Politechnika Warszawska
Wydział Matematyki i Nauk Informacyjnych

PRACA DYPLOMOWA MAGISTERSKA
NA KIERUNKU MATEMATYKA

AUTOMATYCZNA KATEGORYZACJA TEMATYCZNA TEKSTÓW PRZY UŻYCIU METRYK W PRZESTRZENI CIĄGÓW ZNAKÓW

TEXT CLUSTERING BASING ON STRING METRICS

Autor:
Natalia Potocka

Promotor:
dr Marek Gągolewski

Warszawa, grudzień 2015

....

podpis promotora podpis autora

Streszczenie

Ogromna ilość danych w postaci tekstowej przyczyniła się do rozwoju narzędzi automatycznie grupujących teksty, między innymi pod względem tematycznym. Dzięki temu znalezienie informacji w nich zawartych staje się łatwiejsze. Aby kategoryzować artykuły tematycznie, należy zwrócić uwagę na dwa elementy: sposób reprezentacji danych oraz algorytm używany do wykrywania skupień. Reprezentacja powinna zachować informacje o temacie. Stąd zazwyczaj przyjmuje się następujące założenia: po pierwsze kolejność słów nie gra roli, gdyż istotny jest jedynie ich rozkład, a po drugie fleksja słowa nie jest ważna, tylko jego znaczenie. Można zatem scharakteryzować każdy tekst jako wektor liczności wyrazów w nim występujących. Takie podejście jednakże jest wysoce nieskuteczne z powodu dużej wymiarowości danych (równej liczbie wszystkich słów występujących w tekstach) oraz ich rzadkości. Często stosowaną praktyką jest grupowanie wyrazów, by następnie reprezentować teksty przy użyciu wektorów liczności grup słów. Stąd niejednokrotnie stosowanym elementem przetwarzania danych tekstowych jest sprowadzenie słowa do jego rdzenia. Takie podejście nie uwzględnia jednak błędów w pisowni, literówek czy też braków znaków diakrytycznych, które często znajdują się w danych tekstowych. Można więc określić miary odległości na przestrzeni napisów (ciągów o dowolnej długości nad pewnym zbiorem skończonym, zwanym alfabetem), przyporządkowujące słowa do z góry określonych grup (definiowanych przez formy podstawowe). Celem niniejszej pracy jest zbadanie wpływu doboru tychże odległości na jakość automatycznej kategoryzacji tematycznej tekstów na podstawie artykułów z polskiej Wikipedii[1]. W pierwszym etapie grupowane są słowa przy użyciu wybranych odległości. Dalej reprezentujemy artykuły jako liczności występowania poszczególnych grup słów w danym tekście. Na podstawie tak uzyskanych danych przeprowadzamy analizę skupień tekstów. Ocena wyników odbywa się na podstawie otrzymanych grup z prawdziwymi kategoriami przypisanymi do każdego tekstu Wikipedii. Zauważmy, że użycie takiego zbioru danych wiąże się z kilkoma istotnymi wyzwaniami. Po pierwsze język polski jest językiem gramatycznie bardzo złożonym i zawiera dużą liczbę słów. Dalej zbiór tekstów z polskiej Wikipedii jest względnie duży, co rodzi potrzebę odpowiedniego zarządzenia danymi oraz optymalizacji procesów z powodu ograniczonych zasobów obliczeniowych i pamięciowych. Co więcej niektóre metody analizy danych nie dają się efektywne stosować na dużych zbiorach danych.

[1] `http://pl.wikipedia.org/`

Abstract

The vast amount of data in text form contributed to the development of tools that automatically group texts, among other thematically. As a result, finding the information contained in them becomes easier. To categorize articles by subject, one should pay attention to the following elements: a way to represent the data and the algorithm used to detect clusters. Representation should keep information about the topic. Thus, usually, the following assumptions are made: firstly, order of words does not matter, because only their distribution is relevant, and secondly inflection of words is not important, but their meaning. One can therefore characterize each text as a vector of frequencies of words occurring in it. However, such an approach, is highly inefficient due to the high dimensionality of data (equal to the total number of words found in all the texts) and their sparsity. A common practice is to group words and then represent texts using vectors of frequencies of groups of words. Therefore, often used component of processing text data is to reduce words to their root. Such an approach, however, does not include spelling errors, typos or deficiencies of diatric marks, which are often found in the text data. Then, one can determine the measure of distance on the strings (strings of any length over a finite set called the alphabet), assigning words to predetermined groups (defined by basic forms). The aim of this study is to investigate the influence of the selection of these distances on the quality of automatic thematic categorization of texts based on articles from Polish Wikipedia[2]. In the first step, words are grouped using the selected distances. Moreover, articles are represented as frequencies of particular groups of words that appeared in the text. On the basis of the obtained data, we cluster the articles. Evaluation of the results is carried out on the basis of the groups with real categories assigned to each of the text of Wikipedia. Note that the use of such a data set is associated with several major challenges. Firstly, the Polish language is the language grammatically very complex and contains a large number of words. Next, set of texts from Polish Wikipedia is relatively large, which raises the need for adequate data management and process optimization due to limited computing and storage resources. Furthermore, some data analysis methods cannot be effectively used on large data sets.

[2] http://pl.wikipedia.org/

Spis treści

Rozdział 1

Wstęp

Rozwój internetu przyczynił się do powstania ogromnej ilości danych w postaci tekstowej. W konsekwencji stają się one coraz powszechniej wykorzystywanym źródłem, często cennej, wiedzy, choć są trudne w analizie z powodu ich objętości, różnorodności oraz podatności na błędy. Jednym z istotnych zagadnień związanych z danymi tekstowymi jest grupowanie. W wielu przypadkach kategoryzowanie dokumentów pod kątem podobieństwa tematycznego znacząco ułatwia wyszukiwanie informacji. Z grupowaniem wiążą się dwa zasadnicze elementy: sposób reprezentacji danych oraz algorytm używany do wykrywania skupień. Oczywistym jest, że gdy chcemy kategoryzować teksty pod kątem tematycznym, to ich reprezentacja powinna zachowywać informacje o temacie. Zazwyczaj przyjmuje się naturalne założenie, że kolejność słów nie ma znaczenia – istotny jest jedynie ich rozkład. Podstawowym podejściem jest charakteryzacja tekstu przy użyciu liczności poszczególnych słów w dokumencie – cały zbiór jest wówczas opisany przez macierz $[m_{ij}]$, której ij-ty element to liczność wystąpień j-tego słowa w i-tym tekście. Jednakże taka reprezentacja jest najczęściej nieefektywna – dane opisane w ten sposób mają bardzo duży wymiar (równy liczbie wszystkich unikalnych słów w nich zawartych), a przy tym są one bardzo rzadkie – liczba zerowych elementów macierzy jest bardzo duża (często ponad 90%). Analiza skupień na takim zbiorze danych jest często wysoce nieskuteczna, a czasem wręcz niemożliwa z powodu braku zasobów pamięciowych i obliczeniowych. Rodzi się więc potrzeba zmniejszenia wymiaru danych.

Często stosowaną praktyką jest grupowanie słów, by następnie reprezentować teksty przy użyciu wektorów $(n_1, \ldots, n_K)$, gdzie n_i jest licznością słów z i-tej grupy, $i = 1, \ldots, K$, a K liczbą grup słów. Słowa można pogrupować na wiele sposobów. Przede wszystkim można oprzeć się na założeniu, że fleksja danego słowa jest nieistotna, tzn. ważne jest znaczenie wyrazu, a nie jego odmiana, co wydaje się być rozsądnym podejściem w kategoryzacji tematycznej. Stąd niejednokrotnie stosowanym elementem przetwarzania danych tekstowych jest *stemming*, czyli sprowadzenie słowa do jego rdzenia, np. wyrazy `robiący, robiłem, robię` sprowadzają się do słowa `robić`. W najprostszej wersji podejście to wymaga dysponowania słownikiem słów wraz z ich odmianami. Rzadko jednak zawiera on wszystkie istniejące wyrazy, zwłaszcza gdy liczba możliwych odmian jest bardzo duża (jak np. w języku polskim). Ponadto podejście to nie uwzględnia częstych niedoskonałości danych – słownik nie zawiera słów z błędami w pisowni i literówkami, które mogą występować w tekstach. Coraz częściej przedmiotem analizy są nieformalne teksty lub wiadomości, w których np. piszący świadomie nie używa znaków diakrytycznych. Zachodzi zatem potrzeba radzenia sobie z błędnie zapisanymi wyrazami.

Z wymienionych wyżej powodów często stosowaną praktyką jest grupowanie wyrazów w oparciu o pewne miary podobieństwa określone na przestrzeni napisów (ciągów o dowolnej długości nad pewnym zbiorem skończonym, zwanym alfabetem). Można wyróżnić tutaj dwa podejścia: przyporządkowywanie słów do z góry określonych grup (definiowanych przez formy podstawowe) lub wykrywanie skupień spośród wyrazów występujących w tekstach. Niniejsza praca porusza problem wyboru odległości przy zastosowaniu pierwszego z wymienionych podejść. W literaturze można znaleźć wiele różnych definicji odległości (np. [18, 6, 20]). W głównej mierze opierają się one na porównywaniu liczby wystąpień takich samych sekwencji znaków lub zliczeniu operacji, które przetwarzają jeden napis w drugi. Dodajmy, że miary te znajdują zastosowanie również w innych dziedzinach, m.in. biologii obliczeniowej, przetwarzaniu sygnałów czy korekcie błędnego tekstu.

Celem niniejszej pracy jest zbadanie wpływu doboru odległości na przestrzeni napisów na jakość automatycznej kategoryzacji tematycznej tekstów. Innymi słowy, badana jest jakość grupowania artykułów dla różnych reprezentacji tekstu. Reprezentacje te odnoszą się do różnych grup słów, otrzymanych przy użyciu różnych odległości. Danymi, na których przeprowadzona została analiza jest zbiór artykułów polskiej Wikipedii[1]. W pierwszym etapie grupowane są słowa przy użyciu wybranych odległości. Dalej reprezentujemy artykuły jako liczności występowania poszczególnych grup słów w danym tekście. Na podstawie tak uzyskanych danych przeprowadzamy analizę skupień tekstów. Ocena wyników odbywa się na podstawie otrzymanych grup z prawdziwymi kategoriami przypisanymi do każdego tekstu Wikipedii.

Zauważmy, że użycie polskiej Wikipedii wiąże się z kilkoma istotnymi wyzwaniami. Po pierwsze język polski jest językiem gramatycznie bardzo złożonym, przede wszystkim ze względu na to, że zawiera dużo odmian (przez przypadki, liczby, osoby, czasy, tryby, strony i inne). Stąd liczba istniejących słów w języku polskim jest bardzo duża. Dalej określenie formy bezokolicznikowej czy mianownikowej często nie jest łatwe, a czasem niemożliwe bez znajomości kontekstu (np. słowo `piła` może być zarówno rzeczownikiem, jak i być odmianą czasownika `pić`). Po drugie zbiór tekstów z polskiej Wikipedii jest względnie duży – ponad milion artykułów, na które składają się ponad dwa miliony unikalnych słów. Przetwarzanie tak obszernego zbioru rodzi potrzebę odpowiedniego zarządzenia danymi (zbudowaniem adekwatnej bazy danych) oraz optymalizacji procesów z powodu ograniczonych zasobów obliczeniowych i pamięciowych. Po trzecie niektóre metody analizy danych nie dają się efektywne stosować na dużych zbiorach danych.

Organizacja niniejszej pracy jest następująca. Rozdział drugi to przegląd odległości na przestrzeni ciągów znaków. Podano ich formalne definicje, własności, jak i przykłady użyć oraz zastosowania. Zostały one podzielone na trzy kategorie: odległości oparte na operacjach edycyjnych, oparte na q-gramach oraz miary heurystyczne. Trzeci rozdział poświęcony jest analizie skupień. Opisane zostały algorytmy k-średnich przy użyciu zarówno metody wsadowej, jak i metod najszybszego spadku. Ponadto w rozdziale tym omówione zostały algorytmy hierarchiczne wraz z różnymi kryteriami odmienności między skupieniami. Co więcej są tam opisane metody oceny jakości podziału na skupienia. Kolejny rozdział dotyczy części praktycznej pracy. Przedstawiony został algorytm zastosowany na analizowanym zbiorze wraz z dokładnym opisem badania. Ponadto w rozdziale czwartym znajdują szczegółowe wyniki. Ostatnia część dotyczy kierunków dalszych prac.

[1] `http://pl.wikipedia.org/`

Rozdział 2

Odległości na przestrzeni ciągów znaków

2.1. Podstawowe definicje

Definicja 2.1. *Niech $\Sigma = \{\sigma_1, \ldots, \sigma_k\}$ będzie skończonym uporządkowanym zbiorem o liczności $|\Sigma|$, zwanym* alfabetem. Napisem *nazywamy skończony ciąg znaków z Σ. Zbiór wszystkich napisów o długości n nad Σ jest oznaczony przez Σ^n, podczas gdy przez $\Sigma^* = \bigcup_{n=0}^{\infty} \Sigma^n$ rozumiemy zbiór wszystkich napisów utworzonych ze znaków z Σ [6].*

O ile nie podano inaczej, używamy zmiennych s, t, u, v, w, x, y jako oznaczenie napisów oraz a, b, c do oznaczenia napisów jednoznakowych albo po prostu *znaków*. Pusty napis jest oznaczany przez ε. Przez $|s|$, dla każdego napisu $s \in \Sigma^*$, rozumiemy jego długość, czyli liczbę znaków w napisie. Ciąg napisów i/lub znaków oznacza ich złączenie, np. *stu* to napis powstały ze złączenia napisów s, t oraz u, natomiast *abc*, to napis powstały ze złączenia znaków a, b oraz c. Dla rozróżnienia napisów od zmiennych reprezentujących napis, te pierwsze oznaczamy pismem maszynowym, np. `napis`.

Poprzez s_i rozumiemy i-ty znak z napisu s, dla każdego $i \in \{1, \ldots, |s|\}$. Podciąg kolejnych przylegających do siebie znaków z napisu nazywamy *podnapisem*. Podnapisem napisu s, który zaczyna się od i-tego znaku, a kończy na j-tym znaku, oznaczamy przez $s_{i:j}$, tj. $s_{i:j} = s_i s_{i+1} \ldots s_j$ dla $i \leq j$. Zakładamy również, że jeśli $j < i$, to $s_{i:j} = \varepsilon$ [6, 31].

Definicja 2.2. *Załóżmy, że napis s jest reprezentacją złączenia trzech, być może pustych, podnapisów w, x i y, tj. $s = wxy$. Wówczas podnapis w nazywamy* przedrostkiem, *natomiast podnapis y – przyrostkiem [6].*

Definicja 2.3. *Podnapis złożony z kolejnych, przylegających do siebie znaków, o ustalonej długości $q \geq 1$ jest nazywany* q-gramem. *q-gramy o q równym jeden, dwa lub trzy mają specjalne nazwy:* unigram, bigram *i* trigram. *Jeśli $q > |s|$, to q-gramy napisu s są napisami pustymi [6]. Maksymalna liczba wystąpień różnych q-gramów w napisie s wynosi $|s| - q - 1$.*

Przykład 2.1. Niech Σ będzie alfabetem złożonym z 26 małych liter alfabetu łacińskiego oraz niech $s =$ `ela`. Wówczas mamy $|s| = 3$, $s_1 =$ `e`, $s_2 =$ `l`, $s_3 =$ `a`. Podnapis 1:2 napisu s

to $s_{1:2} = \texttt{el}$. W napisie tym mamy do czynienia jedynie z q-gramami o q równym jeden, dwa oraz trzy, odpowiednio: `e`, `l`, `a`; `el`, `la` oraz `ela`.

We wszystkich przykładach niniejszego rozdziału zakładamy, jeśli nie podano inaczej, że Σ składa się z 32 liter polskiego alfabetu oraz liter `q`, `v` i `x`.

2.2. Odległości na przestrzeni ciągów znaków

Określanie odległości między napisami jest ważną częścią przetwarzania danych, zwłaszcza w problemach takich jak dopasowanie statystyczne, wyszukiwanie tekstów, klasyfikacja tekstów czy sprawdzanie pisowni. Największa trudność polega na policzeniu podobieństwa między dwoma napisami w terminach metryk na przestrzeni ciągów znaków. W niniejszym podrozdziale zajmiemy się odległościami na napisach, tj. funkcjami $d : \Sigma^* \times \Sigma^* \to [0, \infty)$. W literaturze można znaleźć wiele różnych funkcji tego typu, które różnią się genezą powstania, podejściem do problemu oraz zastosowaniami. W pracy zajmiemy się odległościami, która można podzielić na trzy garupy:

- oparte na operacjach edycyjnych (*edit operations*),
- oparte na q-gramach,
- miary heurystyczne.

Aby wyliczyć odległość opartą na operacjach edycyjnych, trzeba określić liczbę fundamentalnych transformacji potrzebnych do przetworzenia jednego napisu w drugi. Mogą się w nich zawierać zamiany, usunięcia, wstawienia oraz transpozycje znaków. Temu rodzajowi odległości poświęcimy największą część niniejszego rozdziału. Odległości oparte na q-gramach pozwalają na określenie podobieństwa między dwoma napisami przez porównanie liczby występowania q-elementowych ciągów znaków. Natomiast miary heurystyczne są rzadko stosowane, gdyż nie mają silnych matematycznych podstaw, ale zostały rozwinięte jako praktyczne narzędzie stosowane w konkretnych przypadkach.

Odległości między napisami były i nadal są niezbędne w problemach biologii obliczeniowej, przetwarzaniu sygnałów oraz przeszukiwaniu tekstów. Ten pierwszy obszar zastosowań koncentruje się na wyszukiwaniu wzorców w DNA i sekwencjach białkowych, które mogą być zapisane jako długie napisy nad specyficznym alfabetem (np. {`A, C, T, G`}). Wyszukiwanie w nich szczególnych ciągów znaków okazuje się być fundamentalną operacją przy składaniu części DNA uzyskanych z eksperymentów, czy też określaniu jak bardzo różnią się dwie sekwencje genów [20]. Więcej na temat zastosowań odległości między napisami w biologii obliczeniowej można znaleźć w [28], [21] czy też [26].

Kolejnym obszarem zastosowań odległości na przestrzeni ciągów znaków jest przetwarzanie sygnałów. Jednym z jego zagadnień jest określenie transmitowanego tekstu, mając dany sygnał audio. Nawet uproszczony problem, sprowadzony jedynie do rozpoznawania słowa z małego zbioru alternatyw jest złożony, jako że sygnał może być skompresowany w czasie, części słów mogą być niewyraźne itd. Idealne dopasowanie jest praktycznie niemożliwe [20]. Innym problemem jest korekta błędów pisowni. Fizyczna transmisja sygnału podatna jest na błędy. Aby zagwarantować transmisję fizycznym kanałem, konieczne jest, żeby odzyskać poprawną wiadomość po modyfikacji uzyskanej wskutek transmisji. W tym przypadku można nie wiedzieć nawet czego szukać, a trzeba dostać tekst, który jest poprawny i najbliższy otrzymanej

wiadomości [20]. Więcej na temat zastosowań odległości między napisami w przetwarzaniu sygnałów można znaleźć w [18], [32], czy też [8].

Problem korekty źle napisanego tekstu jest prawdopodobnie najstarszym potencjalnym zastosowaniem dla odległości na napisach. Odniesienia do tego problemu można znaleźć nawet z lat 20. ubiegłego wieku [19]. Od lat 60. optymalne dopasowanie napisów przy pomocy odległości stało się jednym z najbardziej popularnych narzędzi, które radziłoby sobie z poprawą tekstów. Przykładowo, ok. 80% błędów można poprawić używając tylko jednego wstawienia, usunięcia, zamiany bądź transpozycji znaków [7]. Bardzo ważnym zastosowaniem poprawy tekstów jest wydobywanie informacji (ang. *Information Retrieval* – IR) – jest to jedna z najbardziej wymagających gałęzi zastosowań odległości na napisach. W IR chodzi o wydobycie istotnych informacji z dużych zbiorów tekstów, a optymalne dopasowanie napisów jest jego kluczowym narzędziem. Obecnie, każde narzędzie wydobywające informacje z tekstów, dopasowuje napisy bądź wzorce, aby zniwelować liczbę błędów w tekście. Więcej na temat zastosowań odległości między napisami w problemie optymalnego dopasowania napisów można znaleźć w literaturze, m.in. [6], [20], [33], [34], [22] czy też [16].

2.2.1. Odległości oparte na operacjach edycyjnych

Ścieżka edycyjna i bazowe operacje edycyjne. *Odległość edycyjna* $ED(s,t)$ między dwoma napisami s i t to minimalna liczba operacji edycyjnych potrzebna do przetworzenia s w t (i ∞, gdy taki ciąg nie istnieje) [20]. *Ścisłą odległością edycyjną* nazywamy minimalną liczbę nienakładających się operacji edycyjnych, które pozwalają przekształcić jeden napis w drugi, i które nie przekształcają dwa razy tego samego podnapisu [6].

Napis może zostać przetworzony w drugi poprzez wykonanie na nim ciągu przekształceń jego podnapisów. Ten ciąg nazywany jest *ścieżką edycyjną (śladem edycji)*, podczas gdy przekształcenia są nazywane *bazowymi operacjami edycyjnymi*. Bazowe operacje edycyjne, które polegają na przekształceniu napisu s w napis t, są oznaczane przez $s \to t$. Zbiór wszystkich bazowych operacji edycyjnych oznaczamy przez $\mathbb{B}$ [6].

Bazowe operacje edycyjne są zazwyczaj ograniczone do:

- usunięcia znaku: `l` $\to \varepsilon$, tj. usunięcia litery `l` , np. `ela` $\to$ `ea`,
- wstawienia znaku: $\varepsilon \to$ `k`, tj. wstawienia litery `k`, np. `ela` $\to$ `elka`,
- zamiany znaku: `e` $\to$ `a`, tj. zamiany litery `e` na `a`, np. `ala` $\to$ `ela`,
- transpozycji: `el` $\to$ `le`, tj. przestawienia dwóch przylegających liter `e` i `l`, np. `ela` $\to$ `lea`.

Często transpozycja znaków nie należy do zbioru operacji bazowych, jako że można ją zastąpić usunięciem i wstawieniem znaku. W niniejszej pracy jednak operacja ta należy do zbioru operacji bazowych.

Własność 2.1. *Zakładamy, że $\mathbb{B}$ spełnia następujące własności [6]:*

- *jeśli $s \to t \in \mathbb{B}$, to odwrotna operacja $t \to s$ również należy do $\mathbb{B}$;*
- *$a \to a \in \mathbb{B}$ (operacja identycznościowa dla jednego znaku należy do $\mathbb{B}$);*
- *zbiór $\mathbb{B}$ jest zupełny: dla dwóch dowolnych napisów s i t, istnieje ślad edycji, który przekształca s w t.*

Zauważmy, że zbiór $\mathbb{B}$ nie musi być skończony.

Odległość edycyjna. Podobieństwo dwóch napisów może być wyrażone jako długość ścieżki edycyjnej, dzięki której jeden napis zostaje przekształcony w drugi:

Definicja 2.4. *Mając dany zbiór bazowych operacji edycyjnych,* odległość edycyjna $ED(s,t)$ *jest równa długości najkrótszej ścieżki edycyjnej, która przekształca napis s w napis t. Najkrótsza ścieżka, która przekształca napis s w napis t jest nazywana* optymalną *ścieżką edycyjną [6].*

Przykład 2.2. Weźmy napisy `foczka` oraz `kozak`. Ścieżka edycyjna między nimi może mieć następującą postać:

$$\texttt{foczka} \xrightarrow{trans.\ z\ i\ k} \texttt{fockza} \xrightarrow{trans.\ c\ i\ k} \texttt{fokcza} \xrightarrow{trans.\ o\ i\ k} \texttt{fkocza} \xrightarrow{us.\ f} \texttt{kocza} \xrightarrow{us.\ c} \texttt{koza} \xrightarrow{wst.\ k} \texttt{kozak}.$$

Optymalna ścieżka natomiast ma następującą postać:

$$\texttt{foczka} \xrightarrow{zm.\ f\ na\ k} \texttt{koczka} \xrightarrow{us.\ c} \texttt{kozka} \xrightarrow{trans.\ k\ i\ a} \texttt{kozak}.$$

Do przykładowych odległości edycyjnych zaliczamy odległość: Hamminga, najdłuższego wspólnego podnapisu (*longest common substring*), Levenshteina, optymalnego dopasowania napisów (*optimal string alignment*), Damareu-Levenshteina. Odległości te różnią się zbiorem bazowych operacji edycyjnych. Jeśli w zbiorze tym znajduje się tylko zamiana znaków, to mamy do czynienia z odległością Hamminga. Gdy zbiór bazowych operacji edycyjnych zawiera wstawienia i usunięcia znaków, to jest to odległość najdłuższego wspólnego podnapisu. Gdyby $\mathbb{B}$ powiększyć o zamianę znaków, to otrzymamy odległość Levenshteina. Dwie ostatnie odległości, tj. optymalnego dopasowania napisów oraz Damareu-Levenshteina, mają w zbiorze bazowych operacji edycyjnych usunięcie, wstawienie, zamianę oraz transpozycję znaków. Formalne definicje powyższych funkcji znajdują się w dalszej części niniejszego rozdziału.

