STUDIA INFORMATICA 2013
Volume 34 Number 1 (110)

Artur NIEWIAROWSKI, Marek STANUSZEK
Politechnika Krakowska, Instytut Informatyki

MECHANIZM ANALIZY PODOBIEŃSTWA KRÓTKICH FRAGMENTÓW TEKSTÓW, NA BAZIE ODLEGŁOŚCI LEVENSHTEINA[1]

Streszczenie. Artykuł przedstawia implementację mechanizmu typu text mining, bazującego na algorytmie odległości transformacyjnej autorstwa Vladimira Levenshteina[4] , skutecznie wykrywającego podobieństwo wyrazów o różnej długości. Algorytm zastosowano do analizy podobieństwa jednozdaniowych fragmentów tekstów. Opracowany mechanizm cechuje szybkość analizy zdań i łatwość implementacji.

Słowa kluczowe: analiza języka naturalnego, analiza danych tekstowych, text mining, algorytm odległości Levenshteina

MECHANISM OF ANALYSIS OF SIMILARITY SHORT TEXTS, BASED ON THE LEVENSHTEIN DISTANCE

Summary. This paper presents the proposal of text mining mechanism based on Levenshtein Distance Algorithm (LDA)[4], which effectively detect the similarity of different length words. This algorithm for similarity analysis of sentences is used and successfully detects similarities between single sentences. Mechanism is characterized by speed of data analysis and simplify of implementation.

Keywords: Natural Language Processing (NLP), Natural Language Understanding (NLU), Data Mining, Text Mining, Levenshtein Distance Algorithm

1. Wprowadzenie do problemu

Istnieje wiele algorytmów porównujących fragmenty tekstów pod względem podobieństwa, m.in. oparte na modelu grafowm, modelu przestrzeni metrycznej, czy najbardziej popu-

[1] Prezentowane w pracy wyniki badań zrealizowano w ramach tematu nr F-3/105/DS/2012 finansowanego z dotacji na naukę przez Ministerstwo Nauki i Szkolnictwa Wyższego w roku 2012.

larnym modelu wektorowym [1, 2]. Mechanizm zaproponowany w niniejszej pracy bazuje na algorytmie odległości edycyjnej [4]. Mechanizm ten pozwala na obliczenie podobieństwa krótkich fragmentów tekstu, przy zachowaniu zadowalającej precyzji i efektywnie niskiego czasu wykonania. W przeciwieństwie do wymienionych algorytmów, przedstawiony model uwzględnia kolejność występowania wyrazów w zdaniach oraz ich podobieństwo, pomijając metody ważenia, eliminacji i ekstrakcji terminów [3]. Wymienione konkurencyjne metody są nastawione na badanie dużych dokumentów tekstowych i ich implementacja jest bardziej złożona od tej zaprezentowanej w publikacji.

2. Model analizy danych

Weźmy pod uwagę zagadnienie analizy podobieństwa ciągów wyrazów, gdzie pojedynczy wyraz jest ciągiem liter i cyfr wybranego alfabetu[2]. Wyrazy są rozdzielane znakiem spacji i zestawione w danym ciągu w tekst. Długością tekstu jest liczba wyrazów w ciągu (N_T), natomiast długością wyrazu jest liczba liter i cyfr w nim występujących (N). Głównym zadaniem proponowanego w pracy algorytmu jest określenie stopnia podobieństwa dwóch ciągów wyrazów (tekstów N i M) o długościach odpowiednio N_T i M_T.

Model mechanizmu[3] analizy podobieństwa danych tekstowych bazuje na algorytmie odległości edycyjnej Levenshteina [4], opracowanym w roku 1965 przez rosyjskiego uczonego Vladimira Levenshteina. Algorytm ten polega na obliczeniu najmniejszej liczby zmian, niezbędnych do przeprowadzenia na znakach dwóch porównywanych ciągów, dla osiągnięcia ich identyczności[4]. Liczba ta jest miarą odległości Levenshteina. Odległość Levenshteina jest uogólnieniem odległości Hamminga (ang. *Hamming distance*) [6]. W przeciwieństwie do odległości Hamminga, odległość Levenshteina umożliwia porównywanie ciągów o różnej długości, co bezpośrednio przekłada się na liczne zastosowania algorytmu, m.in. w korekcie pisowni, maszynowym tłumaczeniu tekstów czy eksploracji danych tekstowych, a szerzej w mechanizmach wyszukiwarek internetowych, procesorach tekstów itp.

