Politechnika Wrocławska

Wydział Informatyki i Zarządzania
kierunek studiów: Informatyka
specjalność: Inżynieria oprogramowania

Praca dyplomowa - magisterska

Mechanizmy grupowania dokumentów w automatycznej ekstrakcji sieci semantycznych dla języka polskiego

Bartosz Broda

słowa kluczowe:
WordNet
grupowanie dokumentów
funkcja podobieństwa semantycznego

krótkie streszczenie:
Praca przedstawia wyniki badań nad metodami grupowania dokumentów i metodami wydobywania leksykalnych relacji semantycznych opartych o hipotezę dystrybucyjną pod kątem ich przydatności w automatycznej ekstrakcji sieci semantycznych dla języka polskiego.

Promotor:	Maciej Piasecki	.....	.....
	imię i nazwisko	*ocena*	*podpis*

Wrocław 2007

Spis treści

Streszczenie

Obecnie trwają pracę nad budową Słowosieci — sieci semantycznej leksemów polskich o strukturze podobnej do WordNetu. Jednym z założeń projektu jest rozbudowa leksykalnych relacji semantycznych metodami automatycznymi. Praca ta prezentuje wyniki badań w tej dziedzinie. Przedstawione i porównane zostały metody wydobywania relacji semantycznych oparte o mechanizmy grupowania jak i hipotezę dystrybucyjną.

Abstract

A semantic network of Polish lexemes similar in structure to that of WordNet is currently being under development. Main assumption of the project is to use automatic methods to extend lexico–semantic relations. This thesis presents results of research in this domain. Methods of automatic semantic relations acquisition based on document clustering and distributional hypothesis are discussed and compared exhaustively.

Wprowadzenie

WordNet [Fel98] jest leksykonem semantycznym dla języka angielskiego. Jego początek powstania datowany jest na 1985 rok. Składa się on z ręcznie opisanego zbioru *synsetów* — grup wyrazów będących „prawie synonimami". Każdemu synsetowi jest przypisana krótka definicja i relacja znaczeniowa z innymi synsetami. WordNet jest darmowy[1], obecnie w wersji 3.0 zawiera ponad 155 tysięcy wyrazów pogrupowanych w ponad 117 tysięcy synsetów. Ze względu na te i inne własności stał się powszechnie stosowanym zasobem językowym [Pia]. Ma on wiele zastosowań: od poprawy jakości tłumaczenia maszynowego po ulepszenie systemów wydobywania informacji.

WordNet istnieje nie tylko dla języka angielskiego, ale również m.in. dla francuskiego, czeskiego czy hebrajskiego. Interesującym projektem jest EuroWordNet [Vos02]. Jego celem jest stworzenie zsynchronizowanych wordnetów dla wielu języków europejskich. Od niedawna na Politechnice Wrocławskiej pod kierownictwem dr Macieja Piaseckiego powstaje również polski wordnet — *Słowosieć* [Der07a, Pia07b]. Jest ona tworzona w ramach projektu o tytule[2] „Automatyczne metody konstrukcji sieci semantycznej leksemów polskich na potrzeby przetwarzania języka naturalnego". Cel projektu został sformułowany jako[Pia07b]:

> Głównym celem projektu jest opracowanie algorytmów automatycznego wykrywania relacji semantycznych języka polskiego, które w połączeniu z pracą lingwistów, dadzą efektywną metodę budowy sieci relacji semantycznych.

W projekcie zostało uwzględnionych pięć podstawowych leksykalnych relacji semantycznych [Der07b]. *Synonimia* (relacja szeroko rozumianej bliskoznaczności) określa przynależność słów do jednego synsetu, np. synset tworzą »zdrowie, kondycja, samopoczucie«. Hiperonimia i hiponimia (bardziej ogólny i bardziej szczegółowy) mówią o zawieraniu się znaczeniowym pojęć. Np. »ssak« jest hiponimem »zwierzęcia« i hiperonimem »psa«. Relacja ta definiuje podstawową strukturę sieci i dlatego jest najważniejszym związkiem z punktu widzenia tej pracy.