Definicja odległości edycyjnej może być również interpretowana jako minimalny koszt, dzięki któremu przekształcamy jeden napis w drugi. Definicję można uogólnić na dwa sposoby. Po pierwsze, bazowe operacje edycyjne mogą mieć przydzielone koszty (wagi) $\delta(a \to b)$ [33]. Zazwyczaj koszt każdej operacji wynosi jeden, jednak można, na przykład, nadać transpozycji mniejszy koszt niż operacji wstawienia znaku. Dalej, można rozszerzyć funkcję kosztu δ na ścieżkę edycyjną $E = a_1 \to b_1, a_2 \to b_2, \ldots, a_{|E|} \to b_{|E|}$ przez $\delta(E) = \sum_{i=1}^{|E|} \delta(a_i \to b_i)$ [6]. Odtąd przez odległość między napisem s a napisem t będziemy rozumieć minimalny ze wszystkich możliwych kosztów ścieżek przekształcających s w t. Odległości zdefiniowane w ten sposób zazwyczaj są nazywane *uogólnionymi* odległościami edycyjnymi.
Po drugie, zbiór operacji edycyjnych $\mathbb{B}$ może zostać rozszerzony o ważone zamiany (pod)napisów, zamiast operacji edycyjnych wykonywanych na pojedynczych znakach [29]. Odległości zdefiniowane w ten sposób zazwyczaj są nazywane *rozszerzonymi* odległościami edycyjnymi. Przykładowo, $\mathbb{B}$ może zawierać operację `x` $\to$ `ks` o koszcie jednostkowym. Wówczas rozszerzona odległość pomiędzy napisami `xero` i `ksero` wynosi jeden, podczas gdy standardowa (zwykła, nierozszerzona) odległość wyniosłaby dwa [6].

Definicja 2.5. *Mając dany zbiór bazowych operacji edycyjnych $\mathbb{B}$ oraz funkcję δ, która nadaje koszt wszystkim bazowym operacjom edycyjnym z $\mathbb{B}$, uogólniona odległość edycyjna między*

napisami s i t jest zdefiniowana jako minimalny spośród kosztów wszystkich możliwych ścieżek edycyjnych, które przekształcają s w t [6].

Zazwyczaj koszt pojedynczej operacji z $\mathbb{B}$ jest równy jeden. Czasem jednak nadaje się poszczególnym operacjom różne koszty, dając np. transpozycji mniejszą wagę niż wstawieniu znaku. Gdy koszt wszystkich operacji jest równy jeden, to mówimy po prostu o odległości edycyjnej, natomiast gdy różne operacje mają różne wagi, to mówimy o *ważonej* odległości edycyjnej.

Własność 2.2. *Zakładamy, że funkcja kosztu $\delta(s \to t)$ ma następujące własności [6]:*

- $\delta(s \to t) \geq 0$ *(koszt operacji jest liczbą nieujemny),*
- $\delta(s \to t) = \delta(t \to s)$ *(symetria),*
- $\delta(s \to s) = 0$ *i* $\delta(s \to t) = 0 \Rightarrow s = t$ *(identyczność),*
- $\forall \gamma > 0$ *zbiór bazowych operacji* $\{s \to t \in \mathbb{B} | \delta(s \to t) < \gamma\}$ *jest skończony (skończoność podzbioru bazowych operacji, których koszt jest ograniczony z góry).*

Zauważmy, że ostatnia własność jest zawsze spełniona dla skończonego zbioru $\mathbb{B}$.

Twierdzenie 2.3. *Z własności 2.1 i 2.2 wynika, że:*

- *dla każdych dwóch napisów s i t, istnieje ścieżka o minimalnym koszcie, tj. dobrze zdefiniowana odległość edycyjna z s do t [6],*
- *ogólna odległość edycyjna z definicji 2.5 jest metryką [33].*

Dowód. Żeby udowodnić, że $ED(s,t)$ jest metryką, musimy pokazać, że $ED(s,t)$ istnieje, jest dodatnio określona, symetryczna oraz subaddytywna (tj. spełnia nierówność trójkąta).

Z własności 2.2 wynika, że funkcja kosztu jest nieujemna i że tylko identyczność ma koszt równy zero. Stąd, bez utraty ogólności, możemy rozważyć jedynie takie ścieżki edycyjne, które nie zawierają operacji identycznościowych. Zatem, jeśli $s = t$, to jedyna optymalna ścieżka (która nie zawiera operacji identycznościowych) jest pusta i ma zerowy koszt. Jeśli $s \neq t$, to z zupełności zbioru bazowych operacji edycyjnych wynika, że istnieje jedna lub więcej ścieżek edycyjnych, które przekształcają s w t. Wszystkie te ścieżki składają się z operacji edycyjnych o ściśle dodatnim koszcie.

Niech γ będzie kosztem ścieżki przekształcającej s w t. Rozważmy zbiór A ścieżek edycyjnych, które przekształcają s w t i których koszt jest ograniczony z góry przez γ. Zbiór A jest niepusty i składa się z operacji edycyjnych o dodatnim koszcie mniejszym niż γ. Zbiór operacji bazowych, których koszt jest ograniczony z góry przez γ jest skończony, co dowodzi, że zbiór A jest również skończony. Ponieważ A jest niepusty i skończony, to ścieżki edycyjne o mininalnym (dodatnim) koszcie istnieją i należą do A. Stąd, $ED(s,t) > 0$ dla $s \neq t$, tj. odległość edycyjna jest dodatnio określona.

Aby udowodnić symetrię odległości edycyjnej, rozważmy optymalną ścieżkę E, która przekształca s w t, oraz odpowiadającą jej odwrotną ścieżkę E_r, która przekształca t w s. Równość ich kosztów $\delta(E) = \delta(E_r)$ wynika z symetrii funkcji kosztu i symetrii zbioru operacji bazowych $\mathbb{B}$.

Aby pokazać subaddytywność, rozważmy optymalną ścieżkę E_1, która przekształca s w t, optymalną ścieżkę E_2, która przekształca t w u, oraz złożenie ścieżek E_1E_2, które przekształca s w u. Z tego, że $\delta(E_1E_2) = \delta(E_1) + \delta(E_2) = ED(s,t) + ED(t,u)$ oraz $\delta(E_1E_2) \geq ED(s,u)$ (gdyż E_1E_2 nie musi być optymalną ścieżka, przekształcającą s w u) wynika, że $ED(s,t) + ED(t,u) \geq ED(s,u)$. ■

Odległość edycyjna jest metryką, nawet gdy funkcja kosztu δ nie jest subaddytywna. Co więcej, ponieważ ciąg nakładających się operacji, które przekształcają s w t, mogą mieć mniejszy koszt niż $\delta(s \to t)$, $\delta(s \to t)$ może być większe niż $ED(s,t)$. Rozważmy, na przykład, następujący alfabet: {`a, b, c`}, gdzie symetria i brak subaddytywności funkcji δ jest zdefiniowana następująco:

$$\delta(\texttt{a} \to \texttt{c}) = \delta(\texttt{b} \to \texttt{c}) = 1$$
$$\delta(\texttt{a} \to \varepsilon) = \delta(\texttt{b} \to \varepsilon) = \delta(\texttt{c} \to \varepsilon) = 2$$
$$\delta(\texttt{a} \to \texttt{b}) = 3$$

Można zobaczyć, że $3 = \delta(\texttt{a} \to \texttt{b}) > \delta(\texttt{a} \to \texttt{c}) + \delta(\texttt{c} \to \texttt{b}) = 2$. Stąd optymalna ścieżka edycyjna $(\texttt{a} \to \texttt{c}, \texttt{c} \to \texttt{b})$ przekształca `a` w `b` z kosztem równym 2.

Ścisła odległość edycyjna. Subaddytywność odległości edycyjnej pozwala używać metod właściwych przestrzeniom metrycznym. Niemniej jednak, problem minimalizacji zbioru nakładających się operacji edycyjnych, może być trudny. Aby zrównoważyć złożoność obliczeniową, zazwyczaj używana jest funkcja podobieństwa, zdefiniowana jako minimum kosztu *ścisłej ścieżki edycyjnej*. Ta ostatnia nie zawiera nakładających się na siebie operacji edycyjnych i nie modyfikuje tego samego podnapisu więcej niż raz. Odpowiadająca jej odległość edycyjna nazywana jest *ścisłą odległością edycyjną* [6]:

Definicja 2.6. *Niech napisy s i t zostaną podzielone na tę samą liczbę, być może pustych, podnapisów: $s = s_1 s_2 \dots s_l$ i $t = t_1 t_2 \dots t_l$, takich, że $s_i \to t_i \in \mathbb{B}$.* Ścisłą ścieżką edycyjną *nazywamy taką ścieżkę, że nie występują w niej następujące operacje:*

- $s_i \to s_{i_j} \to t_i$ *(modyfikacja tego samego podnapisu więcej niż raz),*
- $s_i \to t_i, s_{i+1} \to t_{i+1}, t_i t_{i+1} \to t_k$ *(nakładające się operacje).*

Lemat 2.4. *Dowolna nieścisła odległość edycyjna ogranicza z dołu odpowiadającą jej ścisłą odległość edycyjną [6].*

Rozważmy napisy `ab`, `ba` oraz `acb`. Z jednej strony, najkrótsza nieścisła ścieżka edycyjna, która przekształca `ba` w `acb`, tj. $(\texttt{ba} \to \texttt{ab}, \varepsilon \to \texttt{c})$ zawiera dwie operacje: najpierw zamienia znaki `a` i `b`, a następnie wstawia `c` pomiędzy nie. Zauważmy, że wstawienie przekształca już transformowany napis. Jednakowoż, jeśli kolejne przekształcenia tego samego podnapisu są wykluczone, to najkrótsza ścieżka edycyjna, która przekształca `ba` w `acb`, składa się z trzech operacji edycyjnych, np. $(\texttt{b} \to \varepsilon, \varepsilon \to \texttt{c}, \varepsilon \to \texttt{b})$. Stąd nieścisła odległość edycyjna jest równa dwa, podczas gdy ścisła odległość wynosi trzy [6]. **Optymalne dopasowanie.**

Definicja 2.7. *Niech napisy s i t zostaną podzielone na tę samą liczbę, być może pustych, podnapisów: $s = s_1 s_2 \dots s_l$ i $t = t_1 t_2 \dots t_l$, takich, że $s_i \to t_i \in \mathbb{B}$. Co więcej, zakładamy, że s_i i t_j nie mogą być puste dla $i = j$. Mówimy, że ten podział definiuje* dopasowanie *$A = (s_1 s_2 \dots s_l, t_1 t_2 \dots t_l)$ pomiędzy napisami s i t, w którym podnapis s_i jest dopasowany do podnapisu t_i [6].*

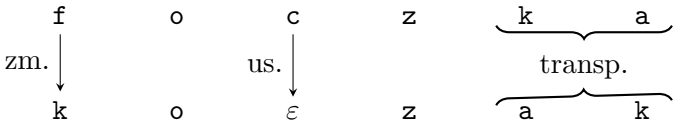
Dopasowanie reprezentuje ścisłą ścieżkę edycyjną $E = s_1 \to t_1, s_2 \to t_2, \ldots, s_l \to t_l$. Definiujemy *koszt dopasowania* A jako koszt odpowiadającej mu ścieżki edycyjnej i oznaczamy go przez $\delta(A)$:

$$\delta(A) = \sum_{i=1}^{l} \delta(s_i \to t_i), \tag{2.1}$$

gdzie do $\mathbb{B}$ należy usunięcie, wstawienie, zamiana oraz transpozycja znaków.

Optymalne dopasowanie to dopasowanie o najmniejszym koszcie [6].

Przykład 2.3. Przykład optymalnego dopasowania pomiędzy słowami `foczka` i `kozak` prezentuje Rys. 2.1. Odpowiadająca mu ścieżka edycyjna składa się z zamiany `f` $\to$ `k`, usunięcia `c` $\to \varepsilon$ oraz transpozycji `ka` $\to$ `ak`.



Rysunek 2.1: Przykład optymalnego dopasowania między napisami `foczka` i `kozak`.

Warto zauważyć, że istnieje różnowartościowe przekształcenie między zbiorem ścisłych ścieżek edycyjnych i zbiorem optymalnych dopasowań: każda ścisła ścieżka edycyjna o minimalnym koszcie reprezentuje dopasowanie o najmniejszym koszcie i odwrotnie. Stąd można zastąpić problem znalezienia optymalnej ścisłej odległości edycyjnej przez problem znalezienia optymalnego dopasowania, co też zastosujemy dalej [6].

Obliczanie odległości edycyjnej. Główną zasadą dynamicznego algorytmu, liczącego koszt optymalnego dopasowania, jest wyrażenie kosztu dopasowania pomiędzy napisami s i t, używając kosztu dopasowania ich przedrostków. Rozważmy przedrostek $s_{1:i}$ o długości i i przedrostek $t_{1:j}$ o długości j, odpowiednio napisów s i t. Załóżmy, że $A = (s_1 s_2 \ldots s_l, t_1 t_2 \ldots t_l)$ jest optymalnym dopasowaniem między $s_{1:i}$ i $t_{1:j}$, którego koszt oznaczamy przez $C_{i,j}$ [6].

Używając równania (2.1) oraz definicji optymalnego dopasowania, łatwo pokazać, że $C_{i,j}$ może zostać policzone przy użyciu następującej ogólnej rekurencji [29]:

$$\begin{aligned} &C_{0,0} = 0, \\ &C_{i,j} = \min\{\delta(s_{i':i} \to t_{j':j}) + C_{i'-1,j'-1} | s_{i':i} \to t_{j':j} \in \mathbb{B}\}. \end{aligned} \tag{2.2}$$

Można zauważyć, że:

- koszt dopasowania napisów s i t jest równy $C_{|s|,|t|}$,
- wszystkie optymalne dopasowania mogą zostać wyznaczone przez odwracanie rekurencji (2.2) (przechodzenie od tyłu), tj. obliczanie najpierw $C_{0,0}$, następnie $C_{1,1}$ itd.

Rozważmy teraz odległość Hamminga, gdzie $s_{i':i} \to t_{j':j}$ to zamiany znaków o koszcie równym jeden. Stąd,

$$\delta(s_{i':i} \to t_{j':j}) = [s_{i':i} \neq t_{j':j}], \tag{2.3}$$

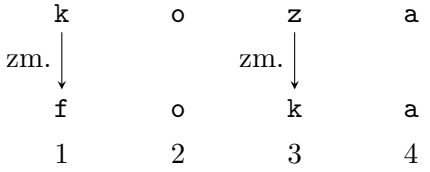
gdzie $[X]$ jest równe jeden, gdy warunek X jest spełniony, zero w przeciwnym przypadku. Co więcej, w tym przypadku możliwa jest tylko jedna kombinacja i' oraz j', mianowicie $i' = i$ oraz $j' = j$. Dalej, odległość ta jest zdefiniowana jedynie dla $|s| = |t|$, zatem $C_{i,j}$ może być policzone jedynie dla $i = j$. Wówczas definicja odległości Hamminga nie jest rekurencyjna i można ją zapisać następująco:

Definicja 2.8. Odległością Hamminga *nazywamy [11]:*

$$d_{\text{hamming}}(s,t) = \begin{cases} \sum_{i=1}^{|s|} \delta(s_i \to t_i) = \sum_{i=1}^{|s|} [s_i \neq t_i], & \textit{gdy } |s| = |t|, \\ \infty, & \textit{w przeciwnym przypadku.} \end{cases}$$

Odległość Hamminga zlicza liczbę indeksów (zob. Rys. 2.2), na których dwa napisy mają różny znak. Odległość ta przyjmuje wartości ze zbioru $\{0, \ldots, |s|\}$, gdy $|s| = |t|$, natomiast jest równa nieskończoności, gdy napisy mają różne długości.

Przykład 2.4. Odległość Hamminga między słowami `koza` i `foka` wynosi $d_{\text{hamming}}(\texttt{koza}, \texttt{foka}) = 2$, natomiast między słowami `kozak` i `foczka` wynosi ona $d_{\text{hamming}}(\texttt{kozak}, \texttt{foczka}) = \infty$, gdyż $|\texttt{kozak}| \neq |\texttt{foczka}|$.



Rysunek 2.2: Przykład dopasowania przy pomocy odległości Hamminga między napisami `koza` i `foka`.

Rozważmy teraz odległość najdłuższego wspólnego podnapisu (ang. *longest common substring*), gdzie $s_{i':i} \to t_{j':j}$ to wstawienia i usunięcia znaków o koszcie równym jeden. Wówczas istnieją dwie kombinacje i' oraz j' z ogólnej rekurencji (2.2), odpowiadające usunięciu i wstawieniu, odpowiednio:

- $i' = i - 1$ oraz $j' = j$,
- $i' = i$ oraz $j' = j - 1$.

Definicja 2.9. *Uwzględniając powyższe uproszczenia, możemy następująco przepisać ogólną postać rekurencji* (2.2) *dla odległości najdłuższego wspólnego podnapisu:*

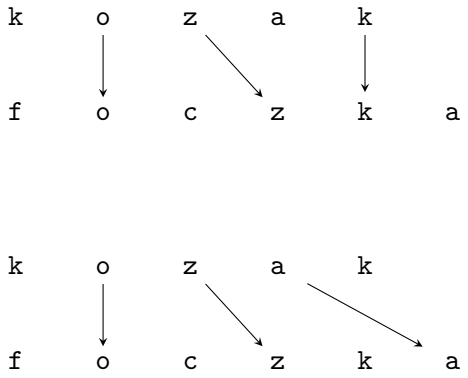
$$C_{i,j} = \begin{cases} 0, & \textit{gdy } i = j = 0, \\ C_{i-1,j-1}, & \textit{gdy } s_i = t_j, i, j > 0, \\ 1 + \min\{C_{i-1,j}, C_{i,j-1}\}, & \textit{wpp..} \end{cases}$$

Odległość najdłuższego wspólnego podnapisu przyjmuje wartości ze zbioru $\{0, |s| + |t|\}$, przy czym maksimum jest osiągane, gdy s i t nie mają ani jednego wspólnego znaku. Odległość tę oznaczamy przez d_{lcs}.

Przykład 2.5. Odległość najdłuższego wspólnego podnapisu między napisami `kozak` i `foczka` wynosi: $d_{\text{lsc}}(\texttt{kozak}, \texttt{foczka}) = 5$, bo $\texttt{kozak} \xrightarrow[1]{us.\ k} \texttt{ozak} \xrightarrow[1]{us.\ a} \texttt{ozk} \xrightarrow[1]{wst.\ f} \texttt{fozk} \xrightarrow[1]{wst.\ c} \texttt{foczk} \xrightarrow[1]{wst.\ a} \texttt{foczka}$.

Powyższy przykład pokazuje, że w ogólności nie ma unikalnej najkrótszej drogi transformacji jednego napisu w drugi, gdyż można zamienić kolejność usuwania (lub wstawiania) znaków i również uzyskać odległość równą 5. Można również usunąć z napisu znak `k` zamiast `a`, otrzymując taką samą odległość między napisami.

Jak sugeruje nazwa, odległość najdłuższego wspólnego podnapisu, ma też inną interpretację. Poprzez wyrażenie *najdłuższy wspólny podnapis* rozumiemy najdłuższy ciąg utworzony przez sparowanie znaków z s i t nie zmieniając ich porządku. Wówczas odległość ta jest rozumiana jako liczba niesparowanych znaków z obu napisów. W powyższym przykładzie może to być przedstawione jak na Rys. 2.3.



Rysunek 2.3: Przykład odległości najdłuższego wspólnego podnapisu między napisami `kozak` i `foczka`.

Jak widać, znaki `k`, `a`, `f`, `c` i `a` w pierwszym przypadku oraz `k`, `k`, `f`, `c` i `k` w drugim, pozostają bez pary, dając odległość równą 5.

Przejdźmy do odległości Levenshteina. Odległość ta dopuszcza, oprócz usunięć i wstawień, także zamiany znaków. Istnieją zatem trzy kombinacje i' oraz j' z ogólnej rekurencji (2.3):

- $i' = i - 1$ oraz $j' = j$,
- $i' = i$ oraz $j' = j - 1$,
- $i' = i - 1$ oraz $j' = j - 1$.

Definicja 2.10. *Stąd ogólna postać rekurencji* (2.2) *dla odległości Levenshteina może zostać przepisana następująco:*

$$C_{i,j} = \min \begin{cases} 0, & gdy\ i = j = 0, \\ C_{i-1,j} + 1, & gdy\ i > 0, \\ C_{i,j-1} + 1, & gdy\ j > 0, \\ C_{i-1,j-1} + [s_i \neq t_j], & gdy\ i, j > 0. \end{cases} \tag{2.4}$$

Odległość Levenshteina oznaczamy przez d_{lv}.

Przykład 2.6. Odległość Levenshteina między napisami `kozak` i `foczka` wynosi: $d_{\mathrm{lv}}(\texttt{kozak}, \texttt{foczka}) = 4$, bo $\texttt{kozak} \xrightarrow[1]{zm.\ k\ na\ f} \texttt{fozak} \xrightarrow[1]{wst.\ c} \texttt{foczak} \xrightarrow[1]{zm.\ a\ na\ k} \texttt{foczkk} \xrightarrow[1]{zm.\ k\ na\ a} \texttt{foczka}$.

Powyższy przykład ilustruje dodatkową elastyczność w porównaniu do odległości najdłuższego wspólnego podnapisu, bowiem daje ona mniejszą wartość odległości między napisami, jako że w przypadku pierwszego znaku potrzebujemy jedynie zamiany, zamiast wstawienia i usunięcia [31]. Co więcej, ścieżka edycyjna między tymi słowami może być inna i zawierać usunięcie i wstawienie zamiast dwóch ostatnich zamian znaków.

Przypomnijmy, że mówimy o ważonej odległości, gdy zmienimy koszty poszczególnych operacji na różne od jeden. Gdy za koszt przyjmiemy np. $(0, 1, 1, 0, 3)$ dla usunięć, wstawień i zamian znaków odpowiednio, to uogólniona odległość Levenshteina między napisami `kozak` i `foczka` wynosi: $d_{\mathrm{lv}}(\texttt{kozak}, \texttt{foczka}) = 1,9$, bo $\texttt{kozak} \xrightarrow[0,3]{zm.\ k\ na\ f} \texttt{fozak} \xrightarrow[1]{wst.\ c} \texttt{foczak} \xrightarrow[0,3]{zm.\ a\ na\ k} \texttt{foczkk} \xrightarrow[0,3]{zm.\ k\ na\ a} \texttt{foczka}$.

Ważona odległość Levenshteina spełnia definicję metryki, gdy koszt usunięcia jest równy kosztowi wstawienia znaku. W przeciwnym przypadku nie spełnia ona założenia o symetrii. Jednakowoż, symetria zostaje zachowana przy jednoczesnej zamianie s i t oraz kosztów usunięcia i wstawienia znaku, jako że liczba usunięć znaków przy przetwarzaniu napisu s w napis t jest równa liczbie wstawień znaków przy transformacji napisu t w napis s [31]. Dobrze obrazuje to następujący przykład:

Przykład 2.7. Przyjmijmy za koszt usunięcia, wstawienia i zamiany znaku odpowiednio $(0, 1, 1, 0, 3)$. Wówczas uogólniona odległość Levenshteina dla napisów `kozak` i `foczka` wynosi:

$$d_{\mathrm{lv}}(\texttt{kozak}, \texttt{foczka}) = 1,9, \tag{2.5}$$

gdyż

$$\texttt{kozak} \xrightarrow[0,3]{zm.\ k\ na\ f} \texttt{fozak} \xrightarrow[1]{wst.\ c} \texttt{foczak} \xrightarrow[0,3]{zm.\ a\ na\ k} \texttt{foczkk} \xrightarrow[0,3]{zm.\ k\ na\ a} \texttt{foczka},$$

natomiast

$$d_{\mathrm{lv}}(\texttt{foczka}, \texttt{kozak}) = 1, \tag{2.6}$$

gdyż

$$\texttt{foczka} \xrightarrow[0,3]{zm.\ f\ na\ k} \texttt{koczka} \xrightarrow[0,1]{us.c} \texttt{kozka} \xrightarrow[0,3]{zm.\ k\ na\ a} \texttt{kozaa} \xrightarrow[0,3]{zm.\ a\ na\ k} \texttt{kozak}.$$

Gdy za koszty przyjmiemy $(1, 0, 1, 0, 3)$, to uogólniona odległość Levenshteina wynosi:

$$d_{\mathrm{lv}}(\texttt{kozak}, \texttt{foczka}) = 1,$$

gdyż

$$\texttt{kozak} \xrightarrow[0,3]{zm.\ k\ na\ f} \texttt{fozak} \xrightarrow[0,1]{wst.\ c} \texttt{foczak} \xrightarrow[0,3]{zm.\ a\ na\ k} \texttt{foczkk} \xrightarrow[0,3]{zm.\ k\ na\ a} \texttt{foczka},$$

czyli analogicznie, jak w przypadku (2.6). Natomiast

$$d_{\mathrm{lv}}(\texttt{foczka}, \texttt{kozak}) = 1,9,$$

bo

$$\texttt{foczka} \xrightarrow[0{,}3]{zm.\ f\ na\ k} \texttt{koczka} \xrightarrow[1]{us.c} \texttt{kozka} \xrightarrow[0{,}3]{zm.\ k\ na\ a} \texttt{kozaa} \xrightarrow[0{,}3]{zm.\ a\ na\ k} \texttt{kozak},$$

czyli analogicznie, jak w przypadku (2.5).

Lemat 2.5. *Ścisła odległość Levenshteina o jednostkowym koszcie operacji bazowych jest równa nieścisłej odległości Levenshteina o jednostkowym koszcie operacji bazowych [6].*

Powyższe wynika natychmiast z obserwacji, że optymalna ścieżka edycyjna zawiera jednoznakowe usunięcia, wstawienia oraz zamiany, które nigdy nie modyfikują podnapisu więcej niż raz.