Zasada działania algorytmu polega na porównywaniu znaków mieszczących się na odpowiednich pozycjach w badanych dwóch ciągach (o rozmiarach N i M), a następnie na tej podstawie uzupełnianiu macierzy liczbami o wymiarach odpowiadających długości badanych łańcuchów, według schematu (1). Odległością Levenshteina jest wartość liczbowa **d**(N+1,M+1) macierzy.

[2] Alfabet – zestaw liter, czyli symboli graficznych dźwięków danego języka, ułożony w tradycyjnym porządku. Encyklopedia PWN.

[3] Pojęcie mechanizm w publikacji określa współpracujące ze sobą algorytmy w ramach analizy danych.

[4] Dopuszczalnymi operacjami na ciągach są: dodanie znaku, usunięcie znaku, podmiana znaku na inny.

$$K = \mathop{\mathrm{I}}_{i=1}^{N} \mathop{\mathrm{I}}_{j=1}^{M} \mathbf{d}(i,j) = \min(\mathbf{d}(i-1,j)+1, \mathbf{d}(i,j-1)+1, \mathbf{d}(i-1,j-1)+\beta)$$

$$\begin{cases} \beta = 0 : a(i) \equiv b(j) \\ \beta = 1 : a(i) \neq b(j) \\ \mathbf{d}(i,0) = i \\ \mathbf{d}(0,j) = j \\ \mathbf{d}(0,0) = 0 \end{cases} \tag{1}$$

gdzie:

d – macierz o rozmiarach (N+1, M+1) utworzona z dwóch porównywanych ciągów,

K – odległość Levenshteina,

$\mathop{\mathrm{I}}_{i=1}^{N}$ – symbol oznaczający pętlę iteracyjną dla i = (1, ..., N),

N, M – rozmiary dwóch badanych ciągów,

$d(i,j)$ – (i,j)-ty element macierzy **d,**

min – funkcja zwracająca wartość najmniejszą z podanych,

β – zmienna przybierająca odpowiednio wartości 0 lub 1,

$a(i)$ – i-ty element ciągu znaków a,

$b(j)$ – j-ty element ciągu znaków b.

Przykładową konstrukcję macierzy **d** dla dwóch łańcuchów tekstowych *studium* oraz *studia* przedstawia rysunek 1.

		s	t	u	d	i	u	m
	0	1	2	3	4	5	6	7
s	1	0	1	2	3	4	5	6
t	2	1	0	1	2	3	4	5
u	3	2	1	0	1	2	3	4
d	4	3	2	1	0	1	2	3
i	5	4	3	2	1	0	1	2
a	6	5	4	3	2	1	1	2

		s	t	u	d	i	a
	0	1	2	3	4	5	6
s	1	0	1	2	3	4	5
t	2	1	0	1	2	3	4
u	3	2	1	0	1	2	3
d	4	3	2	1	0	1	2
i	5	4	3	2	1	0	1
a	6	5	4	3	2	1	0

Rys. 1. Macierze **d** utworzone przez algorytm odległości Levenshteina
Fig. 1. Matrixes **d** made by Levenshtein Distance Algorithm

Jak wykażą poniższe badania, algorytm odległości Levenshteina może zostać wykorzystany z powodzeniem również do analizy podobieństwa tekstów (zdań). Wyrazy w tekście będą traktowane jak znaki w oryginalnym podejściu algorytmu Levenshteina. Dodatkowo zostanie zaimplementowana funkcja zwracająca miarę podobieństwa dwóch ciągów przy komparacji wyrazów (zamiast operacji: $a(i) \equiv b(j)$), z uwzględnieniem wartości progowej q, określającej, czy wartość miary podobieństwa dla danego przypadku klasyfikuje go jako rzeczywiste podobieństwo dwóch ciągów.

Miara podobieństwa dwóch ciągów P jest obliczana wg zależności (2):

$$P = 1 - (\frac{K}{K_{max}});\ K_{max} = \max(N, M),\quad \begin{matrix} K \geq 0, M > 0, N > 0 \\ P \in \langle 0,1 \rangle \end{matrix} \tag{2}$$

gdzie: K_{max} – wartość długości najdłuższego z badanych ciągów.