---

1. http://wordnet.princeton.edu/
2. Projekt badawczy nr 3 T11C 018 29 finansowany przez Ministerstwo Nauki i Szkolnictwa Wyższego

Pozostałymi relacjami są *meronimia i holonimia* (część i całość, np. »klamka« jest częścią »drzwi«), *antonimia* (znaczenie przeciwstawne, np. »wysoki« a »niski«) i *troponimia* (relacja zawierania się pojęciowego jednej czynności w drugiej, np. »utykać« a »chodzić«).

Cel i zakres pracy

Aby przyśpieszyć powstanie i zmniejszyć koszty budowy Słowosieci należy zastosować metody do wspomagania tworzenia ręcznego, krytyki opisanych ręcznie synsetów, bądź tworzenia metodą w pełni automatyczną części sieci.

Dla języka angielskiego zostało opracowane wiele metod służących do wydobywania leksykalnych relacji semantycznych z tekstu czy też budowy leksykonów, zarówno tematycznych jak i ogólnego przeznaczenia. Dla języka polskiego została przeprowadzona nieporównywalnie mniejsza liczba badań niż dla języka angielskiego.

> Dlatego celem pracy było dokonanie analizy porównawczej metod grupowania dokumentów i metod wydobywania leksykalnych relacji semantycznych opartych o hipotezę dystrybucyjną pod kątem ich przydatności w automatycznej ekstrakcji sieci semantycznych dla języka polskiego.

Aby cel został w pełni zrealizowany zostały wyodrębnione następujące zadania:

- Opracowanie przeglądu metod grupowania możliwych do zastosowania w dziedzinie dokumentów tekstowych pod kątem użyteczności w konstrukcji szkieletu dla hierarchicznej sieci semantycznych relacji leksykalnych dla języka polskiego.
- Opracowanie przeglądu metod wydobywania leksykalnych relacji semantycznych opartych o hipotezę dystrybucyjną[3].
- Zaprojektowanie i zaimplementowanie systemu komputerowego umożliwiającego badanie powyższych metod.
- Przeprowadzenie eksperymentów i ocena wyników.

Struktura

Praca składa się z pięciu rozdziałów. W rozdziale pierwszym zostało przedstawione zagadnienie grupowania dokumentów tekstowych, sposobów ich reprezentacji i wydobywania z dokumentów słów reprezentatywnych. Rozdział drugi wprowadza w podstawy teoretyczne dotyczące hipotezy dystrybucyjnej i metod wydobywania wiedzy o semantyce leksykalnej wprost z tekstu. Budowa systemu umożliwiającego przeprowadzenie eksperymentów została opisana w rozdziale trzecim. Opis metodologii badań, wyniki i ich omówienie zostały zebrane w rozdziale czwartym. Rozdział piąty podsumowuje pracę, ocenia zaprezentowane podejście i wskazuje dalsze kierunki badań.

3. Mówi ona o tym, że znaczenie słów i relacje pomiędy nimi mogą zostać wydobyte bezpośrednio z tekstu, zob. rozdział 2.

Rozdział 1

Grupowanie danych w kolekcji dokumentów tekstowych

Grupowanie danych [Maz05] (analiza skupień, np. [Kor05], klasteryzacja, np. [Kor05, Maz05]) jest procesem, który „ma za cel wykrycie skupień, czyli rozłącznych podzbiorów zbioru obserwacji, wewnątrz których obserwacje są sobie w jakimś sensie bliskie, natomiast różne podzbiory są od siebie — w porównaniu z elementami wewnątrz każdego podzbioru — odległe" [Kor05]. Grupowanie jest wykorzystywane w wielu dziedzinach związanych z eksploracją danych (ang. *data mining*). Grupowanie często jest traktowane jako jeden z wczesnych etapów przetwarzania [Pal04, str. 123] w konkretnych aplikacjach. W dziedzinie przetwarzania języka naturalnego w pracy [Man01, str. 497–498] zostają zakreślone dwa główne zastosowania grupowania: poszukiwawcza analiza danych (EDA, ang. *exploratory data analysis*) i generalizacja.