Zgodnie z lematem 2.5 nieścisła odległość Levenshteina jest równa ścisłej odległości Levenshteina. Z drugiej strony, ścisła odległość edycyjna jest równa kosztowi optymalnego dopasowania. Stąd rekurencja (2.2) liczy nieścisłą odległość Levenshteina. Spójrzmy na następujące bezpośrednie uogólnienie rekurencji (2.4), dodające transpozycję do zbioru bazowych operacji edycyjnych [6]:

$$C_{i,j} = \min \begin{cases} 0, & \text{gdy } i = j = 0 \\ C_{i-1,j} + 1, & \text{gdy } i > 0 \\ C_{i,j-1} + 1, & \text{gdy } j > 0 \\ C_{i-1,j-1} + [s_i \neq t_j], & \text{gdy } i, j > 0 \\ C_{i-2,j-2} + 1, & \text{gdy } s_i = t_{j-1}, s_{i-i} = t_j \text{ oraz } i, j > 1 \end{cases} \tag{2.7}$$

Rekurencja (2.7) liczy czyli ścisłą odległość Damerau-Levenshteina, która nie zawsze jest równa odległości Damerau-Levenshteina. Dla przykładu, odległość między napisami `ba` i `acb` wyliczona przy pomocy rekurencji (2.7) jest równa trzy, natomiast odległość Damerau-Levenshteina między tymi napisami wynosi dwa (zob. poniżej).

Lemat 2.6. *Nieścisła odległość Damerau-Levenshteina oraz ścisła odległość Damerau- Levenshteina są różnymi funkcjami. Co więcej, ścisła odległość Damerau-Levenshteina nie jest metryką, gdyż nie jest subaddytywna [6].*

Dowód. Ścisła odległość Damerau-Levenshteina traktuje transpozycję (tj. zamianę dwóch przylegających do siebie znaków) jako bazową operację edycyjną. Aby udowodnić lemat podamy przykład, w którym zakaz modyfikacji znaków już stransponowanych odróżnia odległość Damerau-Levenshteina od ścisłej odległości Damerau-Levenshteina [6]. ■

Ścisła odległość Damerau-Levenshteina nie spełnia nierówności trójkąta, gdyż

$$\texttt{ba} \xrightarrow[1]{transp.\ b\ i\ a} \texttt{ab} + \texttt{ab} \xrightarrow[1]{wst.c} \texttt{acb},$$

natomiast

$$\texttt{ba} \xrightarrow[1]{us.\ b} \texttt{a} \xrightarrow[1]{wst.\ c} \texttt{ac} \xrightarrow[1]{wst.\ b} \texttt{acb},$$

zatem

$$2 = ED(\texttt{ba}, \texttt{ab}) + ED(\texttt{ab}, \texttt{acb}) \leq ED(\texttt{ba}, \texttt{acb}) = 3.$$

Ponieważ ścisła i nieścisła odległość Damerau-Levenshteina są różnymi funkcjami, tę pierwszą nazywa się często *odległością optymalnego dopasowania napisów*. Od tego momentu w

niniejszej pracy ścisłą odległość Damerau-Levenshteina nazywamy odległością optymalnego dopasowania napisów, natomiast nieścisłą odległość Damerau-Levenshteina nazywamy odległością Damerau-Levenshteina [31].

Rekurencyjna definicja odległości Damerau-Levenshteina została po raz pierwszy podana przez Lowrance'a i Wagnera [34]. W ich definicji zamiana zostaje zastąpiona przez minimalizację po możliwych transpozycjach między danym znakiem a wszystkimi nie przetransformowanymi znakami, przy czym koszt transpozycji wzrasta wraz z odległością między transponowanymi znakami [31]. Innymi słowy, do $\mathbb{B}$ należą wstawienia, usunięcia, zamiany oraz operacje $axb \to bya$ o koszcie równym $|x| + |y| + 1$ [6]. Mając tak zdefiniowane $\mathbb{B}$ ogólna rekurencja (2.2) dla odległości Damerau-Levenshteina przedstawia się następująco:

$$C_{i,j} = \min \begin{cases} 0, & \text{gdy } i = j = 0 \\ C_{i-1,j} + 1, & \text{gdy } i > 0 \\ C_{i,j-1} + 1, & \text{gdy } j > 0 \\ C_{i-1,j-1} + [s_i \neq t_j], & \text{gdy } i, j > 0 \\ \min\limits_{\substack{0<i'<i,\ 0<j'<j \\ s_i=t_{j'},\ s_{i'}=t_j}} \{C_{i'-1,j'-1} + (i - i') + (j - j') - 1\} & \end{cases} \tag{2.8}$$

Co więcej, Lowrance i Wagner wykazali, że wewnętrzne minimum w rekurencji (2.8) jest osiągane dla największych $i' < i$ oraz $j' < j$, które spełniają $s_i = t_{j'}$ oraz $s_{i'} = t_j$. Odległość Damerau-Levenshteina oznaczamy przez d_{dl}.

Przykład 2.8. Odległość optymalnego dopasowania napisów oraz Damerau-Levenshteina między napisami `kozak` i `foczka` wynosi 3, bo `kozak` $\xrightarrow[1]{zm.\ k\ na\ f}$ `fozak` $\xrightarrow[1]{wst.\ c}$ `foczak` $\xrightarrow[1]{transp.\ a\ i\ k}$ `foczka`.

W przypadku odległości Levenshteina, optymalnego dopasowania napisów oraz Damerau-Levenshteina, maksymalna odległość między napisami s i t wynosi $\max\{|s|, |t|\}$. Jednak warto zauważyć, że gdy liczba dopuszczalnych operacji edycyjnych rośnie, to liczba dopuszczalnych ścieżek między napisami wzrasta, co pozwala czasem zmniejszyć odległość między napisami. Dlatego relację między zaprezentowanymi powyżej odległościami można podsumować następująco [31]:

$$\left.\begin{array}{r} \infty(\geq |s|) \geq d_{\text{hamming}}(s,t) \\ |s| + |t| \geq d_{\text{lcs}}(s,t) \\ \max\{|s|, |t|\} \end{array}\right\} \geq d_{\text{lv}}(s,t) \geq d_{osa}(s,t) \geq d_{dl}(s,t) \geq 0.$$

Jako że odległości Hamminga i najdłuższego wspólnego podnapisu nie mają wspólnych bazowych operacji edycyjnych, to nie ma pomiędzy nimi porządku relacyjnego. Górne ograniczenie $|s|$ odległości Hamminga jest zachowane jedynie, gdy $|s| = |t|$.

2.2.2. Odległości oparte na q-gramach

Przypomnijmy, że q-gramem nazywamy napis składający się z q kolejnych (przylegających) znaków. q-gramy związane z napisem s są otrzymywane przez przesuwanie przez napis s „okna" o szerokości q znaków i zapisaniu występujących q-gramów. Przykładowo digramy

napisu `ela` to `el` i `la`. Oczywiście taka procedura nie ma sensu, gdy $q > |s|$ lub gdy $q = 0$. Z tego powodu definiujemy następujące przypadki brzegowe dla wszystkich odległości $d_q(s,t)$ opartych na q-gramach:

$$\begin{aligned} &d_q(s,t) = \infty, \text{ gdy } q > \min\{|s|,|t|\},\\ &d_0(s,t) = \infty, \text{ gdy } |s| + |t| > 0,\\ &d_0(\varepsilon,\varepsilon) = 0. \end{aligned}$$

Najprostszą odległością między napisami, opartą na q-gramach, otrzymuje się przez wypisanie unikalnych q-gramów w obu napisach i porównanie które są wspólne. Jeśli przez $\mathcal{Q}(s,q)$ oznaczymy zbiór unikalnych q-gramów występujących w napisie s, to możemy zdefiniować odległość Jaccarda [31]:

Definicja 2.11. *Niech $\mathcal{Q}(s,q)$ oznacza zbiór unikalnych q-gramów występujących w napisie s. Wówczas* odległość Jaccarda*, d_{jac}, między napisami s i t definiuje się jako:*

$$d_{\text{jac}}(s,t,q) = 1 - \frac{|\mathcal{Q}(s,q) \cap \mathcal{Q}(t,q)|}{|\mathcal{Q}(s,q) \cup \mathcal{Q}(t,q)|},$$

gdzie $|\cdot|$ oznacza liczność zbioru.

Odległość Jaccarda przyjmuje wartości z przedziału $[0,1]$, gdzie 0 odpowiada pełnemu pokryciu zbiorów, tj. $\mathcal{Q}(s,q) = \mathcal{Q}(t,q)$, natomiast 1 oznacza puste przecięcie, tj. $\mathcal{Q}(s,q) \cap \mathcal{Q}(t,q) = \emptyset$.

Przykład 2.9. Odległość Jaccarda między napisami `papaja` i `japa` dla $q = 2$ wynosi: $d_{\text{jac}}(\texttt{papaja}, \texttt{japa}, 2) = 0,25$, bo $\mathcal{Q}(\texttt{papaja}, 2) = \{\texttt{pa}, \texttt{ap}, \texttt{aj}, \texttt{ja}\}$, a $\mathcal{Q}(\texttt{japa}, 2) = \{\texttt{ja}, \texttt{ap}, \texttt{pa}\}$, więc odległość wynosi $1 - \frac{3}{4} = 0,25$.

Inną odległością opartą na q-gramach jest odległość q-gramowa. Otrzymuje się ją przez wylistowanie q-gramów występujących w obu napisach i policzenie q-gramów, które nie są wspólne dla obu napisów [31]. Formalnie można to zapisać następująco:

Definicja 2.12. *Niech $s = s_1 s_2 \dots s_n$ będzie napisem z Σ^* i niech $x \in \Sigma^q$ będzie q-gramem. Jeśli $s_i s_{i+1} \dots s_{i+q-1} = x$ dla pewnego i, to x* wystąpiło *w s. Niech $\mathbf{v}(s,q)$ będzie wektorem o długości $|\Sigma|^q$, którego zmienne oznaczają liczbę wystąpień wszystkich możliwych q-gramów z Σ^q w s. Niech $s,t \in \Sigma^*$ oraz $q > 0$ będzie liczbą naturalną.* Odległość q-gramową *między napisami s i t definiuje się następująco [30]:*

$$d_{\text{qgram}}(s,t,q) = \|\mathbf{v}(s,q) - \mathbf{v}(t,q)\|_1 = \sum_{i=1}^{|\Sigma|^q} |v_i(s,q) - v_i(t,q)|. \tag{2.9}$$

Wzór (2.9) definiuje odległość q-gramową między napisami s i t jako odległość L_1 pomiędzy $\mathbf{v}(s,q)$ i $\mathbf{v}(t,q)$. Zauważmy, że zamiast sprawdzać wystąpienie wszystkich możliwych q-gramów z Σ^q w napisach s i t, wystarczy policzyć jedynie liczbę faktycznie występujących q-gramów w obu napisach, by obliczyć odległość q-gramową [31].

Przykład 2.10. Niech $\Sigma = \{\mathtt{a}, \mathtt{j}, \mathtt{p}\}$. Wówczas odległość q-gramowa między napisami `papaja` i `japa` dla $q = 2$ wynosi: $d_{\text{qgram}}(\mathtt{papaja}, \mathtt{japa}, 2) = 2$. Wszystkie możliwe digramy występujące w napisach `papaja` i `japa` to `aj`, `ap`, `ja` i `pa`. Zatem $\mathbf{v}(\mathtt{papaja}, 2) = (1, 1, 1, 2)$, a $\mathbf{v}(\mathtt{japa}, 2) = (0, 1, 1, 1)$. Stąd $d_{\text{qgram}}(\mathtt{papaja}, \mathtt{japa}, 2) = \|(1, 1, 1, 2) - (0, 1, 1, 1)\|_1 = 2$.

Maksymalna liczba wystąpień różnych q-gramów w napisie s wynosi $|s| - q - 1$. Stąd maksymalna odległość q-gramowa między napisami s i t wynosi $|s| + |t| - 2q - 2$, osiągana, gdy s i t nie mają wspólnych q-gramów [31].

Skoro zdefiniowana została odległość q-gramowa w języku wektorów, każda miara podobieństwa w (całkowitej) przestrzeni wektorowej może zostać zastosowana. Przykładowo można zdefiniować *odległość cosinusową* między napisami s i t:

$$d_{\cos}(s, t, q) = 1 - \frac{\mathbf{v}(s, q) \cdot \mathbf{v}(t, q)}{\|\mathbf{v}(s, q)\|_2 \|\mathbf{v}(t, q)\|_2}, \tag{2.10}$$

gdzie $\|\cdot\|_2$ oznacza zwykłą normę Euklidesową. Odległość cosinusowa wynosi zero, gdy $s = t$ oraz jeden, gdy s i t nie mają wspólnych q-gramów. Odległość ta powinna być interpretowana jako kąt pomiędzy $\mathbf{v}(s, q)$ i $\mathbf{v}(t, q)$, jako że drugie wyrażenie równania (2.10) przedstawia cosinus kąta między dwoma wektorami.

Przykład 2.11. Niech $\Sigma = \{\mathtt{a}, \mathtt{j}, \mathtt{p}\}$. Wówczas odległość cosinusowa między napisami `papaja` i `japa` dla $q = 2$ wynosi: $d_{\cos}(\mathtt{papaja}, \mathtt{japa}, 2) \approx 0,127$, bo $\mathbf{v}(\mathtt{papaja}, 2) = (1, 1, 1, 2)$, a $\mathbf{v}(\mathtt{japa}, 2) = (0, 1, 1, 1)$ (zob. przykład 2.10), więc $d_{\cos}(\mathtt{papaja}, \mathtt{japa}, 2) = 1 - \frac{4}{\sqrt{3} \cdot \sqrt{7}} \approx 0,127$.

Wszystkie trzy odległości oparte na q-gramach są nieujemne i symetryczne. Odległości Jaccarda i q-gramowa spełniają również nierówność trójkąta (dowód dla tej pierwszej poniżej), w odróżnieniu od odległości cosinusowej. Żadna z powyższych miar nie spełnia warunku identyczności, ponieważ zarówno $\mathcal{Q}(s, q)$, jak i $\mathbf{v}(s, q)$ jest funkcją wiele-do-jednego. Jako przykład, zauważmy, że $\mathcal{Q}(\mathtt{abaca}, 2) = \mathcal{Q}(\mathtt{acaba}, 2)$ oraz $\mathbf{v}(\mathtt{abaca}, 2) = \mathbf{v}(\mathtt{acaba}, 2)$, więc $d_{\text{jac}}(\mathtt{abaca}, \mathtt{acaba}, 2) = d_{\text{qgram}}(\mathtt{abaca}, \mathtt{acaba}, 2) = d_{\cos}(\mathtt{abaca}, \mathtt{acaba}, 2) = 0$. Innymi słowy, odległość oparta na q-gramach równa zero, nie gwarantuje, że $s = t$. Jeszcze inaczej, odległość Jaccarda i q-gramowa są pseudometrykami. Inne własności $\mathbf{v}(s, q)$ można znaleźć w [30].

Udowodnimy teraz, że odległość Jaccarda spełnia nierówność trójkąta.

Lemat 2.7. *Niech $A := \mathcal{Q}(s, q), B := \mathcal{Q}(t, q)$ dla ustalonego q. Wówczas odległość Jaccarda można napisać następująco:*

$$\begin{aligned} d_{\text{jac}}(s, t, q) =& 1 - \frac{|\mathcal{Q}(s, q) \cap \mathcal{Q}(t, q)|}{|\mathcal{Q}(s, q) \cup \mathcal{Q}(t, q)|} = \\ & 1 - \frac{|A \cap B|}{|A \cup B|} =: d(A, B). \end{aligned}$$

d spełnia nierówność trójkąta:

$$\forall A, B, C \quad d(A, C) \geq d(A, B) + d(B, C).$$

Dowód. Przypuśćmy, że nierówność nie jest spełniona, tj. istnieją zbiory A, B oraz C, takie, że $d(A, C) > d(A, B) + d(B, C)$. Wówczas

$$1 - \frac{|A \cap C|}{|A \cup C|} > 1 - \frac{|A \cap B|}{|A \cup B|} + 1 - \frac{|B \cap C|}{|B \cup C|}$$

lub równoważnie

$$\frac{|A \cap B|}{|A \cup B|} + \frac{|B \cap C|}{|B \cup C|} - \frac{|A \cap C|}{|A \cup C|} > 1. \tag{2.11}$$

Ponieważ postulujemy, że nierówność trójkąta nie jest spełniona, chcemy aby lewa strona ostatniej nierówności była jak najmniejsza. Stąd wystarczy rozważyć przypadki, gdy $A \subseteq B$ lub $B \subseteq A$, gdyż w przeciwnym przypadku, dla $B' = A \cap B$ mamy

$$\frac{|A \cap B'|}{|A \cup B'|} = \frac{|A \cap B|}{|A \cup (A \cap B)|} \geq \frac{|A \cap B|}{|A \cup B|}.$$

Zastępujemy zatem B przez B' i dostajemy:

$$\frac{|A \cap B'|}{|A \cup B'|} + \frac{|B \cap C|}{|B \cup C|} - \frac{|A \cap C|}{|A \cup C|} \geq \frac{|A \cap B|}{|A \cup B|} + \frac{|B \cap C|}{|B \cup C|} - \frac{|A \cap C|}{|A \cup C|}.$$

Analogicznie możemy rozważyć jedynie przypadki gdy $C \subseteq B$ lub $B \subseteq C$. Przeanalizujmy teraz cztery przypadki.

Przypadek 1. $A \subseteq B$ oraz $C \subseteq B$. Wówczas

$$\begin{aligned}\frac{|A \cap B|}{|A \cup B|} + \frac{|B \cap C|}{|B \cup C|} - \frac{|A \cap C|}{|A \cup C|} &= \frac{|A|}{|B|} + \frac{|C|}{|B|} - \frac{|A \cap C|}{|A \cup C|} = \frac{|A| + |C|}{|B|} - \frac{|A \cap C|}{|A \cup C|} = \\ &= \frac{|A \cup C| + |A \cap C|}{|B|} - \frac{|A \cap C|}{|A \cup C|} \leq \frac{|A \cup C|}{|B} + \frac{|A \cap C|}{|B|} - \frac{|A \cap C|}{|B|} = \frac{|A \cup C|}{|B} \leq 1\end{aligned}$$

Przypadek 2. $A \subseteq B$ oraz $B \subseteq C$. Wówczas

$$\frac{|A \cap B|}{|A \cup B|} + \frac{|B \cap C|}{|B \cup C|} - \frac{|A \cap C|}{|A \cup C|} = \frac{|A|}{|B|} + \frac{|B|}{|C|} - \frac{|A|}{|C|} \leq \frac{|A|}{|C|} + \frac{|B|}{|C|} - \frac{|A|}{|C|} = \frac{|B|}{|C|} \leq 1$$

Przypadek 3. $C \subseteq B$ oraz $B \subseteq A$. Wówczas

$$\frac{|A \cap B|}{|A \cup B|} + \frac{|B \cap C|}{|B \cup C|} - \frac{|A \cap C|}{|A \cup C|} = \frac{|B|}{|A|} + \frac{|C|}{|B|} - \frac{|C|}{|A|} \leq \frac{|B|}{|A|} + \frac{|C|}{|A|} - \frac{|C|}{|A|} = \frac{|B|}{|A|} \leq 1$$

Przypadek 4. $B \subseteq A$ oraz $B \subseteq C$. Wówczas

$$\frac{|A \cap B|}{|A \cup B|} + \frac{|B \cap C|}{|B \cup C|} - \frac{|A \cap C|}{|A \cup C|} = \frac{|B|}{|A|} + \frac{|B|}{|C|} - \frac{|A \cap C|}{|A \cup C|} \leq \frac{|A \cap C|}{|A|} + \frac{|B|}{|C|} - \frac{|A \cap C|}{|A|} = \frac{|B|}{|C|} \leq 1$$

Wszystkie cztery powyższe przypadki są w sprzeczności ze stwierdzeniem, że wyrażenie (2.11) jest ostro większe od 1. Stąd założenie jest nieprawdziwe, więc nierówność trójkąta jest spełniona [35]. ■

2.2.3. Miary heurystyczne

Odległość Jaro została stworzona w amerykańskim Bureau of the Census (rządowa agencja, która jest odpowiedzialna m.in. za spis ludności Stanów Zjednoczonych) w celu połączenia rekordów, które były wpisane w niewłaściwe pola formularza oraz zlikwidowaniu literówek. Pierwszy publiczny opis tej odległości pojawił się w instrukcji obsługi [13], co może wyjaśniać dlaczego nie jest rozpowszechniona w literaturze informatycznej. Jednak odległość ta została

skutecznie zastosowana w statystycznych problemach dopasowania w przypadku dość krótkich napisów, głównie imion, nazwisk oraz danych adresowych [31].

Rozumowanie stojące za odległością Jaro jest następujące: błędny znak oraz transpozycje znaków są spowodowane błędem przy wpisywaniu, ale mało prawdopodobne jest znalezienie błędnego znaku w miejscu odległym od zamierzonego, żeby mogło to być spowodowane błędem przy wpisywaniu. Stąd odległość Jaro mierzy liczbę wspólnych znaków w dwóch napisach, które nie są zbyt odległe od siebie i dodaje karę za dopasowanie znaków, które są stransponowane. Formalna definicja wygląda następująco [31]:

Definicja 2.13. *Niech $\lfloor x \rfloor$ oznacza największą liczbę całkowitą, nie większą niż x. Niech s i t będą napisami z Σ^*. Niech m oznacza liczbę wspólnych znaków z s i t, przy czym zakładając, że $s_i = t_j$, to znak ten jest* wspólny *dla obu napisów, jeśli:*

$$|i - j| < \left\lfloor \frac{max\{|s|, |t|\}}{2} \right\rfloor$$

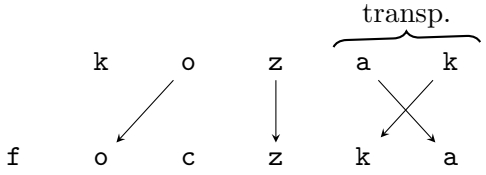
i każdy znak z s może być wspólny ze znakiem z t tylko raz. W końcu, jeśli s' i t' są podnapisami utworzonymi z s i t poprzez usunięcie znaków, które nie są wspólne dla obu napisów, to T jest liczbą transpozycji potrzebnych to otrzymania t' z s'. Transpozycje znaków nieprzylegających są dozwolone.

Wówczas odległość Jaro *definiuje się jako:*

$$d_{\text{jaro}}(s,t) = \begin{cases} 0, & gdy\ s = t = \varepsilon, \\ 1, & gdy\ m = 0\ i\ |s| + |t| > 0, \\ 1 - \frac{1}{3}(\frac{m}{|s|} + \frac{m}{|t|} + \frac{m-T}{m}) & w\ przeciwnym\ przypadku. \end{cases} \tag{2.12}$$

Odległość Jaro przyjmuje wartości z przedziału $[0, 1]$, gdzie zero oznacza, że $s = t$, natomiast jeden wskazuje na kompletną odmienność napisów z $m = T = 0$.

Przykład 2.12. Odległość Jaro między napisami `kozak` i `foczka` wynosi: $d_{\text{jaro}}(\texttt{kozak}, \texttt{foczka}) \approx 0,261$, bo liczba wspólnych znaków wynosi $m = 4$, a liczba potrzebnych transpozycji wynosi $T = 1$ (zob. Rys. 2.4), co daje odległość równą $d_{\text{jaro}}(\texttt{kozak}, \texttt{foczka}) = 1 - \frac{1}{3}(\frac{3}{5} + \frac{4}{6} + \frac{3}{4}) = \frac{47}{180} \approx 0,261$.



Rysunek 2.4: Przykład odległości Jaro między napisami `kozak` i `foczka`.

Winkler rozszerzył odległość Jaro przez włączenie dodatkowej kary za błędny znak wśród pierwszych czterech znaków napisu [31]:

Definicja 2.14. *Niech s i t będą napisami z Σ^*, $\ell(s,t)$ oznacza długość najdłuższego wspólnego przedrostka, mającego maksymalnie cztery znaki i niech p będzie liczbą z przedziału $[0, \frac{1}{4}]$. Wówczas odległość Jaro-Winklera dana jest wzorem [36]:*

$$d_{\text{jw}}(s,t,p) = d_{\text{jaro}}(s,t)[1 - p\ell(s,t)] \tag{2.13}$$

Czynnik p określa jak bardzo różnice w czterech pierwszych znakach w obu napisach wpływają na odległość między nimi. Zmienna p jest liczbą z przedziału $[0, \frac{1}{4}]$, by mieć pewność, że odległość Jaro-Winklera miała wartości w przedziale $[0, 1]$ ($0 \leq d_{jw}(s,t) \leq 1$). Jeśli $p = 0$, to odległość ta redukuje się do odległości Jaro i wszystkie znaki wnoszą taki sam wkład do funkcji odległości. Jeśli $p = \frac{1}{4}$, to odległość Jaro-Winklera jest równa zero nawet wówczas gdy tylko cztery pierwsze znaki w obu napisach pokrywają się. Powód jest taki, że podobno ludzie są mniej skłonni do popełniania błędów w czterech pierwszych znakach lub też są one lepiej zauważalne, więc różnice w pierwszych czterech znakach wskazują na większe prawdopodobieństwo, że dwa napisy są rzeczywiście różne [31]. Winkler [36] używał w swoich badaniach $p = 0,1$ i zauważył lepsze rezultaty niż dla $p = 0$.