Wartość K_{max} jest liczbą kroków zmieniających jeden ciąg w drugi, dla przypadku w którym ciągi zawierają taki zestaw znaków, dla którego należy wykonać operację podmiany dla wszystkich znaków ciągu krótszego i odpowiednią liczbę operacji dodania znaków w tym ciągu, w celu przeprowadzenia go w ciąg dłuższy[5] (tzn. przypadek, w którym odległość Levenshteina jest równa długości najdłuższego ciągu). Zasada działania zależności (2) jest przedstawiona za pomocą przykładów w tabeli 1. Tabela zawiera cztery zestawienia porównania dwóch ciągów, w postaci: obliczonej odległości Levenshteina (K) oraz miary odległości (P). Zależność (2) daje możliwość klasyfikacji i późniejszego szacowania treści będącej plagiatem względem całości badanych tekstów (zasada dotyczy również wyrazów).

Tabela 1

Przykład użycia zależności (2)

Lp.	Ciąg 1	Ciąg 2	K	P
1	Studium	Studia	2	$\approx 0{,}71$ (tj. $\approx 71\%$ podobieństwa)
2	Studia	Studia	0	1 (tj. 100% podobieństwa)
3	Studia	Berek	6	0 (tj. 0% podobieństwa)
4	XXXX	XXYY	2	0,5 (tj. 50% podobieństwa)

Klasyfikowanie tekstów może zostać przeprowadzone na podstawie formuły (3):

$$\text{if}\,(P = \text{fSim}(txt1, txt2) \geq q)\ \text{then}\ true;\ \text{else}\ false; \tag{3}$$

gdzie:

fSim – funkcja z zaimplementowanym mechanizmem analizy danych, zwracająca miarę podobieństwa dwóch ciągów tekstowych,

txt1 – ciąg tekstowy poddawany analizie,

txt2 – oryginalny ciąg tekstowy,

q – wartość z przedziału <0,1>, będąca wyznacznikiem klasyfikacji tekstu jako plagiat (miara podobieństwa dwóch ciągów, określana przez użytkownika). Odpowiednio dobrana do: dziedziny tekstów, języka lub innych informacji, np. o znanej (lub spodziewanej) ilości błędów ortograficznych zawartych w badanych tekstach.

Zależność (4) przedstawia algorytm pozwalający na obliczenie odległości Levenshteina dla dwóch badanych tekstów.

[5] Jeżeli ciągi są tej samej długości, ciągiem krótszym dla opisanej reguły jest dowolny z dwóch ciągów.

$$K_T = \prod_{i_T=1}^{N_T} \prod_{j_T=1}^{M_T} \mathbf{d}(i_T, j_T) = \min(\mathbf{d}(i_T - 1, j_T) + 1, \mathbf{d}(i_T, j_T - 1) + 1, \mathbf{d}(i_T - 1, j_T - 1) + \beta_T)$$

$$\begin{cases} \beta_T = 0 : \text{fSim}(a_T(i_T), b_T(j_T)) >= q \\ \beta_T = 1 : \text{fSim}(a_T(i_T), b_T(j_T)) < q \\ \mathbf{d}(i_T, 0) = i_T \\ \mathbf{d}(0, j_T) = j_T \\ \mathbf{d}(0,0) = 0 \end{cases} \tag{4}$$

gdzie:

K_T - odległość Levenshteina dwóch badanych tekstów,

$\mathbf{d_T}$ – macierz utworzona z dwóch porównywanych tekstów o rozmiarach wynikających z liczby wyrazów w poszczególnych tekstach,

$\mathbf{d_T}(i_T, j_T)$ – (i_T, j_T)-ty element macierzy $\mathbf{d_T}$,

β_T – zmienna przyjmująca odpowiednio wartości 0 lub 1,

$a_T(i_T)$ – i_T-ty wyraz tekstu a_T,

$b_T(j_T)$ – j_T-ty wyraz tekstu b_T,

N_T, M_T – długości badanych tekstów.

W celu obliczenia miary podobieństwa dla dwóch badanych tekstów należy użyć zależności opisanej równaniem (2).

Asymptotyczna złożoność obliczeniowa [7] przedstawionego algorytmu jest rzędu: $O(n^4)$. Wynika to z budowy mechanizmu, który ostatecznie składa się z czterech zagnieżdżonych w sobie pętli iteracyjnych (dwie pętle w zależności (4) i dwie pętle w zależności (1)).

