти текст на помилки. Потрібно побудувати математичну модель і навчити її на основі бази слів з частотами Пітера Норвіга [1].

Вирішення задачі. Запишемо постановку задачі математично:

$$\hat{Z} = \arg\max_{Z \in V} P(Z|X), \quad (1)$$

де V – список всіх слів природної мови.

Використаємо теорему Баєса:

$$\hat{Z} = \arg\max_{Z \in V} P(Z|X) = \arg\max_{Z \in V} \frac{P(X|W) \cdot P(W)}{P(X)} = \arg\max_{Z \in V} P(X|Z) \cdot P(Z), \quad (2)$$

де P(Z) – апріорна ймовірність слова Z в мові, цей член являє собою статистичну модель природної мови (ми будемо використовувати модель unigram), значення цього члена легко обчислюється з бази слів мови;

P(X|Z) – ймовірність того, що правильне слово Z було помилково написано, як X (цей член називається channel model). Маючи досить велику базу, яка б мала всі способи помилитися при написані кожного слова мови, то обчислення цього члена не викликало б труднощів, але такої великої бази не існує, тому використаємо базу, яка містить 333333 слова англійської мови, і тільки для 7481 є слова з помилками.

Обчислимо значення ймовірності P(X|Z), для цього потрібно використати відстань Левенштейна – це мінімальна кількість операцій вставки, видалення і заміни, необхідних для перетворення одної послідовності символів (рядків) в іншу.

Для двох слів X і Z ми можемо обчислити список операцій необхідних для перетворення першого слова в друге, позначимо список операції буквою f, тоді ймовірність написати слово X як Z дорівнюватиме ймовірності зробити весь список помилок f за умови, що ми писали саме X, а мали на увазі Z:

$$P(X|Z) = P(f|Z \to X). \quad (3)$$

Спростимо:

– порядок проходження операцій помилки не має значення;

– помилка не буде залежати від того, яке слово ми написали і від того, що мали на увазі.

$$P(X|Z) = P(f|Z \to X) = P(f) = \prod_{i=1}^{t} P(f_i), \quad (4)$$

Для того, щоб обчислити ймовірність помилок, потрібно мати будь-яку базу слів з їх помилковим написанням. Запишемо фінальну формулу:

$$\hat{Z} = \arg\max_{Z \in V} P(Z|X) = \arg\max_{Z \in V} \frac{P(X|W) \cdot P(W)}{P(X)} = \arg\max_{Z \in V} P(X|Z) \cdot P(Z) =$$
$$\arg\max_{Z \in V} P(Z) \cdot \prod_{i=1}^{t} P(f_i) = \arg\max_{Z \in V} \ln P(Z) + \sum_{i=1}^{t_{Z \to X}} \ln P(f_i^{Z \to X}). \quad (5)$$

Якщо у нас є достатній набір текстів, ми можемо обчислити апріорні ймовірності слів у мові, також маючи базу слів з їх помилковим написанням, ми можемо обчислити ймовірності помилок. Цих двох баз достатньо для реалізації моделі [2].

При проході по всій базі слів при arg max ми, жодного разу не зустрінемо слова з нульовою ймовірністю, але при обчисленні операцій редагування для перетворення слова X до слова Z, можуть виникнути такі операції, що не

зустрічалися в нашій базі помилок. Тут нам допоможе перетворення Лапласа – якщо деяка операція корекції f зустрічається в базі n раз, при тому що найбільше помилок в базі m, а типів корекції t (наприклад для заміни, не те скільки разів зустрічається заміна «a на b», а скільки всього унікальних пар «* на *»), то розмита ймовірність виглядає так:

$$P(f) = \frac{n+k}{m+k \cdot t}, \quad (6)$$

де k – коефіцієнт розмиття.

Тоді ймовірність операції, яка жодного разу не зустрічалася в навчальній базі (n = 0) буде дорівнювати:

$$P(f_{new}) = \frac{k}{m+k \cdot t}. \quad (7)$$

Якщо нам потрібні тільки слова не більше ніж в t операцій редагування від поточного слова, то їх довжина відрізняється від поточного не більше ніж на t. Для прискорення роботи алгоритму можна створювати хеш-таблицю, в якій ключами є довжини слів, а значеннями множина слів цієї довжини. Це дозволяє значно скоротити простір пошуку.

Для тестування наведеної вище моделі можна скористатися базою Пітера Норвіга, в якій зберігається 333333 слова з частотами і 7481 слово з помилковими написаннями [3].

Висновки. Було проаналізовано роботу алгоритму для перевірки орфографії в тексті основаного на моделі Noisy Channel. Було навчено та побудовано математичну модель алгоритму. Було запропоновано підхід, за допомогою якого можна віднайти таке слово, яке ми найбільш ймовірно мали на увазі при написані слова X. Даний підхід дозволяє спростити і прискорити роботу алгоритму за допомогою використання хеш-таблиці.

Список використаних джерел

1. Daniel Jurafsky, James H. Martin. Speech and Language Processing, 2nd Edition / Publisher: Prentice Hall. – 2008. – 1024 p.

2. Christopher D. Manning, Hinrich Schütze. Foundations of Statistical Natural Language Processing, 1st Edition / Publisher: The MIT Press. – 1999. – 620 p.

3. Roger Mitton. Spellchecking by computer / Journal of the Simplified Spelling Society, Vol 20, № 1, 1996, pp. 4-11.