Przykład 2.13. Odległość Jaro-Winklera między napisami `faktura` i `faktyczny` dla $p = 0$, $p = 0,1$ oraz $p = 0,25$ wynosi odpowiednio:

$$d_{\text{jw}}(\texttt{faktura}, \texttt{faktyczny}, p = 0,00) \approx 0,328 = d_{\text{jaro}}(\texttt{faktura}, \texttt{faktyczny})$$
$$d_{\text{jw}}(\texttt{faktura}, \texttt{faktyczny}, p = 0,10) \approx 0,197$$
$$d_{\text{jw}}(\texttt{faktura}, \texttt{faktyczny}, p = 0,25) = 0$$

Łatwo zauważyć z równań (2.12) i (2.13), że odległości Jaro i Jaro-Winklera, dla $p \neq \frac{a}{4}$, są nieujemne, symetryczne i spełniają warunek identycznościowy. Nierówność trójkąta w obu przypadkach nie jest jednak spełniona. Rozważmy następujący przykład: $s = \texttt{ab}, t = \texttt{cb}, u = \texttt{cd}$. Jako że napisy s i u nie mają wspólnych znaków, to odległość Jaro między nimi wynosi $d_{\text{jaro}}(s,u) = 1$, podczas gdy $d_{\text{jaro}}(s,t) = d_{\text{jaro}}(t,u) = \frac{1}{3}$, więc w tym przypadku $d_{jaro}(s,u)$ jest większe od $d_{\text{jaro}}(s,t) + d_{jaro}(t,u)$. Z tego łatwo zauważyć, że odległość Jaro-Winklera nie spełnia nierówności trójkąta dla tego samego przykładu dla $p \in [0, \frac{1}{4}]$ [31].

W niniejszym rozdziale przedstawiono odległości określone na przestrzeni ciągów znaków. Mając do wyboru wachlarz różnych funkcji nasuwa się pytanie, której użyć. Ostateczna decyzja zależy od konkretnego przypadku, jednak istnieją pewne ogólne reguły. Wybór pomiędzy odległościami opartymi na operacjach edycyjnych i q-gramach z jednej strony, a miarami heurystycznymi z drugiej zależy w dużej mierze od długości napisów – te ostatnie są dedykowane krótszym napisom takim jak np. dane osobowe. W odróżnieniu od odległości opartych na operacjach edycyjnych i miarach heurystycznych, odległości oparte na q-gramach można łatwo policzyć dla bardzo długich tekstów, jako że liczba q-gramów możliwych do utworzenia z języka naturalnego (dla niezbyt małego q, tj. $q \geq 3$) jest z reguły o wiele mniejsza niż liczba q-gramów, którą można otrzymać z całego alfabetu. Wybór pośród odległości opartych na operacjach edycyjnych zależy przede wszystkim od dokładności jaką chce się otrzymać. Przykładowo do wyszukiwania haseł w słowniku, gdzie różnice między dobranymi napisami są niewielkie, odległości pozwalające na więcej operacji edycyjnych (tak jak np. odległość Damerau-Levenshteina) mogą dać lepsze rezultaty. Odległości Jaro i Jaro-Winklera zostały skonstruowane do krótkich, napisanych przez człowieka, napisów, więc ich zakres zastosowania powinien być jasny.

Rozdział 3

Analiza skupień metodą k-średnich

Analiza skupień polega na wyróżnieniu w zbiorze ustalonej liczby rozłącznych i niepustych podzbiorów obserwacji w jakimś sensie do siebie podobnych, równocześnie zachowując maksymalne zróżnicowanie obserwacji pomiędzy poszczególnymi podzbiorami [15]. W niniejszym rozdziale przedstawimy analizę skupień metodą k-średnich w trzech odsłonach: metodę wsadową, przy użyciu stochastycznego spadku gradientu oraz metodę pośrednią, tzw. miniwsadową. Dodatkowo zaprezentujemy metody hierarchiczne analizy skupień.

3.1. Metoda k-średnich

[TO DO: DOROBIC PRZYKLADY:
- (2.1, 2.1) POKAZAC SKUPIENIA OTRZYMANE PRZY UZYCIU 3/4 ALGORYTMOW NP. NA IRISIE
- (2.3) NARYSOWAC JAKIS DENDROGRAM
- (2.3) POKAZAC ODMIENNOSCI NAJBLIZSZEGO, NAJDALSZEGO, SREDNIEGO
- (2.4) PRZYKLADY JAKOSCI NA PODSTAWIE JAKICHS LOSOWYCH KLAS
]

Rozważmy przestrzeń euklidesową $\mathbb{R}^p$ i niech będzie dana liczba skupień $k \in \mathbb{N}$. Wówczas zadanie znalezienia skupień o wyżej wymienionych własnościach można sprowadzić do dobrze określonego zadania optymalizacyjnego. Weźmy próbę n-elementową obserwacji $\mathbf{x}_i,\ i = 1, \ldots, n$ o wartościach w $\mathbb{R}^p$. Suma kwadratów odległości euklidesowej między obserwacjami próby wynosi [15]:

$$T = \frac{1}{2} \sum_{i}^{n} \sum_{j}^{n} \|\mathbf{x}_i - \mathbf{x}_j\|_2^2, \tag{3.1}$$

gdzie $\|\cdot\|_2$ oznacza normę euklidesową. Niech funkcja $C : \{1, \ldots, n\} \to \{1, \ldots, k\}$ oznacza przydzielenie danej obserwacji danemu skupieniu, tzn. jeśli $C(i) = l$, to oznacza, że $\mathbf{x}_i$ należy do l-tego skupienia. Zakładając, że dokonano podziału próby na k podzbiorów, można całkowitą sumę kwadratów rozłożyć na sumę kwadratów odległości między obserwacjami z tego samego skupienia oraz na sumę kwadratów odległości między obserwacjami z różnych

skupień [15]:

$$T = W + B = \frac{1}{2}\sum_{l=1}^{k}\sum_{C(i)=k}\sum_{C(j)=k}\|\mathbf{x}_i - \mathbf{x}_j\|_2^2 + \frac{1}{2}\sum_{l=1}^{k}\sum_{C(i)=k}\sum_{C(j)\neq k}\|\mathbf{x}_i - \mathbf{x}_j\|_2^2. \tag{3.2}$$

Mając tak sformułowany rozkład sumy T widzimy, że zmieniając podział punktów na skupienia zmienia się zarówno suma W, jak i B. Można więc sformułować problem analizy skupień jako zadanie minimalizacji sumy W lub, równoważnie, maksymalizacji sumy B. Maksymalizacja B to po prostu maksymalizacja rozproszenia punktów z różnych podzbiorów, co jest równoznaczne z minimalizacją rozproszenia punktów z tego samego skupienia. Stąd, rozwiązaniem problemu analizy skupień jest dokonanie takiego podziału próby, aby zminimalizować sumę W. Niestety, ze względu na złożoność obliczeniową, niemożliwe jest bezpośrednie rozwiązanie tego problemu [15]. Można pokazać, że jest on NP-trudny [3].

Przez n_l oznaczmy liczność l-tego skupienia i niech $\mathbf{m}_l = \frac{1}{n_l}\sum_{C(i)=l}\mathbf{x}_i$ oznacza wektorową średnią obserwacji z l-tego skupienia. Łatwo zauważyć, że [15]:

$$W = \frac{1}{2}\sum_{l=1}^{k}\sum_{C(i)=l} n_l\|\mathbf{x}_i - \mathbf{m}_l\|_2^2. \tag{3.3}$$

Średnie $\mathbf{m}_l$, $l = 1, \ldots, k$ nazywamy *środkami* lub *centroidami skupień*. Równanie (3.3) można uprościć do następującej postaci:

$$\widetilde{W} = \frac{1}{2}\sum_{l=1}^{k}\sum_{C(i)=l}\|\mathbf{x}_i - \mathbf{m}_l\|_2^2 = \frac{1}{2}\sum_{i=1}^{n}\|\mathbf{x}_i - \mathbf{m}_{C(i)}\|_2^2 \ . \tag{3.4}$$

Algorytmy, które rozwiązują problem minimalizacji sumy (3.4), znane są pod nazwą *metody k-średnich*.

3.2. Algorytmy k-średnich

3.2.1. Algorytm wsadowy

Algorytm *wsadowy* (ang. *batch algorithm*) zostaje zainicjalizowany przez losowe wyznaczenie k punktów jako początkowe środki skupień. Dalej następuje przydzielenie wszystkich punktów próby do najbliższego skupienia, a następnie przeliczenie środków jako średniej ze wszystkich obserwacji w danym skupieniu. Procedura ta jest powtarzana aż do ustabilizowania się algorytmu, tj. do momentu aż żaden punkt próby nie zmieni skupienia [9].

Algorytm wsadowy jest najbardziej popularnym i najczęściej stosowanym algorytmem, gdyż jest szybki i zazwyczaj daje dobre rezultaty. Jednakże jeśli liczba obserwacji w zbiorze jest bardzo duża, to obliczanie średnich z obserwacji we wszystkich skupieniach jest dość kosztowne obliczeniowo, zbiegając w czasie $O(knp)$, gdzie p to liczba zmiennych. Stąd Bottou i Bengio [5] zaproponowali algorytm oparty na stochastycznym spadku gradientu.

Algorytm 3.1 Algorytm wsadowy k-średnich

1: dane: liczba skupień k, zbiór danych X
2: losowa inicjalizacja $\mathbf{m}_l,\ \forall l = 1, \ldots, k$
3: **repeat**
4: **for** $i = 1, \ldots, n$ **do**
5: $C(i) = \arg\min_l \|\mathbf{x}_i - \mathbf{m}_l\|_2^2$
6: **end for**
7: **for** $l = 1, \ldots, k$ **do**
8: $\mathbf{m}_l = \frac{1}{n_l} \sum_{C(i)=l} \mathbf{x}_i$
9: **end for**
10: **until** zbieżność

3.2.2. Algorytmy oparte na metodach najszybszego spadku

Algorytmy oparte na metodach najszybszego spadku (zwane też metodami największego spadku oraz algorytmami gradientowymi) są często stosowane np. w regresji liniowej [4]. Idea polega na szukaniu minimum z danej funkcji kosztu, w kolejnych krokach algorytmu aktualizując zmienną, w kierunku, w którym spadek gradientu był największy. Każda aktualizacja zależy od parametru, zwanego *parametrem uczenia*, który musi być odpowiednio dobrany. W niniejszym podrozdziale opiszemy trzy algorytmy gradientowe.

Mając daną funkcję kosztu $\widetilde{W} = \widetilde{W}(\mathbf{m}, \mathbf{x}_i) = \frac{1}{2} \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{m}_{C(i)}\|_2^2$, możemy znaleźć minimum używając tzw. *spadku gradientu*. W każdej iteracji algorytmu uaktualniamy wektor $\mathbf{m}$ na podstawie gradientu $\widetilde{W}(\mathbf{m}, \mathbf{x}_i)$:

$$\mathbf{m}_l^{(t+1)} = \mathbf{m}_l^{(t)} + \gamma \sum_{i=1}^{n} \frac{\partial \widetilde{W}(\mathbf{m}, \mathbf{x}_i)}{\partial \mathbf{m}}, \tag{3.5}$$

gdzie γ jest odpowiednio dobranym *parametrem uczenia*, a t oznacza iterację algorytmu [4]. Zaobserwowano, że parametrem uczenia, które daje najlepsze rezultaty dla algorytmu k-średnich jest $\frac{1}{n_{C(i)}}$ [5]. Stąd też algorytm *wsadowy*, w każdej iteracji algorytmu aktualizuje wektor $\mathbf{m}$ następująco [5]:

$$\mathbf{m}_l^{(t+1)} = \mathbf{m}_l^{(t)} + \sum_{C(i)=l} \frac{1}{n_l} (\mathbf{x}_i - \mathbf{m}_l^{(t)}). \tag{3.6}$$

Algorytm *stochastyczny największego spadku*, ozn. SGD (ang. *stochastic gradient descent*), jest daleko idącym uproszczeniem. Zamiast liczyć gradient z $\widetilde{W}(\mathbf{m}, \mathbf{x}_i)$ wprost, każda iteracja estymuje gradient na podstawie *jednej losowo wybranej* obserwacji $\mathbf{x}_i$ [4]:

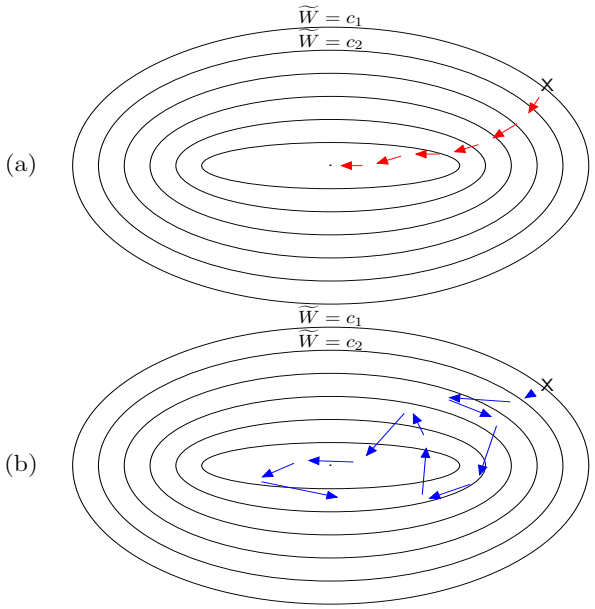
$$\mathbf{m}_l^{(t+1)} = \mathbf{m}_l^{(t)} + \gamma \frac{\partial \widetilde{W}(\mathbf{m}, \mathbf{x}_i)}{\partial \mathbf{m}}, \tag{3.7}$$

gdzie γ jest odpowiednio dobranym parametrem uczenia. Tak samo jak w przypadku algorytmu wsadowego, parametrem uczenia, które daje najlepsze rezultaty jest $\frac{1}{n_{C(i)}}$ [5]. Stąd też algorytm stochastyczny w każdej iteracji algorytmu aktualizuje wektor $\mathbf{m}$ następująco [5]:

$$n_l^{(t+1)} = n_l^{(t)} + \begin{cases} 1, & \text{gdy } l = C(i), \\ 0, & \text{wpp.} \end{cases} \tag{3.8}$$

$$\mathbf{m}_l^{(t+1)} = \mathbf{m}_l^{(t)} + \begin{cases} \frac{1}{n_l}(\mathbf{x}_i - \mathbf{m}_l^{(t)}), & \text{gdy } l = C(i), \\ 0, & \text{wpp.} \end{cases} \tag{3.9}$$

Algorytm SGD opiera się na przeliczaniu średniej po każdym przydzieleniu obserwacji do skupienia, choć z powodu stochastycznego szumu, takie rozwiązanie może nie prowadzić do lokalnego minimum, a jedynie w jego otoczenie. Na Rys. 3.1 przedstawiono przykładową drogę aktualizacji parametrów.



Rysunek 3.1: Przykładowa droga wsadowego spadku gradientu (Rys. a) i stochastycznego spadku gradientu (Rys. b).

Kolejne elipsy oznaczają warstwice $\widetilde{W}(\mathbf{m}, \mathbf{x}_i)$ w zależności od wartości zmiennej $\mathbf{m}$, a centrum oznacza (lokalne) minimum tej funkcji. Jeśli algorytm rozpoczyna działanie w punkcie oznaczonym przez X, to w przypadku algorytmu wsadowego, w kolejnych iteracjach zmienna $\mathbf{m}$ zmienia swoją wartość, przybliżając się do (lokalnego) minimum funkcji $\widetilde{W}$. Natomiast algorytm stochastyczny w każdej iteracji przybliża się w stronę minimum w sposób quasi-losowy, tzn. może nigdy nie osiągnąć właściwego minimum, a jedynie „krążyć" wokół niego.

Algorytm stochastyczny rozpoczyna się tak samo jak algorytm wsadowy, tj. inicjalizacją losowych k środków skupień. Następnie zbiór obserwacji jest mieszany i obserwacje po kolei są przydzielane do najbliższego skupienia. Środki skupień przeliczane są po każdym przydzieleniu punktu do skupienia. Procedura ta powtarzana jest do uzyskania zbieżności [5]. Algorytm ten jest dużo szybszy od dwóch wcześniejszych, kosztem dokładności rozwiązania [4].

Algorytm *mini-wsadowy* (ang. *mini-batch k-means*) jest połączeniem dwóch poprzednich algorytmów, tj. w każdej iteracji przydzielanych do najbliższego skupienia jest b losowo wybranych obserwacji, po czym następuje przeliczenie środków skupień [27]. Algorytm ten jest porównywalnie szybki do algorytmu SGD, osiągając przy tym lepsze rezultaty z powodu mniejszego stochastycznego szumu.

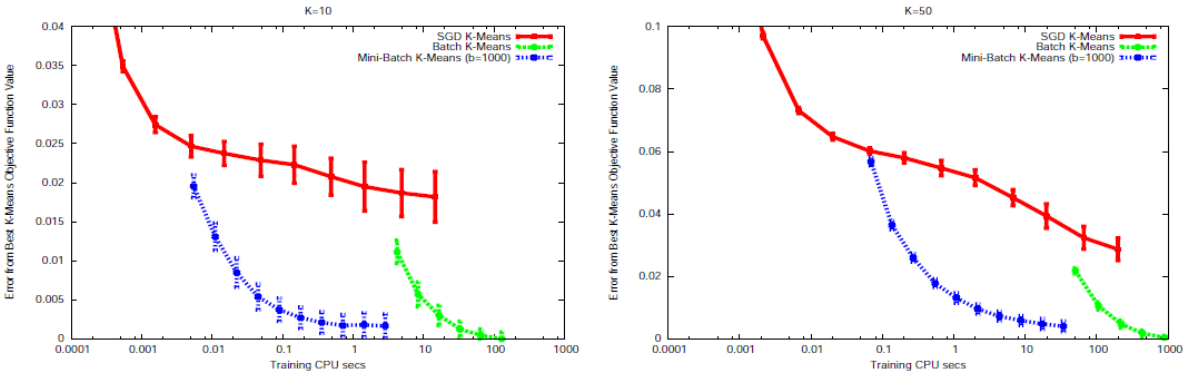
Algorytm 3.2 Algorytm SGD k-średnich

dane: liczba skupień k, zbiór danych X
losowa inicjalizacja $m_l,\ \forall l = 1, ..., k$
inicjalizacja $n_l = 0,\ \forall l = 1, \ldots, k$
repeat
    losowo wybierz jedną obserwację $\mathbf{x}_i$ z X
    $C(i) = \arg\min_l \|\mathbf{x}_i - \mathbf{m}_l\|_2^2$
    $n_{C(i)} = n_{C(i)} + 1$
    $\mathbf{m}_{C(i)} = \mathbf{m}_{C(i)} + \frac{1}{n_{C(i)}} \|\mathbf{x}_i - \mathbf{m}_{C(i)}\|_2$
until zbieżność

Algorytm 3.3 Algorytm mini-wsadowy k-średnich

dane: liczba skupień k, zbiór danych X, parametr b
losowa inicjalizacja $m_l,\ \forall l = 1, ..., k$
inicjalizacja $n_l = 0,\ \forall l = 1, ..., k$
repeat
    $B = b$ obserwacji losowo wybranych z X
    for $i : \mathbf{x}_i \in B$ **do**
        $C(i) = \arg\min_l \|\mathbf{x}_i - \mathbf{m}_l\|_2^2$
    end for
    for $i : \mathbf{x}_i \in B$ **do**
        $n_{C(i)} = n_{C(i)} + 1$
        $\mathbf{m}_{C(i)} = \mathbf{m}_{C(i)} + \frac{1}{n_{C(i)}} \|\mathbf{x}_i - \mathbf{m}_{C(i)}\|_2$
    end for
until zbieżność

Porównanie działania trzech wyżej zaprezentowanych algorytmów dla $k = 10, 50$ prezentuje Rys. 3.2. Na osi rzędnych mamy uzyskany błąd analizy, na osi odciętych natomiast – czas działania algorytmu na skali logarytmicznej. Można zauważyć, że najszybszy był algorytm SGD (czerwona krzywa), jednak błąd oszacowania był największy. Z drugiej strony mamy algorytm wsadowy (zielony), który ma bardzo mały błąd, choć czas działania algorytmu był najdłuższy. Algorytm mini-wsadowy (niebieski) połączeniem dwóch poprzednich. Błąd w tym przypadku jest podobny jak błąd algorytmu wsadowego, natomiast czas działania jest o wiele krótszy, porównywalnie do algorytmu SGD.



Rysunek 3.2: Szybkość a jakość zbieżności algorytmów k-średnich. Czerwona krzywa oznacza metodę SGD, zielona – wsadową, natomiast niebieska – mini-wsadową. *Źródło:* [27]

Zauważmy, że wszystkie znane i stosowane wersje algorytmów k-średnich są zbieżne. W tym celu wystarczy, aby w każdej iteracji algorytmu suma $\widetilde{W}$ była zmniejszona. W przeciwnym przypadku algorytm zostaje zatrzymany. Warto zauważyć jednak, że rozwiązanie takie może nie prowadzić do rozwiązania optymalnego, tj. algorytm może zatrzymać działanie w minimum lokalnym wartości $\widetilde{W}$, zamiast zbiec do minimum globalnego. Stąd zaleca się wielokrotne stosowanie danego algorytmu z różnymi warunkami początkowymi [15].

3.3. Metody hierarchiczne

Metody hierarchiczne to zbiór algorytmów analizy skupień, które nie wymagają znajomości liczby skupień. W niniejszym podrozdziale przedstawimy pokrótce schemat ich działania w dowolnej przestrzeni obserwacji.

Powyżej przedstawiliśmy algorytm k-średnich, dzielący zbiór z przestrzeni euklidesowej na podzbiory punktów podobnych do siebie, tj. mieliśmy do czynienia ze zmiennymi liczbowym. Problem ten można uogólnić dla obserwacji z dowolnej przestrzeni. Po pierwsze odległość euklidesową z równania (3.4) można zamienić na dowolną funkcję odmienności d między obserwacjami z danej przestrzeni. Przez funkcję odmienności rozumiemy następującą funkcję:

Definicja 3.1. Funkcją odmienności *nazywamy funkcję* $d : X \times X \to \mathbb{R}_+ \cup \{0\}$*, która* $\forall x, y \in X$ *spełnia następujące warunki:*

- $d(x, x) = 0$,
- $d(x, y) = d(y, x)$.

Po drugie trzeba zastąpić średnie skupień $\mathbf{m}_l, l = 1, \ldots, k$ inną wartością wektorową, która miałaby sens w przypadku np. atrybutów jakościowych. Wartość ta to punkt ze zbioru obserwacji, który minimalizuje sumę odległości między nim samym, a pozostałymi punktami ze skupienia [15]:

$$\mathbf{m}_l = \min_{\mathbf{y} \in X} \sum_{i=C(l)} d(\mathbf{x}_i, \mathbf{y}), \tag{3.10}$$

gdzie X to zbiór obserwacji.

Przejdźmy do metod hierarchicznych analizy skupień. Jak wspomniano wcześniej, nie wymagają one specyfikowania liczby podzbiorów. Zamiast tego trzeba zdefiniować miarę odmienności (rozłącznych) zbiorów obserwacji, opartą na odmienności pojedynczych punktów w tych zbiorach. Jak sugeruje nazwa, metody te konstruują hierarchiczną reprezentację, w której skupienie na każdym poziomie hierarchii powstaje przez połączenie skupień z najbliższego niższego poziomu. Na najniższym poziomie znajdują się skupienia złożone z pojedynczych obserwacji. Najwyższy poziom to skupienie zawierające wszystkie obserwacje ze zbioru [12].

Metody hierarchiczne można podzielić na dwie grupy: aglomeracyjne oraz dzielące. W tej pierwszej, na początku tworzy się tyle skupień ile jest obserwacji w zbiorze, traktując każdą obserwację jako osobne skupienie. Następnie w każdym kroku łączona jest ta para podzbiorów, które są od siebie najmniej odmienne. W ten sposób na kolejnym poziomie otrzymujemy (przynajmniej) o jedno skupienie mniej. Procedura łącząca skupienia najmniej odmienne trwa

nadal, w każdym kolejnym kroku zmniejszając liczbę skupień. W ostatnim kroku algorytmu otrzymujemy jedno duże skupienie zawierające wszystkie obserwacje z próby [12, 15].

Metody dzielące działają w przeciwnym kierunku: zaczynamy od jednego skupienia zawierającego cały zbiór. Następnie w każdym kroku algorytmu jedno ze skupień dzielone jest na dwa skupienia, w których odmienność jest największa. W ten sposób na kolejnym poziomie otrzymujemy o jedno skupienie więcej. Procedura dzieląca skupienia najbardziej odmienne trwa nadal, w każdym kolejnym kroku otrzymując coraz więcej skupień. W ostatnim kroku algorytmu dostajemy podzbiory jednoelementowe, dostając n rozłącznych skupień (gdzie n to liczba obserwacji). Warto przy tym zauważyć, że algorytm ten jest znacznie bardziej złożony obliczeniowo niż algorytm aglomeracyjny [12, 15].

Zastanówmy się teraz w jaki sposób mierzyć odmienność między podzbiorami. Ponieważ metody dzielące są o wiele bardziej złożone obliczeniowo, skupimy się na metodach aglomeracyjnych, jako tych częściej stosowanych w praktyce. Odmienność między skupieniami można definiować na różne sposoby, jednak literatura podaje zazwyczaj najbardziej popularne, tj. kryterium Warda, odmienność centroidów, mediany, najbliższego sąsiada, najdalszego sąsiada oraz średnią odmienność [12, 15].