EDA[1] polega na takim przedstawieniu danych, aby możliwe było wyciągnięcie wniosków o całym zbiorze danych, m.in. odkryciu struktury danych, czy znalezieniu anomalii. Bardzo ciekawym przykładem tej techniki jest tworzenie mapy, czyli sposobu wizualizacji kolekcji dokumentów tekstowych, w której każdy dokument jest reprezentowany jako punkt w przestrzeni dwuwymiarowej, a relacja geometryczna pomiędzy punktami określa podobieństwo pomiędzy nimi [Koh00]. System WEBSOM [Lag00] oparty na idei *samoorganizujących się sieci* (SOM, ang. *Self–Organizing Maps*), pozwala na wizualizację w postaci mapy ogromnej kolekcje dokumentów, ułatwiając tym samym nawigowanie po nich. Ciekawy przykład zastosowania sieci SOM do eksploracji dokumentów w dziedzinie astronomii został przestawiony w [Poi98] — dla każdego fragmentu głównej mapy zostaje stworzona mapa pomocnicza pozwalające na dokładniejszą nawigację w podzbiorze kolekcji dokumentów. Inne podejście do EDA w tej dziedzinie prezentuje system BEATCA[Klo06]. Struktura dokumentów jest tworzona przy użyciu *gazu neuronowego* (GNG, ang. *Growing Neural Gas*), a wizualizacja nie jest ograniczona jedynie do dwuwymiarowej przestrzeni, ale też do torusa, cylindra czy sfery.

Generalizacja natomiast jest formą uczenia. Proces grupowania na wyjściu dostarcza grupy, z których można dokonać generalizacji. W [Man01] przedstawiona jest następująca ilustracja tego zagadnienia: po dokonaniu grupowania wyrazów ze względu

1. Termin poszukiwawczej analizy danych wywodzi się ze statystki i oznacza podejście do formowania hipotez oparte na inspekcji zbioru danych.

na podobny kontekst semantyczny i syntaktyczny, możemy wyodrębnić grupę z dniami tygodnia, ponieważ występują one w tekście w języku angielskim w podobnym otoczeniu („until day–of–week", „last day–of–week", itp.). Gdyby w danych treningowych zabrakło frazy „on Friday", to jesteśmy w stanie wydedukować, że taka fraza może być poprawna, ponieważ zaobserwowaliśmy inne dni tygodnia występujące z przyimkiem „on".

W tej pracy grupowanie ma na celu stworzenie hierarchii dokumentów, która stanowić będzie szkielet dla tezaurusa. Hierarchia taka ma za zadanie reprezentować drzewo (lub las drzew[2]) hiperonimii, w taki sposób aby ogólne grupy, których dokumenty omawiają wspólnie szeroką dziedzinę, znajdowały się blisko korzenia hierarchii. Natomiast im bliżej liści drzewa, tym tematyka jest bardziej szczegółowa, zawężona.

Przykładowo skupienia zawierające dokumenty sportowe znajdujące się na szczycie hierarchii powinny poruszać tematykę mówiącą zarówno o piłce nożnej, koszykówce, czy skokach narciarskich. Liście tego drzewa powinny zawierać grupy „czyste" — mówiące tylko o jednej dyscyplinie. Pozwoli to na identyfikację pojęć bardziej ogólnych w grupach dokumentów przy korzeniu hierarchii a bardziej szczegółowych w liściach — jednocześnie odnoszących się do tej samej dziedziny. Dla przykładowej hierarchii sportu oczekujemy, aby wydobytym ogólnym terminem był np. „zawodnik" czy „gracz" a bardziej szczegółowymi „piłkarz", „koszykarz" czy „skoczek". Zagłębiając się dalej w hierarchię znajdziemy dokumenty np. tylko o piłce nożnej, z których będzie można wywnioskować o istnieniu relacji pomiędzy „bramkarzem" a „piłkarzem".

1.1. Grupowanie dokumentów tekstowych

W dalszej części rozdziału zostanie przedstawiony przegląd najistotniejszych metod grupowania, których implementacja pozwoli wydobyć szkielet hierarchii. Zanim to jednak nastąpi należy odpowiedzieć na dwa fundamentalne pytania wynikające z definicji grupowania: co to znaczy, że obserwacje w obrębie jednego skupienia są sobie *„bliskie"*, a co za tym idzie, jak należy *reprezentować dokument* tak, żeby można było określić bliskość pomiędzy dwoma dokumentami.

Jest to o tyle ważne, że rozwiązanie może być tylko tak dobre, jak stworzony model problemu. W wypadku, gdy nie przechowuje on informacji np. o częściach dokumentów, takich jak tytuł czy nagłówki sekcji, to metoda grupowania będzie traktować wszystkie słowa w dokumencie jako równoważne — co wpłynie na jakość grupowania. Z drugiej strony przechowywanie tych informacji może dyskwalifikować metodę do innego źródła danych, w którym te informacje nie są dostępne lub w znaczny sposób wydłużyć czas działania algorytmu powodując, że stanie się on nieużyteczny dla dużych kolekcji dokumentów.