3. Numeryczna weryfikacja wprowadzonego modelu analizy danych

Dla zbudowania tekstów o różnym podobieństwie wykorzystano bazę danych stu najpopularniejszych cytatów i złotych myśli sławnych ludzi[6] przetłumaczonych przez program Google Translate[7], z języka polskiego na języki: angielski, rosyjski i niemiecki, a następnie przetłumaczona z powrotem na język polski odpowiednio dla wymienionych języków[8].

Poszczególne teksty zostały przetworzone opisanym wyżej mechanizmem, implementowanym w funkcji określonej zależnością (5):

[6] Baza testowa cytatów i złotych myśli została zbudowana na podstawie różnych źródeł dostępnych w Internecie.

[7] Interfejs programu jest dostępny na stronie: translate.google.pl.

[8] Program Google Translate został wybrany ze względu na implementację jednych z najlepszych algorytmów maszynowego tłumaczenia tekstów na świecie, co potwierdzają czołowe miejsca w licznych rankingach czasopism komputerowych i prywatnych testów.

$$P = \mathrm{LevenshteinWSim}(txt1, txt2, q) \tag{5}$$

gdzie:

P – miara podobieństwa dwóch tekstów, wartość zwrócona przez funkcję LevenshteinWSim.

Tabela 2 zawiera fragment wyników analizy tekstów[9].

Tabela 2

Analiza tekstów cytatów i ich tłumaczeń

Lp.	Cytat w j. polskim (oryginalny) AUTOR	Cytat w j. angielskim (przetłumaczony)	Cytat w j. rosyjskim (przetłumaczony)	Cytat w j. niemieckim (przetłumaczony)
		Cytat w j. pol. tłumaczony z j. ang.	Cytat w j. pol. tłumaczony z j. ros.	Cytat w j. pol. tłumaczony z j. niem.
		Procent podobieństwa	Procent podobieństwa	Procent podobieństwa
1	Uczony jest człowiekiem, który wie o rzeczach nieznanych innym i nie ma pojęcia o tym, co znają wszyscy. ALBERT EINSTEIN	The scholar is a man who knows about things unknown to others and have no idea about what they know everyone.	Ученый человек, который знает о том, неизвестные другим, и понятия не имею о том, что они знают все.	Der Gelehrte ist ein Mann, über Dinge unbekannt, andere weiß und haben keine Ahnung, was sie wissen alle.
		Uczony jest człowiekiem, który wie o rzeczach nieznanych innym i nie masz pojęcia o tym, co znają wszyscy.	Nauczył człowiek, który wie o nieznanych innym i nie mają pojęcia o tym, co wiedzą wszystko.	Uczony jest człowiekiem o rzeczach nieznanych, a inne białe i nie mają pojęcia, co wszyscy wiedzą.
		(94,4%)	(66,7%)	(44,4%)
2	Celem naszych czynów powinno być czynienie dobra. PLATON	The purpose of our actions should be doing good.	Целью наших действий должны делать хорошо.	Der Zweck unseres Handelns werden sollte, Gutes zu tun.
		Celem naszych działań powinno być czynienie dobra.	Celem naszych działań powinno być dobrze.	Celem naszych działań powinno być czynienie dobra.
		(85,7%)	(71,4%)	(85,7%)

Jeżeli przyjąć, że plagiatem jest tekst, dla którego wartość funkcji *LevenshteinWSim* zwraca minimum 60% i wyznacznik podobieństwa wyrazów q wynosi 70%, to wykrywalność, na bazie przeanalizowanych danych, wynosi 63% dla języka angielskiego, 44% dla języka rosyjskiego i 39% dla języka niemieckiego. Wartości średnie porównania stu próbek dla poszczególnych języków wynoszą odpowiednio: 68.52% dla języka angielskiego, 55.53% dla języka rosyjskiego, 54.7% dla języka niemieckiego. Jednocześnie należy zwrócić uwagę,

[9] Dokument zawierający pełne wyniki analizy danych znajduje się pod adresem: www.pk.edu.pl/~aniewiarowski/publ/lev_cytaty.pdf

że test za pośrednictwem automatycznych translatorów obniża znacząco wyniki analizy podobieństwa tekstów. Obecnie dostępne translatory nie są programami niezawodnymi i w wielu przypadkach przetłumaczenie danych tekstowych w obie strony przyczynia się do całkowitej utraty sensu zdań. Negatywny wpływ na wyniki analizy dwóch tekstów ma w szczególności zastąpienie pierwotnych wyrazów całkowicie innymi po przetłumaczeniu w obie strony.[10,11]

W przetłumaczonych zdaniach, w których reprezentatywna liczba wyrazów (wartość parametru dla *LevenshteinWSim*) nie została zastąpiona całkowicie innymi wyrazami względem oryginałów (posiadających odmienny zestaw liter je budujący), wykrywalność podobieństwa tekstów jest wysoka.