Metoda Warda jest kryterium stosowanym w hierarchicznej analizie skupień. Ward zaproponował ogólną procedurę metod aglomeracyjnych, gdzie o wyborze połączenia pary skupień decyduje się na podstawie wartości pewnej funkcji celu. Tą funkcją celu może być każda funkcja, która odzwierciedla cel badacza. Wiele standardowych procedur tworzenia skupień jest zawartych w tym bardzo ogólnym podejściu. Aby zilustrować procedurę, Ward użył przykładu, gdy funkcja celu jest sumą kwadratów błędów, a przykład ten jest znany jako *metoda Warda* lub dokładniej jako *metoda minimalnej wariancji Warda* [14].

Metoda ta może być zdefiniowana i zaimplementowana przy pomocy rekurencyjnych wzorów Lance'a-Williamsa [17]. Wzory te to nieskończona rodzina algorytmów grupowania aglomeracyjnego, które są reprezentowane przez rekurencyjny wzór aktualizujący odległości skupień na każdym poziomie. W każdym kroku niezbędne jest, aby zoptymalizować funkcję celu. Rekurencyjna formuła ułatwia więc znalezienie odpowiedniej pary skupień.

Oznaczmy przez D_{ij} odmienność między skupieniem i-tym a j-tym, których liczności wynoszą odpowiednio n_i, n_j, a przez $D_{(ij)l}$ odległość między połączonym skupieniem i-tym i j-tym a skupieniem l-tym.

Definicja 3.2. *Wzory Lance'a-Williamsa definiujemy następująco [17]:*

$$D_{(ij)l} = \alpha_i D_{il} + \alpha_j D_{jl} + \beta D_{ij} + \gamma |D_{il} - D_{jl}|,$$

gdzie $\alpha_i, \alpha, \beta, \gamma$ są parametrami zależnymi od liczności skupień.

Metoda minimalnej wariancji Warda wyrażona w terminach wzorów Lance'a-Williamsa przyjmuje następującą postać:

$$D_{(ij)l} = \frac{n_i + n_l}{n_i + n_j + n_l} D_{il} + \frac{n_j + n_l}{n_i + n_j + n_l} D_{jl} - \frac{n_l}{n_i + n_j + n_l} D_{ij},$$

gdzie n_l to liczność l-tego skupienia. Wartości parametrów z algorytmu Lance'a-Williamsa w powyższym wzorze wynoszą:

$$\alpha_i = \frac{n_i + n_l}{n_i + n_j + n_l}, \quad \beta = \frac{-n_l}{n_i + n_j + n_l}, \quad \gamma = 0.$$

Definicja 3.3. Odmienność centroidów *(ang.* centroid linkage dissimilarity*) między skupieniem i-tym a j-tym, definiujemy jako odległość między środkami skupień:*

$$D_{ij} = d(\mathbf{m}_i, \mathbf{m}_j),$$

gdzie $\mathbf{m}_i, \mathbf{m}_j$ *oznaczają środki, odpowiednio, skupienia i-tego i j-tego.*

W terminach wzorów Lance'a-Williamsa można to wyrazić jako $D_{(ij)l} = \frac{n_i}{n_i+n_j}D_{il} + \frac{n_j}{n_i+n_j}D_{jl} - \frac{n_i \cdot n_j}{(n_i+n_j)^2}D_{ij}$, tj. $\alpha_i = \frac{n_i}{n_i+n_j}, \beta = -\frac{n_i \cdot n_j}{(n_i+n_j)^2}$, a $\gamma = 0$.

Definicja 3.4. Odmienność median *(ang.* median linkage dissimilarity*) między skupieniem i-tym a j-tym, definiujemy jako odległość między medianami obserwacji ze skupień:*

$$D_{ij} = d(\mathbf{med}_i, \mathbf{med}_j),$$

gdzie $\mathbf{med}_i, \mathbf{med}_j$ *oznaczają medianę obserwacji, odpowiednio, skupienia i-tego i j-tego.*

Zauważmy, że odmienność median ma sens tylko w przypadku, gdy d jest kwadratem odległości euklidesowej, gdyż w innym przypadku mediana z obserwacji nie ma sensu.

W terminach wzorów Lance'a-Williamsa można to wyrazić jako $D_{(ij)l} = 0,5D_{il} + 0,5D_{jl} - 0,25D_{ij}$, tj. $\alpha_i = 0,5, \beta = -0,25$, a $\gamma = 0$.

Definicja 3.5. Odmienność najbliższego sąsiada *(ang.* single linkage dissimilarity*) między skupieniem i-tym a j-tym, definiujemy jako najmniejszą spośród wszystkich możliwych odmienności między parami obserwacji z i-tego i j-tego skupienia:*

$$D_{ij} = \min_{\mathbf{x}_a \in C(i), \mathbf{x}_b \in C(j)} d(\mathbf{x}_a, \mathbf{x}_b).$$

W terminach wzorów Lance'a-Williamsa można to wyrazić jako $D_{(ij)l} = 0,5D_{il} + 0,5D_{jl} - 0,5|D_{il} - D_{jl}|$, tj. $\alpha_i = 0,5, \gamma = -0,5$, a $\beta = 0$.

Definicja 3.6. Odmienność najdalszego sąsiada *(ang.* complete linkage dissimilarity*) między skupieniem i-tym a j-tym, definiujemy jako największą spośród wszystkich możliwych odmienności między parami obserwacji z i-tego i j-tego skupienia:*

$$D_{ij} = \max_{\mathbf{x}_a \in C(i), \mathbf{x}_b \in C(j)} d(\mathbf{x}_a, \mathbf{x}_b).$$

W terminach wzorów Lance'a-Williamsa można to wyrazić jako $D_{(ij)l} = 0,5D_{il} + 0,5D_{jl} + 0,5|D_{il} - D_{jl}|$, tj. $\alpha_i = \gamma = 0,5$, a $\beta = 0$.

Definicja 3.7. Odmienność średnią *(ang.* average linkage dissimilarity*) między skupieniem i-tym a j-tym, definiujemy jako średnią odmienności między parami obserwacji z i-tego i j-tego skupienia:*

$$D_{ij} = \frac{1}{n_i n_j} \sum_{\mathbf{x}_a \in C(i), \mathbf{x}_b \in C(j)} d(\mathbf{x}_a, \mathbf{x}_b).$$

W terminach wzorów Lance'a-Williamsa można to wyrazić jako $D_{(ij)l} = \frac{n_i}{n_i+n_j}D_{il} + \frac{n_j}{n_i+n_j}D_{jl}$, tj. $\alpha_i = \frac{n_i}{n_i+n_j}$, a $\beta = \gamma = 0$.

Odmienność najbliższego sąsiada wymaga, żeby jedna odmienność $d(\mathbf{x}_a, \mathbf{x}_b)$, gdzie $\mathbf{x}_a \in C(i)$, $\mathbf{x}_b \in C(j)$, miała małą wartość, aby uznać dwa podzbiory za bliskie sobie, bez względu na odmienności innych obserwacji z tych dwóch grup. Metoda ta będzie zatem mieć skłonność do łączenia skupień połączonych przez szereg bliskich obserwacji pośrednich. Takie zjawisko nazywane jest *efektem łańcuchowym* i uważane jest za wadę tej metody. Skupienia otrzymane w wyniku jej działania często nie są zwarte, jako że podobieństwo skupień określone jest na podstawie dwóch najbliższych obserwacji [12]. Odmienność najbliższego sąsiada daje zazwyczaj skupienia wąskie i wydłużone [15].

Metoda odmienności najdalszego sąsiada ma działanie przeciwne. Dwa skupienia są do siebie podobne, wtedy i tylko wtedy, gdy wszystkie obserwacje z ich połączenia są dość podobne. Jednakże, metoda ta może nie zachowywać własności „bliskości" dwóch podzbiorów, tj. obserwacje przypisane do danego skupienia mogą znajdować się o wiele bliżej obserwacji z innego podzbioru, niż do punktów ze swojego skupienia [12]. W wyniku jej działania otrzymujemy podzbiory o kulistym kształcie [15].

Metoda średniej odmienności jest kompromisem pomiędzy dwoma powyższymi. Próbuje ona dać skupienia relatywnie zwarte i względnie oddalone od siebie [12]. Podobnie jak metoda odmienności najdalszego sąsiada, daje ona skupienia o kształcie kulistym [15].

Jeśli odmienności poszczególnych obserwacji mają silną tendencję do skupiania się, przy czym każde skupienie jest zwarte i dobrze odseparowane, to wszystkie trzy metody dadzą podobne rezultaty [12].

3.4. Metody oceny jakości podziału na skupienia

Dokonawszy podziału zbioru na skupienia, należy ocenić jakość zastosowanego algorytmu. Ocena skuteczności działania algorytmów analizy skupień nie należy do zadań tak prostych jak np. ocena modelu klasyfikacji pod nadzorem. W szczególności żadna metoda ocena nie powinna brać pod uwagę wartości etykiety skupienia, ale powinna sprawdzać, czy zbiór danych jest dobrze podzielony, tzn. czy obserwacje w poszczególnych skupieniach są do siebie „podobne", a obserwacje z różnych skupień – „niepodobne", zgodnie z przyjętą metryką podobieństwa. W niniejszych podrozdziale przedstawimy kilka miar oceny jakości podziału na skupienia.

Przyjmujemy następującego założenia: Niech K i C oznaczają dwa różne podziały n-elementowego zbioru X na skupienia. Zazwyczaj K oznacza podział uzyskany przy pomocy algorytmu dzielącego zbiór, a C jest zbiorem prawdziwych klas, do których należą obserwacje, choć K i C mogą również oznaczać dwa niezależne podziały uzyskane przy pomocy różnych algorytmów.

3.4.1. Indeks Fowlkesa-Mallowsa

Definicja 3.8. *Niech macierz $M = [m_{ij}]$, $i, j = 1, \ldots, k$ oznacza liczbę elementów z X, które należą do i-tego skupienia w K i j-tego skupienia w C. Możemy wówczas zdefiniować* indeks

Fowlkesa-Mallowsa, *ozn. indeks* FM *[10]:*

$$\mathrm{FM} = \frac{T}{\sqrt{P \cdot Q}}, \tag{3.11}$$

gdzie

$$T = \sum_{i=1}^{k}\sum_{j=1}^{k} m_{ij}^2 - n,$$
$$P = \sum_{i=1}^{k}(\sum_{j=1}^{k} m_{ij})^2 - n,$$
$$Q = \sum_{j=1}^{k}(\sum_{i=1}^{k} m_{ij})^2 - n.$$

Wartości indeksu FM znajdują się w przedziale $[0, 1]$..

Przykładowo, jeśli mamy dany sześcioelementowy zbiór, który dzielimy na trzy skupienia i $K = [1, 1, 2, 2, 3, 3]$, a $C = [2, 1, 3, 1, 2, 2]$, to

$$M = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 2 & 0 \end{bmatrix},$$

$T = 2$, $Q = 6$, a $P = 8$. Stąd $\mathrm{FM} = \frac{2}{\sqrt{6 \cdot 8}} \approx 0,29$.

Indeks FM przyjmuje wartości z przedziału $[0, 1]$, gdzie zero oznacza zupełnie różne podziały, natomiast jeden – pełną zgodność. Ponieważ indeks FM jest wprost proporcjonalny do sumy liczby wspólnych elementów w obu podziałach, większa wartość indeksu oznacza większe podobieństwo między uzyskanymi podziałami. Ponadto Fowlkes i Mallows [10] pokazali, że porównując dwa niezależne podziały, to indeks osiąga wartości bliskie zeru, podczas gdy inne wskaźniki, np. indeks Randa [23] często szybko dążą w takim przypadku do 1. Stąd indeks FM jest o wiele bardziej dokładny dla niezależnych podziałów. FM daje również dobre rezultaty w przypadku, gdy do danych zostanie dodany szum – czym większy szum w danych, tym indeks przyjmuje mniejsze wartości. Czyni go tym samym solidnym narzędziem mierzącym jakość uzyskanego podziału. Jednak w przypadku sprawdzenia jakości działania jednego algorytmu, wymagana jest znajomość prawdziwego podziału zbioru, co w praktyce rzadko występuje lub wymaga ręcznego podziału zbioru.

3.4.2. Jednorodność, zupełność oraz miara V

Niech C oznacza zbiór prawdziwych klas, do których należą obserwacje. Mówimy, że podział zbioru jest *jednorodny*, jeśli wszystkie skupienia zawierają jedynie obserwacje z jednej klasy. Podział zbioru jest *zupełny*, jeśli wszystkie obserwacje z danej klasy są w tym samym skupieniu. Jednorodność i zupełność podziału mogą być często w opozycji do siebie, tzn. gdy jednorodność rośnie, to zupełność zazwyczaj maleje i odwrotnie. Przykładowo, rozważmy dwa skrajne podziały. W przypadku pierwszego, gdy przydzielamy wszystkie obserwacje do jednego skupienia, to dostajemy idealną zupełność – wszystkie elementy z jednej klasy należą do tego samego skupienia. Jednakże, podział taki jest tak *nie*jednorodny, jak to tylko

możliwe, skoro wszystkie klasy znajdują się w jednym skupieniu. Z drugiej strony, rozważmy przydzielenie każdej obserwacji do osobnego skupienia. W tym przypadku, podział jest idealnie jednorodny – każde skupienie zawiera jedynie obserwacje z jednej klasy. Jednak w terminach zupełności, taki podział bardzo słaby, chyba że rzeczywiście każda klasa zawiera jeden element. Miara V jest ważoną średnią harmoniczną dwóch powyższych miar [24].

Jednorodność. Żeby spełnić kryteria, jak musi spełniać podział jednorodny, każde ze skupień musi zawierać tylko i wyłącznie te obserwacje, które należą do jednej klasy. To znaczy, że rozkład klas w każdym ze skupień powinien skośny i zawierać tylko jedną klasę, tj. entropia powinna wynosić zero. Aby ustalić jak blisko idealnego znajduje się dany podział, sprawdzamy warunkową entropię rozkładu klas pod warunkiem zaproponowanego podziału. W idealnie jednorodnym podziale, ta wartość, tj. $H(C|K)$, wynosi zero. Jednak w przeciwnym przypadku, wartość ta zależy od wielkości zbioru i rozkładu klas. Stąd, zamiast badać warunkową entropię, normalizujemy tę wartość przez maksymalną redukcję entropii jaką informacja o podziale może przynieść, tj. $H(C)$ [24].

Zauważmy, że $H(C|K)$ jest maksymalne (i równe $H(C)$), kiedy podział na skupienia nie wnosi żadnej nowej informacji – rozkład klas w każdym skupieniu jest równy ogólnemu rozkładowi klas. $H(C|K)$ jest równy zeru, gdy każde skupienie zawiera jedynie obserwacje z jednej klasy, tj. w przypadku idealnie jednorodnego podziału. W przypadku zdegenerowanym, gdy $H(C) = 0$, kiedy istnieje tylko jedna klasa, definiujemy jednorodność jako równą 1. W idealnie jednorodnym rozwiązaniu, taka normalizacja, tj. $\frac{H(C|K)}{H(C)}$, wynosi zero. Stąd, aby utrzymać konwencję, że wartość 1 jest pożądana, a wartość 0 jest niepożądana, definiujemy jednorodność jako [24]:

Definicja 3.9. *Przez* jednorodność *rozumiemy:*

$$h = 1 - \frac{H(C|K)}{H(C)} \tag{3.12}$$

gdzie:

$$H(C) = -\sum_{c=1}^{|C|} \frac{n_c}{n} \cdot \log \frac{n_c}{n},$$

$$H(C|K) = -\sum_{c=1}^{|C|} \sum_{k=1}^{|K|} \frac{n_{c,k}}{n} \cdot \log \frac{n_{c,k}}{n_k},$$

gdzie n_c oznacza liczność klasy c, $c = \{1, \ldots, |C|\}$, a $n_{c,k}$ oznacza liczbę elementów, która należy do klasy c i skupienia k, $k = \{1, \ldots, |K|\}$.

Przykładowo, jeśli mamy dany sześcioelementowy zbiór, który dzielimy na trzy skupienia i $K = [1, 1, 2, 2, 3, 3]$, a $C = [2, 1, 3, 1, 2, 2]$, to $h \approx 0, 54$.

Zgodność. Miara zgodności jest symetryczna do miary jednorodności. Aby spełnić warunki zgodności, podział zbioru musi przydzielić wszystkie obserwacje z jednej klasy, do tego samego skupienia. Żeby policzyć zgodność, badamy rozkład przypisanych skupień w obrębie jednej klasy. W idealnie zgodnym podziale, każdy z rozkładów będzie skośny i będzie zawierać jedynie tylko jedno skupienie. Możemy oszacować poziom skośności, licząc warunkową entropię zaproponowanego podziału pod warunkiem klas, tj. $H(K|C)$. W idealnie zgodnym rozwiązaniu $H(K|C) = 0$. Jednak w najgorszym możliwym przypadku, gdy wszystkie obserwacje z tej samej klasy są we wszystkich skupieniach, rozkład tych ostatnich jest równy

rozkładowi rozmiarów klastrów, $H(K|C)$ jest maksymalne i równe $H(K)$. W końcu, w zdegenerowanym przypadku, kiedy $H(K) = 0$, kiedy mamy tylko jedno skupienie, definiujemy zgodność jako równą 1. Stąd, robiąc symetryczne wyliczenie do poprzedniego, definiujemy zgodność jako [24]:

Definicja 3.10. *Przez* zgodność *rozumiemy:*

$$c = 1 - \frac{H(K|C)}{H(K)}, \tag{3.13}$$

gdzie:

$$H(K) = -\sum_{k=1}^{|K|} \frac{n_k}{n} \cdot \log \frac{n_k}{n},$$
$$H(K|C) = -\sum_{c=1}^{|C|} \sum_{k=1}^{|K|} \frac{n_{c,k}}{n} \cdot \log \frac{n_{c,k}}{n_c},$$

gdzie n_c oznacza liczność klasy c, $c = \{1, \ldots, |C|\}$, a $n_{c,k}$ oznacza liczbę elementów, która należy do klasy c i skupienia k, $k = \{1, \ldots, |K|\}$.

Przykładowo, jeśli mamy dany sześcioelementowy zbiór, który dzielimy na trzy skupienia i $K = [1, 1, 2, 2, 3, 3]$, a $C = [2, 1, 3, 1, 2, 2]$, to $h = 0, 5$.

Miara V. Mając zdefiniowane miary jednorodności i zgodności, możemy wyliczyć ich średnią harmoniczną [24]:

Definicja 3.11. *Niech h oznacza jednorodność, a c zgodność. Przez* miarę V *rozumiemy:*

$$V_\beta = \frac{(1 + \beta) \cdot h \cdot c}{(\beta \cdot h) + c}, \tag{3.14}$$

gdzie $\beta \in (0, 1)$.

Zauważmy, że jeśli β jest mniejsza niż 1, jednorodność ma większą wagę niż zgodność. Często za β przyjmuje się po prostu 1, dając równą wagę obu miarom.

Przykładowo, jeśli mamy dany sześcioelementowy zbiór, który dzielimy na trzy skupienia i $K = [1, 1, 2, 2, 3, 3]$, a $C = [2, 1, 3, 1, 2, 2]$, to dla $\beta = 1$ mamy V $\approx 0, 52$.

Warto zwrócić uwagę na fakt, że jednorodność, zgodność oraz miara V są niezależne od liczby klas, skupień, liczby obserwacji oraz użytego algorytmu. Stąd miary te mogą być używane do porównania każdego algorytmu dzielącego na skupienia, niezależnie od powyższych parametrów. Co więcej, wyliczając zarówno jednorodność, jak i zgodność, może zostać otrzymana bardziej precyzyjna ocena jakości podziału skupień.

Wszystkie trzy wyżej zaprezentowane miary dają wartości z przedziału $[0, 1]$, gdzie 0 oznacza najgorszy możliwy przypadek, natomiast 1 to bardzo dobre rozwiązanie. Mają one intuicyjną interpretację: podział z niską wartością miary V może zostać oceniony w terminach jednorodności i zgodności, aby mieć lepsze pojęcie o błędach jakich dokonał algorytm. Dodatkowo, aby jednorodność, zgodność oraz miara V nie mają założeń o strukturze skupienia: przy ich pomocy można porównywać podział na skupienia uzyskany przy użyciu różnych algorytmów, które mają różne założenia o strukturze skupienia [1].

Z drugiej jest strony powyższe miary nie są znormalizowane pod względem losowego przypisania do skupienia. Oznacza to, że w zależności od liczby obserwacji, podziału na skupienia i klasy, losowy przydział do skupień nie zawsze da takie same wartości jednorodności, zgodności oraz miary V. W szczególności, losowe przyporządkowanie może nie dać wartości powyższych miar równych zero, zwłaszcza gdy liczba skupień jest duża. Problem ten może być bezpiecznie zignorowany, gdy liczba obserwacji jest większa od tysiąca, a liczba skupień mniejsza niż 10. Dla mniejszej próbki i większej liczby skupień, bezpieczniej jest używać miary FM. Dodatkowo, w przypadku sprawdzenia jakości działania jednego algorytmu, wymagana jest znajomość prawdziwego podziału zbioru, co w praktyce rzadko występuje lub wymaga ręcznego podziału zbioru [1].

3.4.3. Miara silhouettes

Sylwetki obserwacji (czy też *silhouettes*) są użyteczne, gdy odległości są określone na skali przedziałowej oraz gdy pożądane są wyraźnie odseparowane skupienia. Co więcej nie wymagana jest znajomość prawdziwych klas, do których należą obserwacje, a mówi jedynie o jakości uzyskanego podziału.

Aby skonstruować sylwetkę obserwacji potrzebne są dwa elementy: podział zbioru na skupienia C oraz miarę odległości d pomiędzy obserwacjami [25].

Definicja 3.12. *Weźmy każdą obserwację z próbki $\mathbf{x}_i$ i oznaczmy przez $C(i)$ skupienie, do którego należy. Średnią odmiennością $\mathbf{x}_i$ od swojego skupienia nazywamy [25]:*

$$a(\mathbf{x}_i) = \frac{\sum_{u \in C(i)} d(\mathbf{x}_i, u)}{n_{C(i)}}.$$

Średnią odmiennością $\mathbf{x}_i$ od skupienia J nazywamy:

$$c(\mathbf{x}_i, J) = \frac{\sum_{u \in J} d(\mathbf{x}_i, u)}{n_J}.$$

Wówczas możemy wyliczyć odległość obserwacji $\mathbf{x}_i$ od najbliższego skupienia innego niż $C(i)$:

$$b(\mathbf{x}_i) = \min_{j \neq i} c(\mathbf{x}_i, J).$$

Skupienie L, dla którego minimum jest osiągnięte (tj. $b(\mathbf{x}_i) = c(\mathbf{x}_i, L)$) nazywamy sąsiadem obserwacji $\mathbf{x}_i$. Jest ono drugim najlepszym wyborem dla tej obserwacji. Stąd znajomość sąsiada jest bardzo użyteczna, gdyż mówi o tym które skupienie zostałoby wybrane, gdyby nie zostało nim skupienie $C(i)$.

Definicja 3.13. *Sylwetka obserwacji jest definiowana następująco [25]:*

$$s(\mathbf{x}_i) = \frac{b(\mathbf{x}_i) - a(\mathbf{x}_i)}{\max b(\mathbf{x}_i), a(\mathbf{x}_i)}.$$

Sylwetki otrzymujemy dla każdej obserwacji z osobna, ale najczęściej interesująca jest miara zdefiniowana dla całego zbioru, stąd sylwetką zbioru nazywamy średnią z sylwetek po

wszystkich obserwacjach:

$$\text{sil} = \sum_{i=1}^{n} s(\mathbf{x}_i).$$

Miara ta daje wartości z przedziału $[-1, 1]$, gdzie -1 oznacza niepoprawny podział, natomiast 1 to bardzo gęste skupienia. Wartości w okolicy zera sugerują nakładające się skupienia. Wartość sylwetki jest wyższa, gdy skupienia są gęste i dobrze odseparowane, co jest jedną z najbardziej pożądanych cech analizy skupień. Miara nie wymaga znajomości prawdziwych klas, na jakie można podzielić zbiór.

Rozdział 4

Kategoryzacja tematyczna tekstów przy użyciu metryk w przestrzeni ciągów znaków

Wikipedia[1] jest to wielojęzyczna encyklopedia internetowa, która działa w oparciu o zasadę otwartej treści. Portal umożliwia każdemu z użytkowników odwiedzających stronę, edycję i aktualizację treści w czasie rzeczywistym. Wikipedia ma ponad $35,9$ miliona artykułów we wszystkich wersjach językowych, w tym prawie 5 milionów w wersji angielskiej i nieco ponad $1,1$ miliona artykułów w języku polskim (dane na grudzień 2015) [2].

Każdy artykuł może być stworzony lub zmodyfikowany przez dowolnego użytkownika. Przy procesie edycji oraz pisania artykułu obowiązują liczne reguły, między innymi takie jak wskazanie źródeł bibliograficznych, podlinkowanie do innych artykułów oraz nadanie artykułowi kategorii tematycznych [2].