1.1.1. Model wektorowy

Najczęściej spotykanym modelem używanym do reprezentacji dokumentów tekstowych jest *model wektorowy* (MW, ang. *Vector Space Model*) [Wei06, Maz05, Koh00,

2. W istocie sprowadza się to do jednego problemu — las drzew można spiąć na szczycie hierarchii poprzez utworzenie grupy zawierającej wszystkie dokumenty z każdego korzenia drzewa w lesie.

Man01, Wu04, Ber03, For06, Man07]. Został on początkowo zaproponowany w dziedzinie wydobywania informacji (IR, ang. *Information Retrieval*) przez Gerarda Saltona [Sal75]. Każdy obiekt w tym modelu jest reprezentowany przez n–wymiarowy wektor cech $\vec{D} = [d_1, d_2, ..., d_n]$, gdzie d_i jest liczbą rzeczywistą.

W modelu tym poszczególne wymiary i odpowiadają kolejnym *terminom*[3], a wartości d_i określają jak dana *cecha* (współrzędna wektora) jest „ważna". Cechy są przeważnie wyliczane na podstawie częstości wystąpień i–tego terminu w dokumencie (opis niektórych schematów ważenia został przedstawiony w o punkcie 1.1.2).

Korpus (duża kolekcja dokumentów) często w MW jest sprowadzany do postaci macierzy dokumenty–terminy(ang. *term–document matrix*) [Wei06]. Dla korpusu o wielkości m możemy stworzyć macierz $M_{m \times n}$, której wiersze odpowiadają dokumentom, a kolumny przetwarzanym terminom.

Wybór modelu wektorowego wprowadza pewne ograniczenia, których należy być świadomym przed jego zastosowaniem. Tracone są wszelkie informacje na temat struktury dokumentów: tytuł, nagłówki, słowa kluczowe a nawet podział na zdania. Pomijane są informacje o kolejności słów (a co za tym idzie związki między nimi) — dokument jest traktowany jako *worek słów*(ang. *bag of words*) [Man01][4]. Podejście to zakłada, że wystąpienia terminów są niezależne od siebie.

Kolejnym istotnym ograniczeniem jest konieczność wyboru terminów dla których zostanie stworzona macierz — liczba wymiarów musi być znana z góry. Dlatego dodanie kolejnego dokumentu do kolekcji może być problematyczne w przypadku, gdy jego autor użył bogatszego słownictwa niż zaobserwowane w dokumentach wcześniej przetworzonych. W takiej sytuacji istnieją dwa wyjścia: albo nowe terminy zostaną pominięte, albo cała macierz zostanie przebudowana.

Pomimo tych ograniczeń model MW ma pewne zalety, które czynią go bardzo atrakcyjnym. W tej reprezentacji możliwe jest zastosowanie wielu istniejących technik sztucznej inteligencji (w tym grupowania) wprost do dziedziny dokumentów. Badania dowiodły, że niektóre relacje semantyczne mogą zostać wydobyte z tekstu z dużą dokładnością z pominięciem kolejności słów[Lan97, Kon06]. Również operacje (liczenie odległości, ważenie, itp.) na wektorach są łatwiejsze w rozumieniu i bardziej wydajne obliczeniowo od konkurencyjnych reprezentacji, np. opartych o model grafowy.

1.1.2. Schematy ważenia dla modelu wektorowego

Operowanie bezpośrednio na łańcuchach znakowych jest niewygodne i nieefektywne. Dlatego podczas wstępnego przetwarzania dokumentów tekstowych i budowy macierzy dokumenty–terminy zostają zebrane dane o częstości występowania terminów w dokumentach.

Ważenie jest procesem, który każdemu terminowi w dokumencie przypisuje wagę zależną od częstości wystąpień terminu w dokumencie [Man07]. Najprostszym sposobem ważenia macierzy jest przypisanie każdej współrzędnej wektora dokumentu d częstości występowania terminu t w dokumencie. Schemat ten jest określany w języku angielskim jako „term frequency" i oznacza się go jako $tf_{t,d}$[Man07].