Czas obliczeń przedstawionego mechanizmu (tj. przetworzenia w sumie trzystu cytatów) wyniósł poniżej 1 sekundy.[12]

4. Podsumowanie

Wyniki badań przeprowadzonych dla celów niniejszego artykułu pokazują, że mechanizm analizy danych tekstowych, oparty na algorytmie odległości transformacyjnej Levenshteina, umożliwia w bardzo szybkim czasie i z zadowalającą precyzją analizę podobieństwa krótkich fragmentów tekstów. W przeprowadzonych testach wykorzystano cytaty i złote myśli, ze względu na swoją specyfikę, tj. ukryte przesłanie filozoficzne i budowę z użyciem wyszukanych wyrazów, co ostatecznie miało na celu utrudnienie translacji na języki obce, a także ewentualne uniemożliwienie dostosowania próbek tekstów do osiągnięcia zadowalających wyników. Opisany mechanizm może znaleźć zastosowanie m.in. na stronach (portalach) internetowych, gdzie jest wymagana szybka analiza danych tekstowych (tzw. w locie, np. analiza komentarzy użytkowników) o niedużych rozmiarach [5] [13].

BIBLIOGRAFIA

1. Manning C. D., Prabhakar R., Hinrich S.: Introduction to Information Retrieval. Cambridge University Press, 2008.

[10] Przykłady: cytaty nr: 17, 78 dla języka rosyjskiego.

[11] W takim przypadku jakość analizy danych poprawia implementacja słownika wyrazów bliskoznacznych.

[12] Mechanizm został zaimplementowany dla celów testowych w języku programowania PHP, w celu prostego i szybkiego dostępu do bazy danych MySQL umieszczonej na tym samym serwerze. Dodatkowo jest dostępna wersja algorytmu w postaci pliku wykonywalnego pod adresem: www.pk.edu.pl/~aniewiarowski/publ-/levenWTest.exe

[13] Zaprezentowany mechanizm znalazł zastosowanie w module większego programu, analizującego podobieństwo tematów prac dyplomowych, w celu wykluczenia ich powtarzalności na przełomie lat.

2. Beeferman D., Berger A., Lafferty J.: Statistical models for text segmentation. Mach. Learn., Vol. 34(1÷3), 1999, s. 177÷210.
3. Lin D.: Automatic retrieval and clustering of similar words. COLING 1998, ACL, 1998, s. 768÷774.
4. Левенштейн В.И.: Двоичные коды с исправлением выпадений, вставок и замещений символов. Доклады Академий Наук СССР 163 (4), 1965, s. 845÷848.
5. Chakrabarti S.: Mining the Web: Analysis of Hypertext and Semi Structured Data. Morgan Kaufmann, 2002.
6. Hamming R. W.: Error Detecting and Error Correcting Codes. The Bell System Technical Journal, Vol. XXIX, April, 1950.
7. Christos H. Papadimitriou.: Złożoność obliczeniowa. Helion, Gliwice 2012.

Wpłynęło do Redakcji 6 grudnia 2012 r.

Abstract

This paper presents a proposition of text mining mechanism that analyzes the similarities of short texts, based on the Levenshtein Distance Algorithm. Originally, Levenshtein distance is used to calculate the number of changes needed to replace one word into second one. The proposed mechanism uses an algorithm to calculate the similarity between two or more sentences.

Analysis of effectiveness was performed based on 100 quotations, imported into the database. Quotations have been translated by Google Translate program, into three languages: English, German and Russian, in both directions.

Research showed a high detection of similarity sentences in a short time.

Adresy

Artur NIEWIAROWSKI: Politechnika Krakowska, Wydział Fizyki, Matematyki i Informatyki, ul. Podchorążych 1, 30-084 Kraków, Polska, aniewiarowski@pk.edu.pl.
Marek STANUSZEK: Politechnika Krakowska, Wydział Fizyki, Matematyki i Informatyki, ul. Warszawska 24, 31-155 Kraków, Polska, marek.stanuszek@pk.edu.pl.