Ta ostatnia zasada może w szczególności przysporzyć nieco kłopotów. Liczba dostępnych kategorii jest bardzo duża (ponad 125 tys. w polskiej wersji językowej), co więcej poziom ich szczegółowości jest zróżnicowany, tzn. mamy kategorie bardzo ogólne (np. *Matematyka*), jak i dość szczegółowe (np. *Działania dwuargumentowe*). Można się spodziewać, że automatyczny podział tekstów na kategorie na podstawie słów, jakie w nich występują, mógłby dać lepszy, bardziej dopasowany do treści, temat. Sprawdzenie, jak automatyczny podział tekstów na kategorie tematyczne, przy użyciu występujących w nich słów oraz ich liczności, jest zasadniczym celem praktycznej części tej pracy. Idea działania jest następująca: algorytm analizuje jakie słowa w jakich licznościach występują w danym artykule i następnie przydziela go do grupy artykułów, które zawierają takie same i podobne słowa w zbliżonych licznościach. To podejście opiera się na założeniu, że artykuły o podobnej tematyce będą zawierały takie same lub podobne do siebie słowa, tzn. że rozkład słów charakteryzuje temat tekstu.

Schemat działania proponowanego przez nas algorytmu jest następujący:

1. Wstępne przetwarzanie danych.
2. Utworzenie skupień „podobnych" słów przy użyciu *stemmingu* oraz algorytmu hierarchicznego.

[1] `www.wikipedia.org`

3. Stworzenie macierzy o wymiarach liczba artykułów × liczba słów, gdzie wartością jest liczność występowania danego słowa w artykule.

4. Użycie algorytmu k-średnich do podzielenia tekstów na skupienia.

Skuteczność algorytmu automatycznej kategoryzacji tematycznej testowana będzie na artykułach polskiej wersji Wikipedii. Zauważmy, że użycie takiego zbioru danych wiąże się z kilkoma istotnymi wyzwaniami. Przede wszystkim wybór tekstów w języku polskim powoduje, że analiza może być trudna. Jest to język bardzo złożony gramatycznie. Wynika to z faktu, że zawiera on bardzo dużo odmian, m.in. przez przypadki, liczby, osoby, tryby, czasy czy strony. Stąd określenie formy podstawowej czyli mianownika w przypadku rzeczowników czy bezokolicznika często nie jest łatwe, a czasem więc niemożliwe bez znajomości szerszego kontekstu. Weźmy pod uwagę słowo `piła`. Może ono oznaczać narzędzie, jak i czas przeszły trzeciej osoby liczby pojedynczej słowa `pić`. Z powodu bardzo dużej możliwości odmian słów ich liczba jest znacząco większa niż np. w języku angielskim.

Co więcej bardzo wiele wyrazów ma wiele znaczeń. Wspomniana już `piła` może wskazywać narzędzie, ale też miasto, film lub gatunek ryby. Podobnie ze słowem `zamek`, które może się odnosić zarówno do budynku, jak też do mechanizmu zamykającego drzwi oraz zamka błyskawicznego. Widać więc, że analiza języka polskiego może być trudna.

Ponadto zbiór polskiej Wikipedii jest względnie duży. Składa się na niego ponad milion artykułów i powyżej dwóch milionów unikalnych słów. Przetwarzanie tak pokaźnej ilości danych w warunkach pojedynczego, prywatnego komputera jest dużym wyzwaniem. Wiąże się to z koniecznością odpowiedniego zarządzania danymi, tj. zbudowaniem adekwatnej bazy danych oraz koniecznością optymalizacji kodów pod względem zużycia zasobów obliczeniowych oraz pamięci RAM.

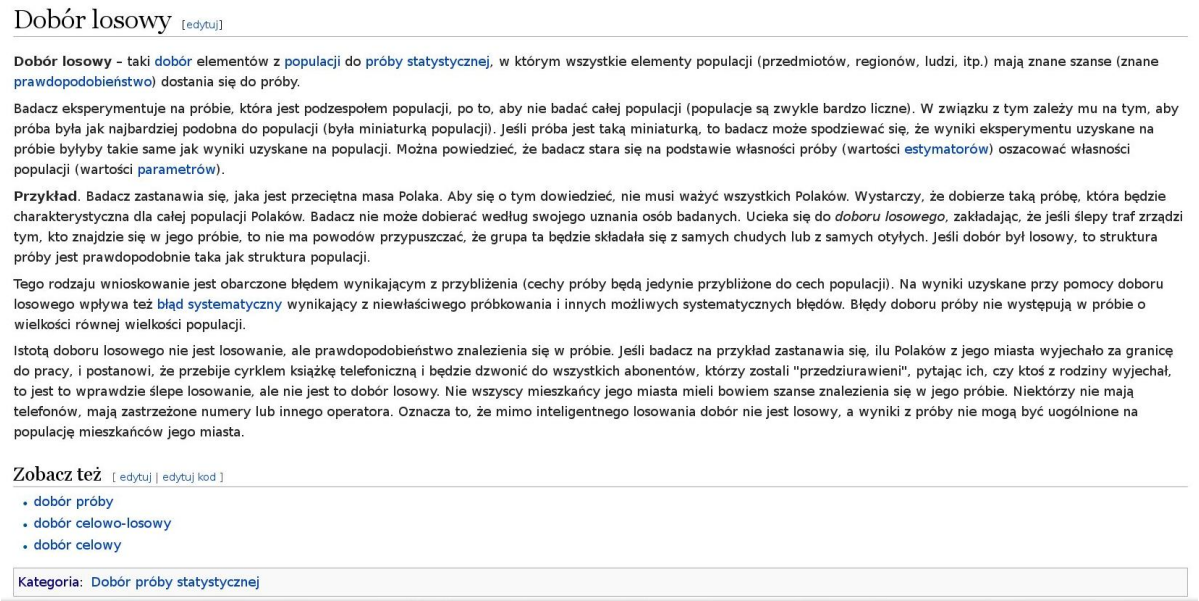
Dalej niektóre metody analizy danych nie dają się efektywnie stosować na dużych zbiorach danych. Istnieje zatem potrzeba zastosowania algorytmów, które dają sensowne wyniki dla zbiorów dużych.

4.1. Opis danych

Omówimy teraz rzeczony zbiór danych.

Zgromadzone dane to zbiór 1 075 568 artykułów z polskiej Wikipedii z dnia 2. listopada 2014 roku w postaci plików XML[2]. Teksty składają się z treści sformułowanych w języku naturalnym, wzorów, kodów, linków wewnętrznych Wikipedii, linków do źródeł zewnętrznych, odniesień do źródeł bibliograficznych, cytatów, przypisów, rysunków (zdjęć, wykresów) wraz z podpisami, tabel, komentarzy, uwag, spisu treści, sekcji „Zobacz też”, kategorii oraz znaczników typowych dla plików języka HTML. Każdy z wyżej wymienionych elementów jest wyróżniony w tekście w inny sposób (np. linki wewnętrzne Wikipedii zawsze znajdowały się wewnątrz podwójnych nawiasów kwadratowych), co nie pozostaje bez znaczenia przy wstępnym przetwarzaniu danych.

[2]Źródło: `http://dumps.wikimedia.org/plwiki/20141102/`



Dobór losowy [edytuj]

Dobór losowy – taki dobór elementów z populacji do próby statystycznej, w którym wszystkie elementy populacji (przedmiotów, regionów, ludzi, itp.) mają znane szanse (znane prawdopodobieństwo) dostania się do próby.

Badacz eksperymentuje na próbie, która jest podzespołem populacji, po to, aby nie badać całej populacji (populacje są zwykle bardzo liczne). W związku z tym zależy mu na tym, aby próba była jak najbardziej podobna do populacji (była miniaturką populacji). Jeśli próba jest taką miniaturką, to badacz może spodziewać się, że wyniki eksperymentu uzyskane na próbie byłyby takie same jak wyniki uzyskane na populacji. Można powiedzieć, że badacz stara się na podstawie własności próby (wartości estymatorów) oszacować własności populacji (wartości parametrów).

Przykład. Badacz zastanawia się, jaka jest przeciętna masa Polaka. Aby się o tym dowiedzieć, nie musi ważyć wszystkich Polaków. Wystarczy, że dobierze taką próbę, która będzie charakterystyczna dla całej populacji Polaków. Badacz nie może dobierać według swojego uznania osób badanych. Ucieka się do *doboru losowego*, zakładając, że jeśli ślepy traf zrządzi tym, kto znajdzie się w jego próbie, to nie ma powodów przypuszczać, że grupa ta będzie składała się z samych chudych lub z samych otyłych. Jeśli dobór był losowy, to struktura próby jest prawdopodobnie taka jak struktura populacji.

Tego rodzaju wnioskowanie jest obarczone błędem wynikającym z przybliżenia (cechy próby będą jedynie przybliżone do cech populacji). Na wyniki uzyskane przy pomocy doboru losowego wpływa też błąd systematyczny wynikający z niewłaściwego próbkowania i innych możliwych systematycznych błędów. Błędy doboru próby nie występują w próbie o wielkości równej wielkości populacji.

Istotą doboru losowego nie jest losowanie, ale prawdopodobieństwo znalezienia się w próbie. Jeśli badacz na przykład zastanawia się, ilu Polaków z jego miasta wyjechało za granicę do pracy, i postanowi, że przebije cyrklem książkę telefoniczną i będzie dzwonić do wszystkich abonentów, którzy zostali "przedziurawieni", pytając ich, czy ktoś z rodziny wyjechał, to jest to wprawdzie ślepe losowanie, ale nie jest to dobór losowy. Nie wszyscy mieszkańcy jego miasta mieli bowiem szanse znalezienia się w jego próbie. Niektórzy nie mają telefonów, mają zastrzeżone numery lub innego operatora. Oznacza to, że mimo inteligentnego losowania dobór nie jest losowy, a wyniki z próby nie mogą być uogólnione na populację mieszkańców jego miasta.

Zobacz też [edytuj | edytuj kod]

- dobór próby
- dobór celowo-losowy
- dobór celowy

Kategoria: Dobór próby statystycznej

Rysunek 4.1: Przykładowy artykuł z portalu Wikipedia. Na niebiesko wyróżnione zostały linki do innych tekstów z portalu. Źródło: `http://pl.wikipedia.org/wiki/Dob%C3%B3r_losowy`

4.2. Wstępne przetwarzanie danych

Ważnym elementem przy pracy z danymi jest ich wstępna obróbka, szczególnie gdy są to dane tekstowe. Jednocześnie nie ma ogólnych, odpowiednich dla wszystkich zagadnień, reguł postępowania – schemat działania trzeba dostosować pod konkretny problem i posiadane dane. W pewnych szczególnych przypadkach kwestię wstępnej obróbki danych można pominąć, jednak prawie zawsze wiąże się to z ryzykiem negatywnego wpływu na działanie algorytmu. Co więcej w przypadku gdy szczególnie istotne są słowa znajdujące się w tekście, przygotowanie danych może okazać się jedną z kluczowych kwestii, jeśli chodzi o jakość działania algorytmu.

Jak wspomniano wcześniej pobrane dane składają się w dużej mierze z treści sformułowanych w języku naturalnym, jak i zawartości technicznej takiej jak linki czy znaczniki języka HTML. Określenie istotności treści zawartej w poszczególnych częściach jest zasadniczym problemem przy wstępnej obróbce danych. Przyjmijmy, że część związaną z językiem naturalnym będziemy nazywać „tekstową”, a pozostałe części tekstu – „techniczną”. Związane są z nimi następujące aspekty (zaznaczmy, że rozważania te wymagają nieformalnego podejścia, intuicji oraz początkowego przejrzenia tekstów, co jest nieodłączną częścią praktycznej analizy danych):

- Część tekstowa zawiera główny opis artykułu, można się więc spodziewać, że w tej części zawarte zostanie meritum tekstu.
- Linki składają się ze słowa, które pojawia się w tekście, jak i odniesienia do innej strony. Ta pierwsza część może więc zawierać dużo informacji o temacie tekstu.
- Wzory oraz kody mogą dużo powiedzieć o tematyce artykułu (np. bardzo łatwo odróżnić wzór chemiczny od matematycznego), jednak w większości składają się one z krótkich ciągów znaków (np. pojedynczych liter), które mogą być charakterystyczne dla wielu problemów.

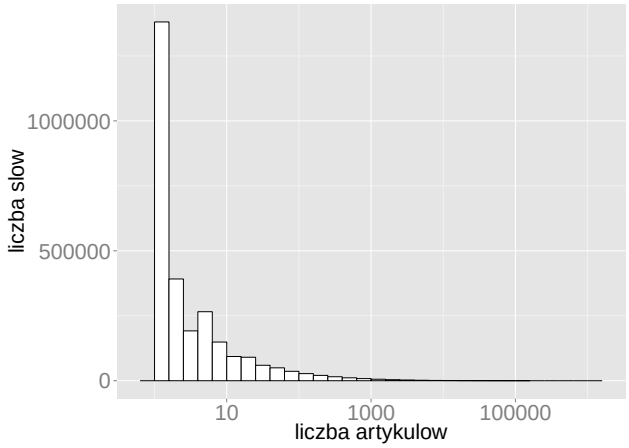
- Cytaty mogą być ważne, choć często mogą mocno odbiegać od głównej tematyki tekstu.
- Przypisy są dodatkową informacją zawartą w tekście, często poruszające tematy poboczne.
- Odniesienia bibliograficzne, choć ważne z punktu widzenia wartości treści zawartych w artykule, nie wnoszą istotnych informacji o tematyce artykułu.
- Część artykułów zawiera bardzo dużo tabel, które czasem stanowią niemal jedyną treść. Jednakowoż służą one uporządkowaniu wiedzy i treść w nich zawarta często nie wnosi istotnych informacji o tematyce tekstu, zawierając jedynie słowa hasłowe bądź wylistowania danych zagadnień.
- Podpisy pod rysunkami są zazwyczaj powtórzeniem zdań bądź ich fragmentów z części opisowej.
- Spis treści zawiera tytuły, które są następnie powtórzone w tekście.
- Sekcja „Zobacz też" może być ważna, gdyż wskazuje na połączenia tematyczne między tekstami.
- Podobnie kategorie wskazują w szczególności na tematy poruszanego zagadnienia.
- Znaczniki języka HTML oraz wyróżnienia powyższych elementów nie wnoszą żadnej informacji o tematyce tekstu i służą jedynie odpowiedniemu wyświetleniu treści.

Wstępną obróbkę danych przeprowadzamy uwzględniając powyższe aspekty. Początkowo z części tekstowej usuwamy wszystkie znaki nie będące literami alfabetu języka polskiego. Poddajemy jej dalszej obróbce po przetworzeniu części technicznej. Słowa, które występują w tekście, a pod którymi znajduje się link, pozostawiamy bez zmian. Wzory oraz kody usuwamy w całości ze względu na trudność w rozróżnieniu czego podany fragment dotyczy oraz ze względu na krótkie znaki, które zawierają. Cytaty pozostawiamy bez zmian. Przypisy usuwamy z artykułów, jako że zawierają jedynie dodatkową, z punktu widzenia głównego tekstu, treść. Źródła bibliograficzne usuwamy w całości. Teksty oczyszczamy także z tabel, rysunków, podpisów pod nimi oraz spisu treści. Sekcję „Zobacz też" oraz kategorie pozostawiamy jako potencjalne źródło podobnej tematyki do tej zawartej w tekście. Wszelkie znaczniki HTML-owe usuwamy jako bezwartościowe z punktu widzenia treści artykułu. Podobnie teksty oczyszczamy z tytułów, które pojawiają się w większości tekstów, tj. *Zobacz też, Linki zewnętrzne* oraz *Bibliografia*.

Tak otrzymane teksty dzielimy na słowa, które przekształcamy do wyrazów o małych literach. Nie chcemy rozróżniać słów ze względu na wielkość liter, gdyż nie powinna ona mieć znaczenia dla tematyki treści. Do każdego tekstu dodajemy informację o tym, jakie słowa i w jakich licznościach w nim występują. Innymi słowy przechowujemy informację o tekstach w ten sposób, że możemy przedstawić zbiór tekstów jako macierz M, gdzie element m_{ij} oznacza liczność j-tego słowa w i-tym artykule. W ten sposób otrzymujemy 2 806 765 unikalnych słów we wszystkich, tj. w 1 075 568, artykułach.

Otrzymane dane przechowujemy w bazie danych SQLite, w 7 tabelach, określających tytuły, kategorie, graf linków między tekstami, słowa oraz tabelę z informacją o tym jaki tekst zawiera jakie słowa w jakich ilościach. Przetworzenie danych wejściowych i utworzenie bazy danych zajęło ok. 12 godzin.

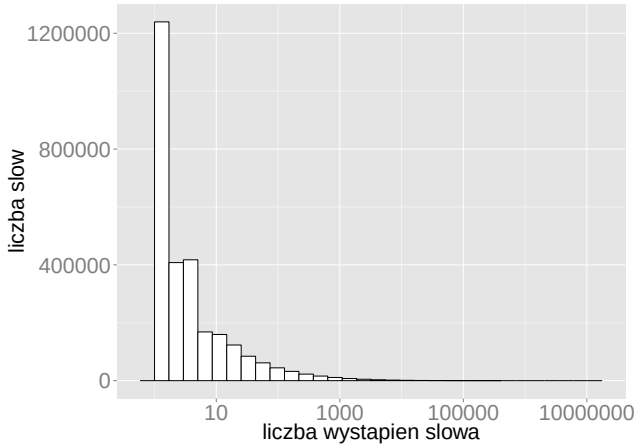
Przejdźmy do podstawowych statystyk dotyczących przetworzonych danych. 49% słów występuje tylko w jednym tekście (zob. Rys. 4.2). Nieco ponad $3,5\%$ słów znajduje się w stu



Rysunek 4.2: Histogram mówiący w ilu tekstach wystąpiło słowo.

i więcej artykułach, natomiast jedynie $0,6\%$ wszystkich wyrazów pojawia się w więcej niż tysiącu tekstów.

44% słów występuje dokładnie raz we wszystkich artykułach (zob. Rys. 4.3). Prawie $4,3\%$ wyrazów pojawia się ponad sto razy, natomiast mniej niż jeden procent słów występuje tysiąc i więcej razy.



Rysunek 4.3: Histogram liczby wystąpień słów we wszystkich artykułach.

Słowa występujące tylko w jednym tekście to zazwyczaj słowa w obcych językach, przykładowo: `jużdortransstroj, youtsos, odety, knežlaz, pallebitzke, rulicach, werkowie`, `rumilla, metyklotiazyd, bazelak`, choć czasem są to słowa będące odmianą słów częściej występujących, np. słowo `uchybiają` jest odmianą czasownika `uchybiać`. Takie słowa warto wziąć pod uwagę przy analizie, gdyż są „podobne" do słów częściej występujących, a więc mogą polepszyć jakość dopasowania pod względem tematycznym.

Słowa najczęściej występujące prezentuje Tabela 4.1. W większości przypadków są to słowa nie istotne w kontekście analizy tematycznej tekstu (ang. *stopwords*). Takich słów można wyróżnić więcej, np.

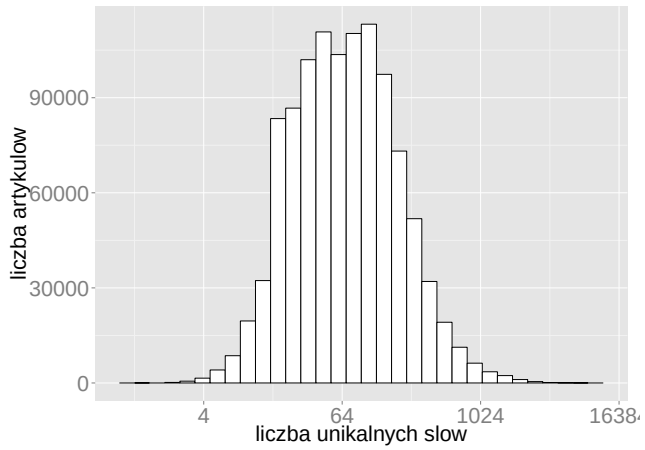
`ach, aj, albo, bardzo, bez, bo, być, ci, cię, ciebie, co, czy, daleko, dla, dlaczego, dlatego, do, dobrze, dokąd, dość, dużo, dwa, dwaj, dwie, dwoje,`

Tabela 4.1: Lista piętnastu najczęściej występujących słów. Kolumna oznaczona jako *Liczba artykułów* oznacza liczbę tekstów, w których wystąpiło dane słowo, natomiast ostatnia kolumna mówi o ilości wystąpień wyrazu ogółem we wszystkich artykułach.

	Słowo	Liczba artykułów	Liczba wystąpień
1	w	1 003 961	10 330 250
2	i	726 209	4 290 653
3	na	689 559	3 215 428
4	z	649 564	3 252 352
5	do	559 672	2 297 910
6	się	537 360	2 094 912
7	roku	420 178	1 292 537
8	a	378 941	919 278
9	od	378 327	935 799
10	jest	343 846	993 143
11	przez	336 596	852 463
12	oraz	288 718	663 760
13	po	286 818	765 075
14	o	273 574	674 431
15	ur	241 888	365 934

```
dziś, dzisiaj, gdyby, gdzie, go, ich, ile, im, inny, ja, ją, jak, jakby, ja-
ki, je, jeden, jedna, jedno, jego, jej, jemu, jeśli, jest, jestem, jeżeli,
już, każdy, kiedy, kierunku, kto, ku, lub, ma, mają, mam, mi, mną, mnie, moi,
mój, moja, moje, może, mu, my, na, nam, nami, nas, nasi, nasz, nasza, na-
sze, natychmiast, nią, nic, nich, nie, niego, niej, niemu, nigdy, nim, nimi,
niż, obok, od, około, on, ona, one, oni, ono, owszem, po, pod, ponieważ,
przed, przedtem, są, sam, sama, się, skąd, tak, taki, tam, ten, to, tobą,
tobie, tu, tutaj, twoi, twój, twoja, twoje, ty, wam, wami, was, wasi, wasz,
wasza, wasze, we, więc, wszystko, wtedy, wy, żaden, zawsze, że
```

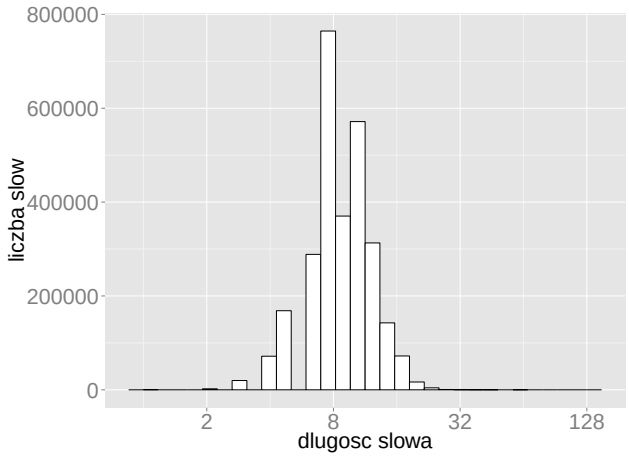
Te i podobne słowa oraz wyrazy jedno-, dwu- i trzyznakowe, należy usunąć ze zbioru danych, gdyż nie wnoszą żadnej istotnej informacji o tematyce tekstu. Dokonujemy tego w kolejnym etapie wstępnej obróbki tekstów, o czym powiemy w następnym podrozdziale.



Rysunek 4.4: Histogram liczby unikalnych słów w artykule.

Średnia liczba unikalnych słów występujących w artykule wynosi 121, mediana to zaledwie 66 unikalnych wyrazów. W 2 272 tekstach wystąpiło mniej niż pięć słów. Artykuły te zawierały przede wszystkim tabele i rysunki, stąd po wstępnej obróbce danych pozostało w nich niewiele wyrazów. Histogram liczby unikalnych słów w tekstach przedstawia Rys. 4.4.

Mediana długości słów wynosi 9, średnia jest nieco wyższa (zob. Rys. 4.5). Najkrótsze występujące słowa są jednoznakowe, jest ich 249. Najdłuższe słowo ma 128 znaków. Słowa o długości ponad 11 znaków stanowią $19,5\%$ wszystkich słów.



Rysunek 4.5: Histogram długości słów.

4.3. Utworzenie skupień „podobnych" słów

Po wstępnej obróbce danych dostajemy 2 806 765 unikalnych słów, z czego blisko połowa występuje jedynie raz we wszystkich tekstach. Algorytm dzielący teksty tematycznie bierze pod uwagę liczności słów, które znajdują się w artykule a daną wejściową jest macierz o wymiarach liczba artykułów $\times$ liczba słów. Jeśliby nie przetworzyć dalej danych, to macierz ta miałaby wymiary 1 075 568 $\times$ 2 806 765, co w postacie gęstej daje ponad 12 TB pamięci masowej. Ponadto przeważająca większość rekordów byłaby równa zeru – rzadkość danych wejściowych wynosi ponad $99,99\%$. Algorytm mógłby nie poradzić sobie z tak dużą ilością danych, zwłaszcza gdy większa część rekordów jest „niewypełniona". Stąd zachodzi potrzeba zmniejszenia wymiaru danych.

Jak wspomnieliśmy wcześniej ze zbioru słów należy usunąć wyrazy nieistotne w kontekście analizy tematycznej. Co więcej warto również zredukować liczbę analizowanych słów o wyrazy, które występują bardzo często. Powód jest następujący: jeśli dane słowo pojawia się w dużej liczbie artykułów, to zachodzi podejrzenie, że wyraz ten nie ma jasnego kontekstu tematycznego. Przykładowo słowo `strona` może odnosić się do strony w konflikcie, kartki czy też kierunku, a więc może dotyczyć różnych tematów. Tego typu wyrazów wyznaczyliśmy 350. Z drugiej strony weźmy po uwagę słowa, które występują bardzo rzadko, np. w jednym lub dwóch tekstach. Artykuł, w którym takie słowo występuje nie da się ich połączyć z żadnym innym tekstem. Zatem wyrazy takie również usuwamy ze zbioru analizowanych słów. Stanowią one ponad 63% wszystkich wyrazów. Dalej zbiór wyrazów redukujemy również o słowa bardzo krótkie, tj. jedno-, dwu- i trzyznakowe, które zazwyczaj są nieistotne w kontekście analizy tematycznej. Takich słów jest 22 145. Łącznie usuwamy ze zbioru 1 787 445 słów, co stanowi prawie 64% wszystkich wyrazów i dostajemy 1 019 320 słów do dalszej analizy.

Ponieważ rekordy składają się z wyrazów, można wyznaczyć skupienia słów „podobnych" do siebie. Odmienność wyrazów można określić za pomocą odległości na przestrzeni ciągów znaków opisanych w rozdziale 2. Następnie dany tekst można przedstawić jako wektor $(n_1, \ldots, n_K)$, gdzie n_i jest licznością słów z i-tej grupy, $i = 1, \ldots, K$, a K liczbą grup słów. Przykładowo, jeśli skupienie A składa się ze słów s, t, u i wystąpiły one w danym artykule, odpowiednio, x, y, z razy, to słowa ze skupienia A wystąpiły w tym tekście łącznie $x+y+z$ razy. Dzieląc wszystkie wyrazy na skupienia, możemy znacząco zmniejszyć liczbę wyrazów, biorąc pod uwagę liczności grup słów, zamiast liczności pojedynczych wyrazów.

Aby podzielić zbiór słów na skupienia, najbardziej naturalne wydaje się zbudowanie macierzy odległości wszystkich słów od siebie i zastosowanie algorytmu aglomeracyjnego. Jednakowoż wiąże się to z dużą złożonością pamięciową i obliczeniową, stąd też podejście takie nie zostało wykorzystane w niniejszej pracy. Inny pomysł polega na przyłączaniu do skupienia słów dopóty, dopóki średnia odległość w zbiorze nie przekroczy zadanej liczby. Wstępne testy na losowej próbce tysiąca słów wykazały, że jakość takiego podziału jest słaba, a czas obliczeń względnie długi.

Stemming. Stąd postanowiono dokonać podziału zbioru słów w inny sposób. Początkowo przeprowadzamy tzw. *stemming*, czyli sprowadzenie słowa do jego rdzenia. Takie podejście opiera się na założeniu, że odmiana słowa nie zmienia jego tematyki, a dzięki temu możemy znacznie ograniczyć liczbę unikalnych słów w zbiorze. Przykładowo słowa `zjednoczonych`, `zjednoczyli`, `drużynom`, `drużynie` zostaną sprowadzone, odpowiednio, do form `zjednoczyć`, `drużyna`. Dzięki takiemu podejściu, każde skupienie będzie miało swoje *słowo-reprezentanta* (środek czy też centroid), które jednoznacznie charakteryzuje podzbiór. Będziemy je nazywać środkiem, słowem-reprezentantem lub po prostu *reprezentantem* skupienia. W ten sposób do odpowiedniego podzbioru słów możemy odnosić się poprzez jego reprezentanta.

Tabela 4.2: Liczba słów na których zastosowano *stemming* w poszczególnych językach. Procent mówi o odsetku jaki dany język stanowił spośród wszystkich analizowanych słów.

	Język	Liczba słów	Procent ogółu
1	polski	430 401	42,2%
2	angielski	27 274	2,7%
3	francuski	8 126	0,8%
4	niemiecki	7 695	0,8%
5	ogółem	473 496	46,5%

Do przeprowadzenia *stemmingu* używamy programu *Hunspell* [3] – korektora pisowni i analizatora morfologicznego używany w wielu programach typu *open source*. Aplikacja ta ma wbudowany słownik słów języka polskiego wraz z ich odmianami. Do każdego wyrazu jest też przypisany jego rdzeń. Sposób działania jest następujący: program znajduje szukane słowo w słowniku, a następnie zwraca jego rdzeń lub nie zwraca nic, jeśli słowo nie pasuje do żadnego wyrazu ze słownika. W szczególności słownik nie zawiera wyrazów, w których popełniono literówki, nie użyto znaków diakrytycznych czy też złączeń słów.

Ponieważ część wyrazów stanowią słowa obcojęzyczne przeprowadzamy *stemming* w języku polskim, angielskim, niemieckim oraz francuskim (zob. Tabela 4.2). W ten sposób grupujemy 473 496 słów, co stanowi ok. 47% wszystkich słów, w 137 223 skupienia. Przykładowe skupienia prezentuje Tabela 4.3. Warto zauważyć, że słowa w podzbiorach są podobne tematycznie, choć do skupień o reprezentantach `niemiecki` oraz `odkryty` trafiły wyrazy o

[3] `http://hunspell.sourceforge.net/`

Tabela 4.3: Przykładowe skupienia uzyskane przy pomocy *stemmingu*.

Reprezentant	Słowa w skupieniu
działalność	działalność, działalności, działalnością, działalnościach, działalnościami
niemiecki	niemiecki, niemieckiej, niemieckiego, niemieckich, niemieckim, niemiecką, niemiecka, niemieccy, niemieckimi, niemiecku, niemieckiemu, nieniemieckich, nieniemieckiej, nieniemieckie
odkryty	odkryta, odkryte, odkryty, odkrytych, odkrytym, odkrytą, odkrytego, odkrytej, odkrytymi, nieodkrytych, nieodkryte, odkrytemu, odkryci, nieodkrytego, nieodkryta, nieodkrytej, nieodkrytą, nieodkrytymi
okres	okres, okresu, okresach, okresem, okresy, okresów, okresami, okresom
postać	postaci, postacie, postać, postacią, postaciami, postaciach, postaciom, postał, postała, postania, postało, postały, postaniu, postanie, postali
praca	pracę, pracach, praca, pracą, pracami, praco
produkcja	produkcji, produkcja, produkcję, produkcją, produkcje, produkcjach, produkcjami, produkcjom
wschód	wschód, wschodu, wschodzie, wschodowi, wschodem
wydawnictwo	wydawnictwo, wydawnictwa, wydawnictw, wydawnictwie, wydawnictwem, wydawnictwach, wydawnictwami, wydawnictwu, wydawnictwom
występ	występy, występów, występ, występu, występach, występie, występem, występami, występom, występowi
występować	występował, występujących, występujący, występowania, występujące, występowanie, występowała, występują, występować, występująca, występowali, występującego, występowały, występującym, występującej, występowało, występującą, występowaniem, występującymi, występowaniu, występuję, występującemu, występowano, występowałyby, występowałby, występowań, występowałaby, występowałoby, występuj, występujemy, występowaliśmy

znaczeniu przeciwnym. Widać więc, że podział taki nie jest idealny, choć z drugiej strony ich grupa znaczeniowa jest podobna.

Podział przy użyciu metryk. Tak zaproponowany podział słów na skupienia wykorzystuje jedynie ok. 47% zbioru wszystkich wyrazów. Co więcej większość z podzbiorów jest zaledwie kilkuelementowa (zob. tabela 4.4).

Tabela 4.4: Rozkład liczby słów w skupieniu.

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1	1	2	3	4	46

Stąd można zastosować następujące schematy postępowania, które po pierwsze zredukują liczbę używanych grup słów, a po drugie wykorzystają dodatkowo zbiór niepogrupowanych wyrazów. Procedury te polegają na:

1. Dołączeniu do skupień słów jeszcze niepogrupowanych.
2. Dołączeniu do skupień zawierających pięć i więcej elementów, podzbiorów o mniejszej liczności.
3. Zastosowaniu najpierw punktu 1, a następnie punktu 2.

Omówimy teraz bliżej na czym polegają powyższe kroki.

Procedura 1. W kroku tym chcemy użyć większej liczby dostępnych słów. Dzięki temu zwiększą się liczności występowania grup słów w tekście, co być może polepszy jakość podziału artykułów na skupienia. Algorytm ten jednak nie zmieni liczby skupień.

Schemat działania jest następujący: bierzemy słowo nieprzydzielone do żadnego skupienia. Liczymy odległość tego wyrazu (przy użyciu odległości zdefiniowanych w rozdziale 2) od wszystkich słów-reprezentantów dotychczasowo otrzymanych skupień. Słowo przydzielamy do tego podzbioru, do którego reprezentanta było mu najbliżej (zob. Algorytm 4.4). Jeśli wyraz ma taką samą (najmniejszą) odległość do kilku środków, wybieramy pierwszy w kolejności. Jeśli odległość słowa do wszystkich reprezentantów jest nieokreślona lub równa nieskończoności, to wyraz ten pomijamy, tj. nie dodajemy go do żadnego ze skupień. Taka sytuacja może się zdarzyć, w przypadku zastosowania odległości opartych na q-gramach, gdy długość słowa jest mniejsza od q.

Algorytm 4.4 Algorytm przydzielający niepogrupowane słowo do skupienia.

dane: zbiór reprezentantów R, zbiór słów do kategoryzowania W, odległość d
for $w \in W$ **do**
    $C(w) = \arg \min_{i:\ r_i \in R} d(w, r_i)$
    if $\text{length}(C(w)) > 1$ **then**
        $C(w) = C(w)[1]$
    end if
end for

Zauważmy, że powyższy algorytm podobny jest do metody aglomeracyjnej analizy skupień z użyciem odmienności centroidu. Różnice polegają na tym, że do utworzonych już skupień przyłączamy pojedyncze obserwacje (słowa), tzn. liczba skupień jest z góry ustalona.

Procedura 2. W tym kroku chcemy zredukować liczbę uzyskanych skupień. Dzięki temu zwiększą się liczności występowania grup słów w tekście, a ponadto zmniejszy się liczba skupień, co może przyczynić się do lepszego działania algorytmu dzielącego teksty na skupienia.

Schemat działania jest następujący (zob. Algorytm 4.5): sprawdzamy, jakie są liczności wszystkich skupień. Jeśli liczność skupienia jest większa od pięciu, to taki podzbiór oznaczamy jako „duży". Do takich skupień będziemy przyłączać mniejsze podgrupy. Jeśli liczność skupienia jest mniejsza lub równa 4, to podzbiór oznaczamy jako „mały". Takie skupienie będziemy przyłączać do podzbiorów „dużych". Te pierwsze skupienia nazwijmy dużymi skupieniami, natomiast te drugie – małymi.

Mając tak podzielone skupienia, weźmy reprezentantów małych podzbiorów. Jeśli długość słowa-reprezentanta nie przekracza trzech znaków, to skupienie takie pomijamy w dalszej analizie. Ma to na celu uniknięcie analizy słów, które nie mają znaczenia, jak zbitek dwóch lub trzech takich samych liter (np. `aa` lub `bbb`). Następnie postępowanie jest podobne jak w Algorytmie 4.4: liczymy odległość środka małego skupienia od wszystkich słów-reprezentantów dużych skupień. Sprawdzamy, która z wyliczonych odległości była najmniejsza. Podzbiór, którego reprezentantem jest analizowane słowo, przydzielamy do tego skupienia, do którego środka było mu (słowu) najbliżej. Jeśli wyraz ma taką samą (najmniejszą) odległość do kilku reprezentantów, wybieramy pierwszego w kolejności.

Zauważmy, że powyższy algorytm podobny jest do metody aglomeracyjnej analizy skupień z użyciem odmienności centroidu. Różnice polegają na tym, że do utworzonych już skupień

Algorytm 4.5 Algorytm łączący małe i duże skupienia.

dane: zbiór reprezentantów R, wektor wielkości skupień $\mathbf{s}$, odległość d
$R_m = \emptyset, R_d = \emptyset$
for $r \in R$ **do**
    if $s_r \leq 4$ **then**
        $R_m = R_m \cup r$
    else
        $R_d = R_d \cup r$
    end if
end for
for $w \in C_m$ **do**
    if $|w| < 4$ **then**
        continue
    end if
    $C(w) = \arg \min_{i:\ r_i \in R_d} d(w, r_i)$
    if length$(C(w)) > 1$ **then**
        $C(w) = C(w)[1]$
    end if
end for

przyłączamy inne skupienia, z tym że podzbiory, do których małe skupienie może zostać dołączone, są z góry ustalone.

Procedura 3. Krok trzeci polega na wykonaniu najpierw procedury pierwszej, a następnie drugiej.

Wybór odległości. Mając trzy powyższe algorytmy, możemy przystąpić do dalszej obróbki zbioru słów. Zanim to jednak nastąpi należy wybrać odległości na przestrzeni napisów, dzięki którym będzie to możliwe. W rozdziale 2 przedstawiono pięć odległości opartych na operacjach edycyjnych, trzy odległości oparte na q-gramach oraz dwie miary heurystyczne.

Odległość Hamminga odrzucamy, gdyż można ją sensownie zastosować jedynie na napisach o tej samej długości. Odległości najdłuższego wspólnego podnapisu, Levenshteina, optymalnego dopasowania napisów i Damerau-Levenshteina różnią się jedynie zbiorem bazowych operacji edycyjnych, często dając tę samą odległość. Stąd postanowiliśmy użyć dwóch „skrajnych" odległości, tj. takich, które pozwalają na najmniejszą i największą liczbę bazowych operacji edycyjnych, czyli odległość najdłuższego wspólnego podnapisu (lcs) i Damerau-Levenshteina (dl).

Z odległości opartych na q-gramach wyselekcjonowaliśmy odległość Jaccarda (jac) oraz q-gramową (qg) jako najbardziej reprezentatywne. W obu przypadkach wybraliśmy $q = 4$. Dzięki takiemu podejściu unikniemy przetwarzania słów o długości mniejszej niż cztery znaki.

Z miar heurystycznych wybraliśmy odległość Jaro, czy też Jaro-Winklera z $p = 0$.

Otrzymane zbiory. Na zbiorze skupień otrzymanym po wykonaniu *stemmingu* zastosowano trzy powyższe algorytmy przy użyciu każdej z czterech odległości, dostając łącznie 16 różnych zbiorów skupień (wliczając w to zbiór, otrzymany ze *stemmingu*). Cała procedura utworzenia grup słów zajęła prawie 50 godzin.

Grupy słów oznaczmy jako *clust_X* gdzie X jest przyrostkiem oznaczającym algorytm i zastosowaną odległość. Oznaczenia są następujące: w przypadku, gdy dołączaliśmy słowa do istniejących skupień (tj. zastosowany był algorytm 4.4) dodajemy jedynie przyrostek oznaczający zastosowaną odległość, tj. *lcs, dl, jw, jac* lub *qg*, np. *clust_lcs*. Jeśli użyliśmy algorytmu 4.5, zmniejszającego liczbę skupień, to dodajemy przyrostek *red_* oraz zastosowaną odległość, np. *clust_red_lcs*. Jeśli oba algorytmy zostały zastosowane, to łączymy je w nazwie, dostając np. *clust_lcs_red_lcs*. Grupę słów otrzymaną po wykonaniu *stemmingu* oznaczamy po prostu *clust*.

Liczbę skupień oraz liczbę słów zawartą w skupieniu dla poszczególnych zbiorów zawiera tabela 4.5. Zbiory, na których zastosowano algorytm 4.4 lub jedynie *stemming* zawierają 137 223 grup wyrazów, co dało redukcję ok. $86,5\%$ względem zbioru słów (tj. 1 019 320). W skupieniach tych znajduje się 976 691 słów, czyli prawie 96% wszystkich wejściowych wyrazów. Druga grupa zbiorów, tj. taka, która jest wynikiem działania Algorytmu 4.5 zawiera dokładnie 33 403 skupienia, co daje redukcję równą 97%. Słowa zawarte w tych skupieniach stanowią ok. 47% wejściowego zbioru wyrazów. Trzecia grupa zbiorów, ma nieco redukcję niż poprzednia i wynosi między 92% a 94%, zawierając jednocześnie ok. 96% wszystkich wyrazów.

Tabela 4.5: Zbiory skupień wraz z ich liczbą oraz liczbą słów w skupieniu. Redukcja oznacza procent zredukowania z wejściowego zbioru słów do liczby otrzymanych skupień. Ostatnia kolumna mówi ile procent wszystkich słów zbioru wejściowego znajduje się w skupieniu.

	Zbiór	Liczba grup słów	Redukcja	Liczba słów	% wsz. słów
1	clust	137 223	86,5%	976 691	95,8%
2	clust_lcs	137 223	86,5%	976 691	95,8%
3	clust_dl	137 223	86,5%	976 691	95,8%
4	clust_jw	137 223	86,5%	976 691	95,8%
5	clust_jac	137 223	86,5%	976 691	95,8%
6	clust_qg	137 223	86,5%	976 691	95,8%
7	clust_red_lcs	33 403	96,7%	473 500	46,5%
8	clust_red_dl	33 403	96,7%	473 500	46,5%
9	clust_red_jw	33 403	96,7%	473 500	46,5%
10	clust_red_jac	33 403	96,7%	473 500	46,5%
11	clust_red_qg	33 403	96,7%	473 500	46,5%
12	clust_lcs_red_lcs	73 101	92,8%	976 691	95,8%
13	clust_dl_red_dl	78 354	92,3%	976 691	95,8%
14	clust_jw_red_jw	79 255	92,2%	976 691	95,8%
15	clust_jac_red_jac	74 642	92,7%	976 691	95,8%
16	clust_qg_red_qg	61 915	93,9%	976 691	95,8%

[TO DO: DODAC PRZYKLADOWE SKUPIENIA DLA WSZYSTKICH 16 ZBIOROW !!!]

4.4. Kateogryzacja tematyczna tekstów

Mając tak zdefiniowane grupy słów możemy przystąpić do podziału zbioru artykułów. Do tego celu użyjemy algorytmu mini-wsadowego k-średnich (Algorytm 3.3 z rozdziału 3). Przypomnijmy, że algorytm ten jest metodą pośrednią pomiędzy algorytmem wsadowym, który w każdej iteracji opiera się na wszystkich obserwacjach, a algorytmem SGD, biorącym w każdej iteracji po jednej obserwacji ze zbioru.

Aby więc użyć algorytmu mini-wsadowego musimy wybrać najpierw liczbę skupień k oraz parametr b, określający ile obserwacji będzie miało swój wkład w każdej iteracji. Zajmijmy się najpierw tą drugą wartością. Ponieważ nie wiemy jak bardzo jakość podziału zależy od parametru b, postanowiliśmy sprawdzić działanie algorytmu dla trzech wartości b: 5 000, 10 000, 35 000. Dostaniemy w ten sposób 48 różnych podziałów, opartych na 16 różnych reprezentacjach tekstów wynikających z otrzymanych grup słów.

Zanim określimy wartość parametru k, zastanówmy się w jaki sposób będziemy mierzyć jakość otrzymanych podziałów. Cztery na pięć zaprezentowanych miar w rozdziale 3 wymaga znajomości prawdziwego podziału zbioru. Przypomnijmy, że nasz zbiór danych to artykuły z polskiej Wikipedii, które mają określoną kategorię tematyczną. Można więc wykorzystać znaną nam wiedzę o kategoriach i na jej podstawie określić jakość podziału otrzymanego w wyniku działania algorytmu. Liczba różnych kategorii, które określają tematykę artykułów wynosi 56 283. Próba wykonania analizy skupień przy użyciu algorytmu mini-wsadowego z tak dużym k, zakończyła się niepowodzeniem, mimo posiadania dużej ilości pamięci RAM (wraz z partycją wymiany (SWAP) ponad 100 GB). Stąd też nastąpiła potrzeba zredukowania tej liczby. Ponieważ struktura kategorii Wikipedii jest drzewiasta[4], można zastąpić kategorię przypisaną do artykułu kategorią ogólniejszą. Po takiej redukcji otrzymano 6 922 grup tematycznych. Jednak rozkład liczby artykułów w otrzymanych kategoriach był mocno skośny. Taka sytuacja jest silnie niesprzyjająca, gdyż chcemy mieć podobne liczności w grupach. Ręcznie podzielono zatem najbardziej liczne tematy na podtematy, natomiast te o najmniejszej liczności połączono zachowując przy tym podobieństwo tematyki. W ten sposób uzyskano sto różnych tematów o podobnym rozkładzie liczby artykułów. Wstępne testy wykazały, że tak otrzymane k pozwala na dokonanie obliczeń w relatywnie krótkim czasie i nie zajmując dużej ilości pamięci RAM.

Wstępne testy na ok. 2% artykułów wykazały dość dobre przyporządkowanie (jednorodność i zgodność na poziomie, ok. $0,3$). Testy na większej próbce ok. 15% tekstów dały wyniki nieco gorsze (jednorodność i zgodność na poziomie nieco poniżej $0,3$). Stąd też postanowiono przeanalizować działanie algorytmu dla trzech różnych liczności próbki, aby sprawdzić, czy jakość działania algorytmu zależy od wielkości próby. Liczba tekstów wziętych do analizy to: 100% (1 075 568 artykułów), ok. 15% (160 000 artykuły) oraz ok. 2% zbioru (25 000 artykułów). Dla tej ostatniej próbki za wartość parametru b, określającego liczbę obserwacji mających wkład w każdej iteracji algorytmu, przyjęliśmy 5 000 oraz 10 000. Większe wartości (tj. 35 000) nie mają sensu, gdyż są większe od liczności próby.

Wszystkie analizy zostały rozpoczęte z tym samym ziarnem losowania, tj. w każdej analizie obserwacje były losowane w tej samej kolejności. Do przeprowadzenia obliczeń użyto 16 komputerów, a łączny czas analiz to ok. 40 godzin. Każdą z reprezentacji tekstów wczytywano jako macierz rzadką, gdzie liczba zerowych elementów wynosiła ok. $99,99\%$. [TO DO !] Niemniej taka reprezentacja zajmowała ok. 3 GB pamięci RAM, natomiast sam proces podziału na skupienia zużywał dodatkowo ok. 4 GB pamięci RAM.

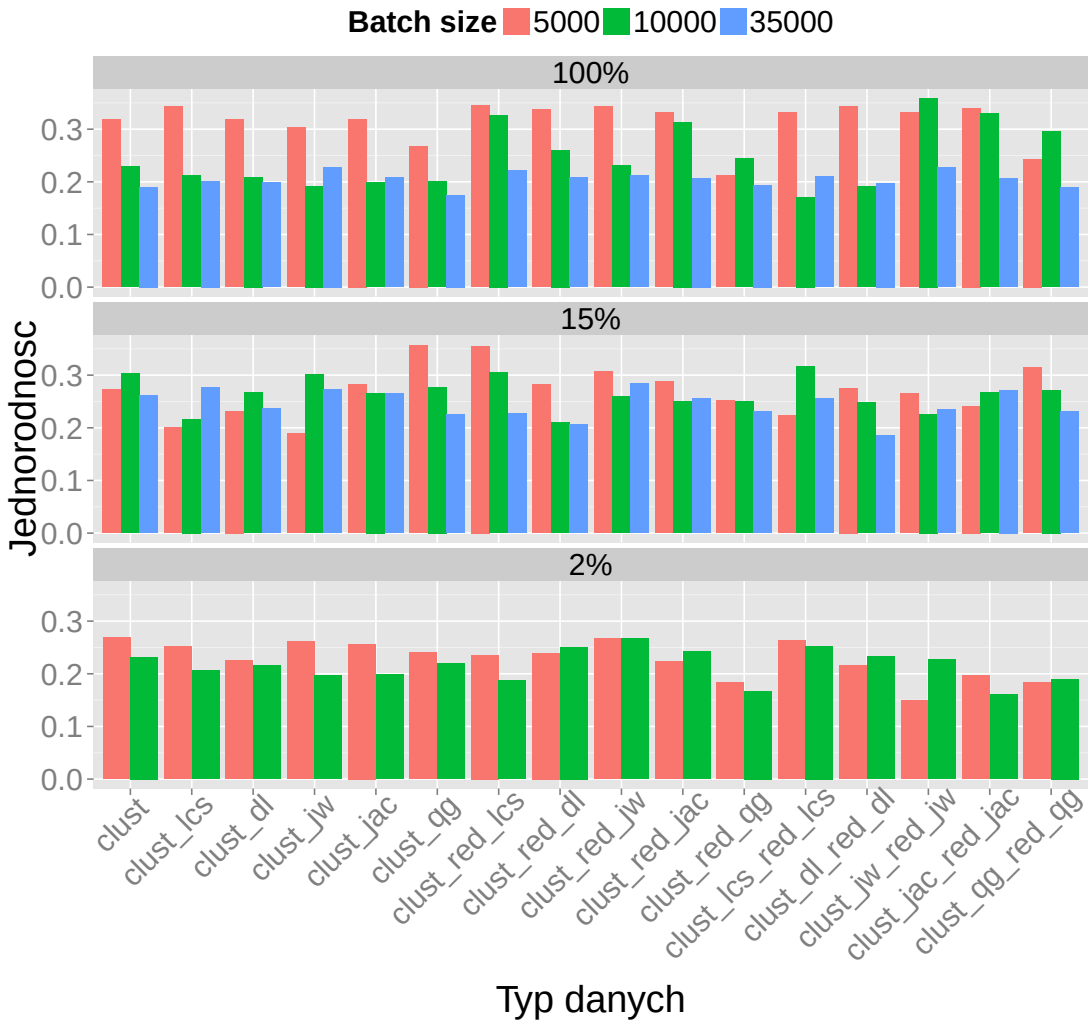
4.5. Analiza wyników

W niniejszym podrozdziale przeanalizujemy uzyskane wyniki dla uzyskanych podziałów tekstów.

[4] por. `http://pl.wikipedia.org/wiki/Wikipedia:Drzewo_kategorii`

Na Rys. 4.6, 4.7, 4.8, 4.9 oraz 4.10 prezentujemy wartości uzyskanych, odpowiednio, jednorodności, zgodności, miary V, indeksu Fowlkesa-Mallowsa oraz miary silhouettes dla wszystkich zbudowanych reprezentacji tekstów i wszystkich wykonanych analiz. Przypomnijmy, że klasami porównawczymi do uzyskanych podziałów są tematy artykułów.

Weźmy pierwszy wskaźnik, tj. jednorodność. Nie widać wyraźnych różnic w wartościach jednorodności w podziale na wielkość zbioru, choć dla całego zbioru wartości te są nieco wyższe niż w pozostałych przypadkach. Średnie wartość jednorodności wynoszą $0,26$ dla 100% i 15% oraz $0,22$ dla 2%. Oznacza to, że większa liczba skupień zawiera teksty z różnych klas (tematów), co jest cechą wysoce niepożądaną. Wynika z tego, że czym większy zbiór, tym lepszy jego podział ze względu na temat.



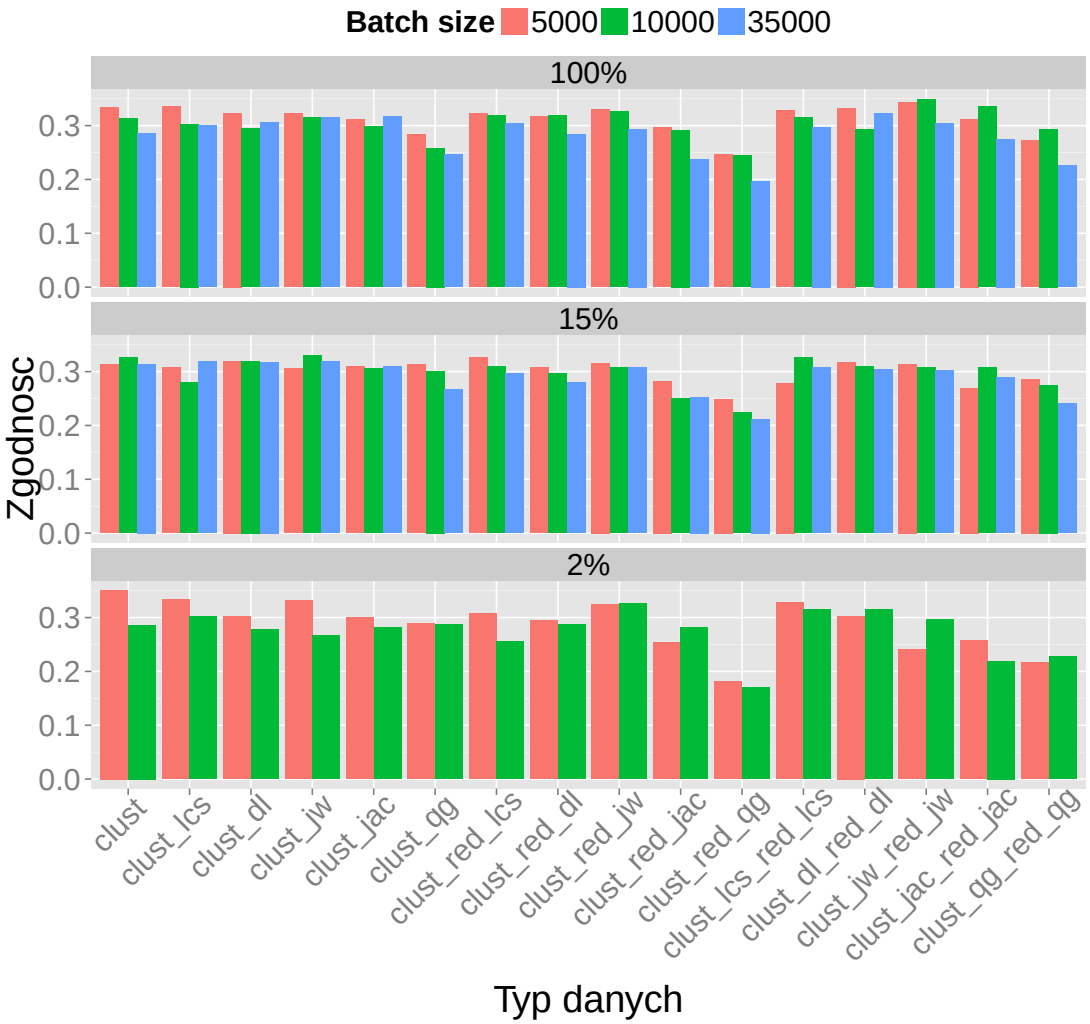
Rysunek 4.6: Jednorodność.

Przyjrzyjmy się teraz wartościom jednorodności ze względu na wielkość parametru b. Można stwierdzić, że w przypadku największego i najmniejszego zbioru mniejsza wartość b daje wyższą wartość jednorodności, co wydaje się być sprzeczne z teorią zaprezentowaną w rozdziale 3.

W końcu spójrzmy na wartości jednorodności w podziale na użyty zbiór grup słów. Najwyższą wartość jednorodności uzyskano dla zbiorów *clust_jw_red_jw* oraz *clust_red_lcs*, *clust_qg* oraz

clust_red_lcs, *clust* oraz *clust_red_jw* dla, odpowiednio, podziału opartego na 100%, 15% oraz 2% zbioru. Wysoką wartość jednorodności uzyskano również dla *clust_lcs_red_lcs*, *clust_lcs*. Stąd też można wnioskować, że pozytywny wpływ na jakość podziału miała procedura zmniejszająca liczbę skupień. Co więcej, najlepsze rezultaty uzyskano przy użyciu odległości Jaro oraz lcs. Warto przy tym zauważyć, że we wszystkich przypadkach najgorsze podziały uzyskano dla odległości q-gramowej.

Przejdźmy do analizy zgodności podziału zbiorów na skupienia. Wartości tej miary zawierały się w przedziale $[0,17, 0,35]$. Dla wszystkich trzech wielkości zbioru uzyskano podobne wyniki i ich średnia wartość wyniosła ok $0,29$. Widać zatem, że wielkość zbioru nie ma istotnego wpływu na wartość zgodności.

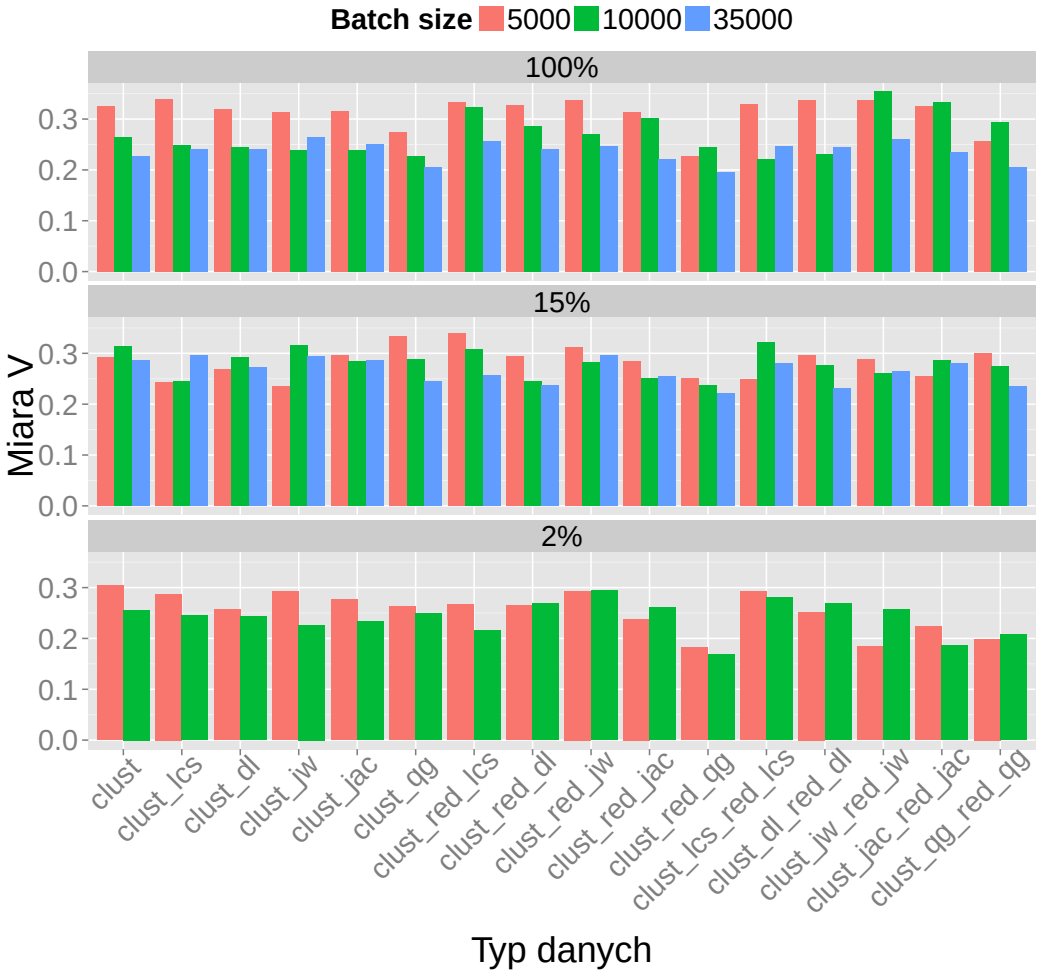


Rysunek 4.7: Zgodność.

Przyjrzyjmy się teraz wartościom zgodności ze względu na wielkość parametru b. Można stwierdzić, że czym mniejsze b, tym wyższa wartość zgodności. Widać to w szczególności dla najmniejszego i największego zbioru.

Dalej spójrzmy na wartości zgodności w podziale na użyty zbiór grup słów. Na pierwszy rzut oka można stwierdzić, że najmniejsze wartości zgodności uzyskano w przypadku zastosowania odległości q-gramowej. Najlepsze rezultaty otrzymano dla grup *clust_jw_red_jw*,

clust_jac_red_jac, *clust_lcs* oraz *clust_jw*. Widać więc, że istotny wpływ na jakość podziału miała procedura przydzielająca niepogrupowane słowa do już istniejących grup słów (Algorytm 4.4). Najlepsze rezultaty dała przy tym odległość Jaro.



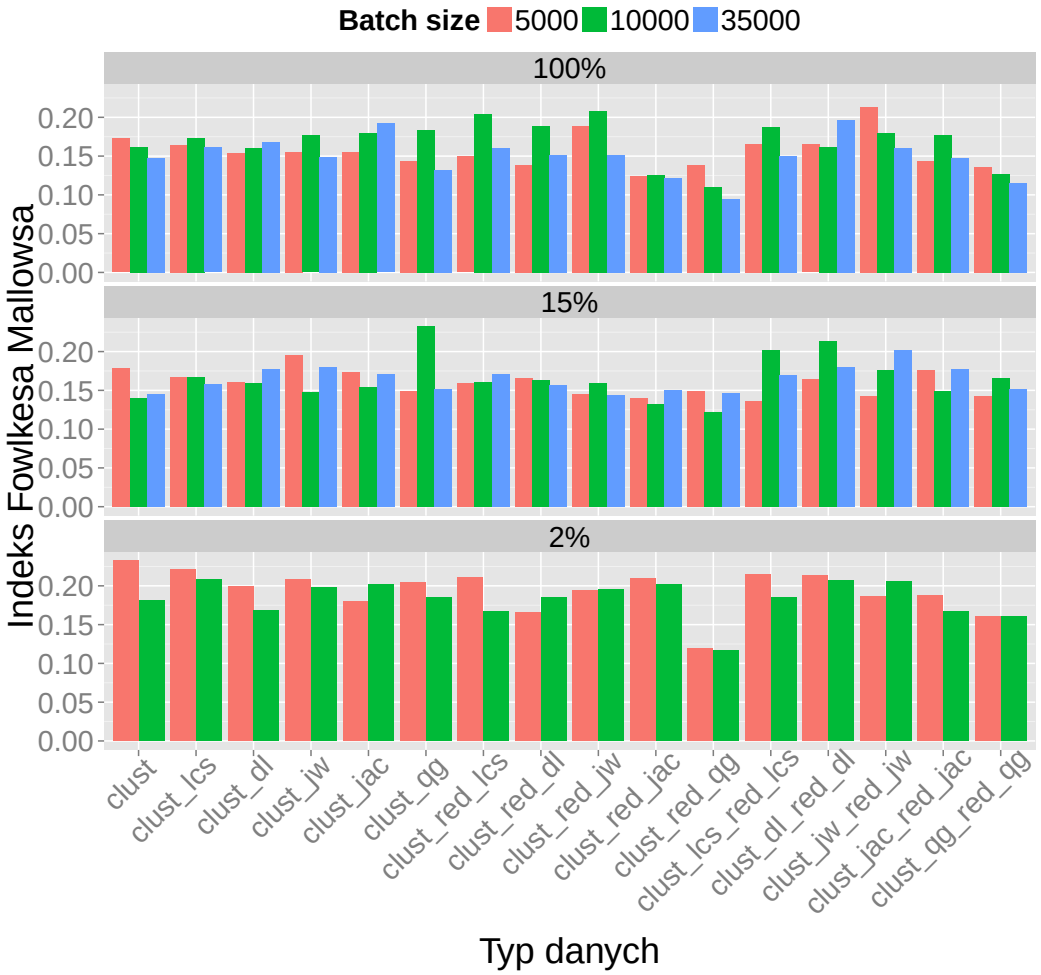
Rysunek 4.8: Miara V.

Ponieważ miara V jest średnią harmoniczną dwóch poprzednich miar, skupimy się jedynie na analizie uzyskanych wyników dla różnych zbiorów wejściowych. Jak można było się spodziewać, najlepsze wyniki uzyskano dla zbiorów, które budowane były przy użyciu odległości Jaro. W drugiej kolejności najlepiej wypadła metryka lcs. Również lepszy podział uzyskano, gdy zastosowano algorytm 4.4 lub najpierw algorytm 4.4, a następnie 4.5.

Przejdźmy do indeksu Fowlkesa-Mallowsa. Najwyższe wartości FM uzyskano dla najmniejszego zbioru, i wynosiły średnio $0,19$; $0,16$ – dla analiz opartych o 15% oraz 100% liczby obserwacji. W tym przypadku więc widać zależność między liczbą obserwacji a wielkością indeksu FM.

Przejdźmy do analizy pod kątem parametru b. W przypadku indeksu FM nie mamy tendencji, że czym mniejsze b, tym wyższa wartość indeksu. Często dla $b = 10\,000$ uzyskano największe wartości tej miary, jednak różnice są niewielkie. Jedynym wyjątkiem jest wartość uzyskana dla *clust_qg* dla 15% obserwacji, gdy miara ta wynosi $0,23$, podczas gdy dla innych wielkości parametru b FM jest równy ok. $0,15$. Można więc uznać ten wynik za wyjątek.

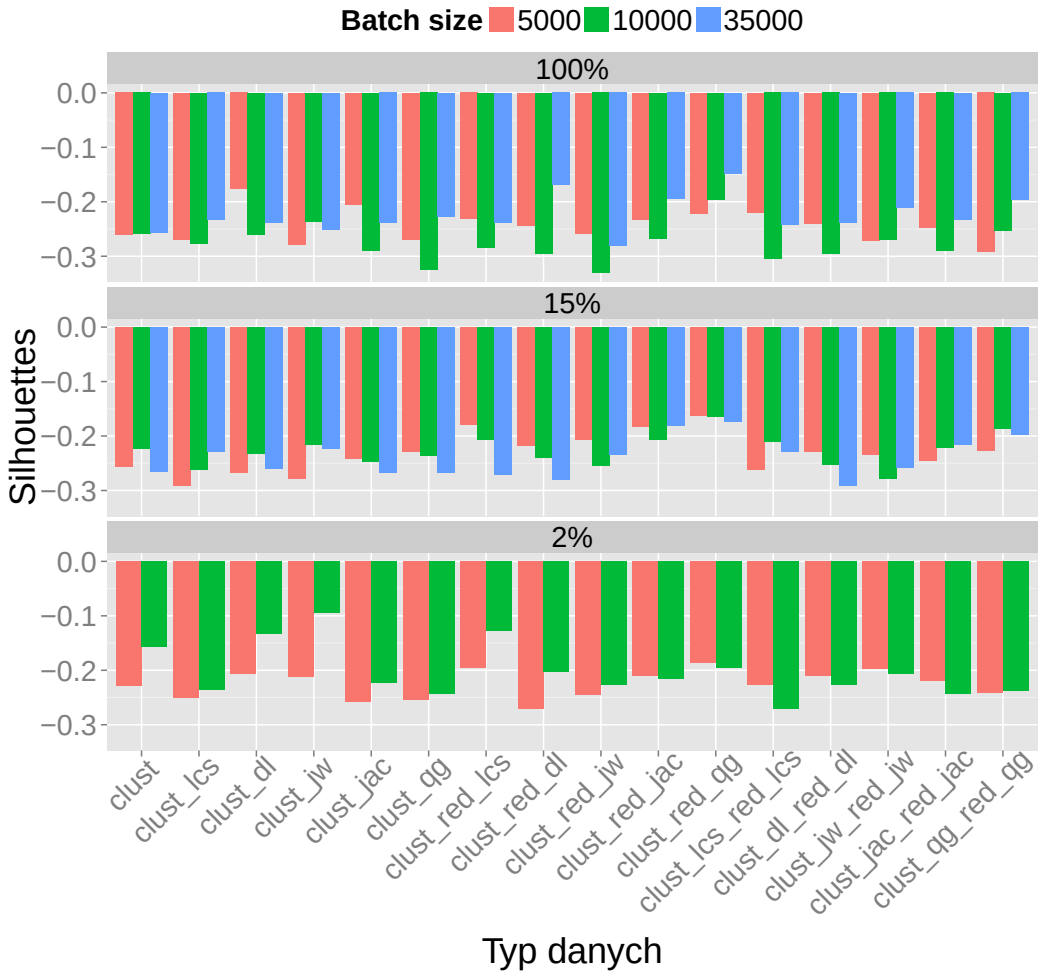
Przyjrzyjmy się teraz wartościom FM pod względem grup słów. Jak wcześniej wspomniano największą wartość wskaźnika uzyskano dla *clust_qg*. Pomijając ten przypadek, najwyższe wartości indeksu FM otrzymano dla *clust*, *clust_lcs_red_lcs*, *clust_jw_red_jw* oraz *clust_dl_red_dl*. Widać więc, że istotny wpływ na wynik analizy miało zastosowanie połączenia Algorytmów 4.4 i 4.5. Co więcej najlepsze rezultaty otrzymano dla odległości lcs, Jaro oraz dl w przypadku wszystkich użytych procedur.



Rysunek 4.9: FMI.

Przejdźmy do miary silhouettes (sylwetek). Ze względu na złożoność obliczeniową niemożliwe było policzenie średniej z sylwetek wszystkich obserwacji. Uzyskane sylwetki są średnią z próbki 10 000 losowych punktów. Wszystkie uzyskane wartości silhouettes są mniejsze od zera, czyli często teksty nie są dobrze przypisane do skupienia, tzn. lepsze byłoby przyporządkowanie danego artykułu do innej grupy niż tej, do której został przypisany.

Największe wartości sylwetek uzyskano dla *clust_red_qg*, co oznacza, że uzyskane przyporządkowania tekstów do skupień były lepsze niż w przypadku pozostałych zbiorów. Najniższe wartości uzyskano dla zbiorów opartych na odległości lcs i Jaro.



Rysunek 4.10: Silhouettes.

Podsumowując, wielkość zbioru, na którym przeprowadzono analizy, nie ma znaczenia na jakość podziału pod względem tematycznym. Uzyskane wartości zaprezentowanych miar nie różniły się znacząco dla różnej liczby analizowanych tekstów. Dalej mniejsze wartości parametru b dawały lepszy podział na skupienia. W przeważającej większości przypadków zaprezentowane miary dawały wyższe wartości dla $b = 5000$ niż w pozostałych sytuacjach.

Najlepsze wyniki uzyskano dla reprezentacji tekstów powstałych przy użyciu odległości Jaro oraz najdłuższego wspólnego podnapisu. Nieco gorsze rezultaty uzyskano dla zbiorów opartych na odległościach Dameru-Levenshteina oraz Jaccarda. Najsłabsze wyniki dawała miara q-gramowa. Dalej lepszy podział dostaliśmy, gdy zbiór przetworzono Algorytmem 4.4, tzn. powiększono grupy słów o niepogrupowane wcześniej wyrazy lub gdy najpierw zastosowaliśmy Algorytm 4.4, a następnie 4.5. Reasumując użycie odległości na przestrzeni ciągów znaków miało pozytywny wpływ na jakość automatycznej kategoryzacji tematycznej tekstów.

[TO DO:
- DLACZEGO WYSZLO TAK SLABO - JESZCZE NIE WIEM
- LICZNOSCI SKUPIEN
- PRZYKLADY JAKIE BYLY ARTYKULY W SKUPIENIACH

- CZY PODZIALY BYLY ZGODNE MIEDZY SOBA
]

4.6. Szczegółowe wyniki

Rozdział 5

Podsumowanie pracy i dalsze kierunki badań

Literatura

[1] Dokumentacja modułu sklearn.metrics języka Python. `http://scikit-learn.org/stable/modules/clustering.html#clustering-evaluation`. Dostęp: 2015-12-01.

[2] Wikipedia - wikipedia, wolna encyklopedia. `http://pl.wikipedia.org/wiki/Wikipedia`. Dostęp: 2015-12-01.

[3] Daniel Aloise, Amit Deshpande, Pierre Hansen, Preyas Popat. Np-hardness of euclidean sum-of-squares clustering. *Machine Learning*, 75(2):245–248, 2009.

[4] Léon Bottou. Stochastic gradient tricks. Grégoire Montavon, Genevieve B. Orr, Klaus-Robert Müller, redaktorzy, *Neural Networks, Tricks of the Trade, Reloaded*, Lecture Notes in Computer Science (LNCS 7700), strony 430–445. Springer, 2012.

[5] Léon Bottou, Yoshua Bengio. Convergence properties of the k-means algorithms. *Advances in Neural Information Processing Systems 7*, strony 585–592. MIT Press, 1995.

[6] Leonid Boytsov. Indexing methods for approximate dictionary searching: Comparative analysis. *Journal of Experimental Algorithmics*, 16:1–91, 2011.

[7] Fred J. Damerau. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176, 1964.

[8] N. R. Dixon, T. B. Martin. *Automatic Speech and Speaker Recognition.* IEEE Press book. IEEE Press, 1979.

[9] Xindong Wu et al. Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1):1–37, 2007.

[10] Ethelbert B. Fowlkes, Colin L. Mallows. A Method for Comparing Two Hierarchical Clusterings. *Journal of the American Statistical Association*, 78(383):553–569, 1983.

[11] Richard W. Hamming. Error Detecting and Error Correcting Codes. *Bell System Techincal Journal*, 29:147–160, 1950.

[12] Trevor J. Hastie, Robert John Tibshirani, Jerome H. Friedman. *The elements of statistical learning: data mining, inference, and prediction.* Springer series in statistics. Springer, New York, 2009.

[13] Matthew A. Jaro. *UNIMATCH: A record linkage system: User manual.* United States Bureau of the Census, 1978.

[14] Joe H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963.

[15] Jacek Koronacki, Jan Ćwik. *Statystyczne systemy uczące się.* Wydawnictwa Naukowo-Techniczne, 2005.

[16] Karen Kukich. Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24(4):377–439, 1992.

[17] G. N. Lance, William T. Williams. A general theory of classificatory sorting strategies 1. hierarchical systems. *The Computer Journal*, 9(4):373–380, 1967.

[18] Vladimir Levenshtein. Binary codes capable of correcting spurious insertions and deletions of ones. *Problems of Information Transmission*, 1:8–17, 1965.

[19] Harry V. Masters. *A study of spelling errors*. University of Iowa Studies in Education, IV, 1927.

[20] Gonzalo Navarro. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, 2001.

[21] Saul B. Needleman, Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48:443–453, 1970.

[22] Olumide Owolabi, Douglas R. McGregor. Fast approximate string matching. *SPE Journal*, 18(4):387–393, 1988.

[23] William M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[24] Andrew Rosenberg, Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning(EMNLP-CoNLL)*, strony 410–420, 2007.

[25] Peter Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20(1):53–65, 1987.

[26] David Sankoff, Joseph B. Kruskal. *Time warps, string edits, and macromolecules: the theory and practice of sequence comparison*. Addison-Wesley, 1983.

[27] D. Sculley. Web-scale k-means clustering. *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, strony 1177–1178, New York, NY, USA, 2010. ACM.

[28] Peter H Sellers. The theory and computation of evolutionary distances: Pattern recognition. *Journal of Algorithms*, 1(4):359 – 373, 1980.

[29] Esko Ukkonen. Algorithms for approximate string matching. *American Journal of Infection Control*, 64(1-3):100–118, 1985.

[30] Esko Ukkonen. Approximate string-matching with q-grams and maximal matches. *Theoretical Computer Science*, 92(1):191 – 211, 1992.

[31] Mark P. J. van der Loo. The stringdist Package for Approximate String Matching. *The R Journal*, 6:111–122, 2014.

[32] Taras K. Vintsyuk. Speech discrimination by dynamic programming. *Cybernetics*, 4(1):52–57, 1968. Russian Kibernetika 4(1):81-88 (1968).

[33] Robert A. Wagner, Michael J. Fischer. The string-to-string correction problem. *Journal of the ACM*, 21(1):168–173, 1974.

[34] Robert A. Wagner, Roy Lowrance. An extension of the string-to-string correction problem. *Journal of the ACM*, 22(2):177–183, 1975.

[35] Anna Wilbik, James M. Keller. A distance metric for a space of linguistic summaries. *Fuzzy Sets and Systems*, 208:79–94, 2012.

[36] William E. Winkler. String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage. *Proceedings of the Section on Survey Research*, strony 354–359, 1990.