3. Pod pojęciem *termin* rozumiane jest pojedyncze słowo (np. „izba") lub grupa słów, której znaczenie jest inne niż poszczególnych słów tworzących grupę (np. „Wysoka Izba").
4. Zwrot „bag of words" jest używany w znaczeniu wielozbióru słów, czyli zbioru, w którym jest dozwolone wielokrotne występowanie elementów[Man01].

Ta prosta idea ma poważną wadę — każdy termin w dokumencie jest uznawany za jednakowo ważny. Intuicja jednak mówi, że niektóre słowa lepiej określają treść dokumentu niż inne, są lepszymi dyskryminatorami. Słowa funkcyjne, np. „ten", „i", „lub" nie są w stanie rozróżnić dokumentów opisujących różne dziedziny. Podstawowym rozwiązaniem tego problemu jest przeskalowanie wartości $tf_{t,d}$ przez częstość globalną w całej kolekcji dokumentów, oznaczaną jako TF_t[Man07][5].

Lepszą informację od częstości w kolekcji dostarcza liczba dokumentów, w których dany termin występuje. Wielkość tą oznacza się jako df_t i nazywa częstością dokumentową (ang. *document frequency*). Wartości df_t są tym większe im termin t występuje w większej liczbie dokumentów — przez co jest mniej dyskryminatywny[6]. Dlatego wygodniej jest stosować odwrotną częstość dokumentową(idf, ang. *inversed document frequency*) definiowaną jako:

$$idf_t = \log \frac{N}{df_t}, \tag{1.1}$$

gdzie: N oznacza liczbę wszystkich dokumentów w korpusie (kolekcji dokumentów). Wartość idf_t jest wysoka dla terminów rzadkich, niska dla terminów częstych.

Łącząc opisane wyżej dwie wagi otrzymujemy definicję jednego z najbardziej popularnego schematu ważenia dokumentów w dziedzinie wydobywania informacji [Sal75, Man07]:

$$tf.idf_{t,d} = tf_{t,d} \times idf_t = tf_{t,d} \times \log \frac{N}{df_t} \tag{1.2}$$

Zastępując każdą komórkę w macierzy przez wartość $tf.idf_{t,d}$ otrzymamy wagi, które przyjmują (za [Man07]):

1. wartości maksymalne dla terminów występujących często w małej liczbie dokumentów, terminy te są dobrymi dyskryminatorami,
2. wartości niskie dla terminów występujących rzadko w małej liczbie dokumentów, lub występujących w dużej liczbie dokumentów, przez co terminy te mają małą siłę rozróżniającą dokumenty,
3. wartości minimalne dla terminów pojawiających się w (prawie) wszystkich dokumentach.

Inny schemat ważenia terminów w modelu wektorowym został zaprezentowany w pracy [Lan97]. Wywodzi się on z teorii informacji i psychologii poznawczej. Ważenie w tym schemacie następuje w następujących krokach.

1. Każda wartość wektora n wymiarowego wektora $\vec{D}$ jest poddawana skalowaniu logarytmicznemu $d_t = \ln(tf_{t,d} + 1)$. Skalowanie logarytmiczne przybliża krzywą uczenia.
2. Następnie zostaje wyliczona entropia Shannona dla dokumentu:

$$H_d = -\sum_{t=1}^{n} p_{t,d} \log p_{t,d}, \tag{1.3}$$

5. Manning et al. używają innego oznaczenia — cf od częstości kolekcji(ang. *collection frequency*).
6. Pod pojęciem dyskryminatywności terminu rozumiemy siłę z jaką dany termin jest w stanie odróżnić dokumenty od siebie.

$$\text{gdzie } p_{t,d} = \frac{tf_{t,d}}{\sum_{t=1}^{n} tf_{t,d}} \tag{1.4}$$

3. Każda wartość przeskalowanego logarytmicznie wektora jest dzielona przez entropię, tj. $d_t = \frac{tf_{t,d}}{H_d}$. Miara odwrotnej entropii jest użyta w celu oszacowania w jakim stopniu wystąpienie słowa definiuje cały dokument.

Schemat ten, ze względu na wykonywane operacje będzie dalej nazywany *logent*.

W pracy [Gol98] zostało przedstawione podejście do ważenia dokumentów opartego o teorię prawdopodobieństwa. Zdarzeniem losowym jest wystąpienie słowa w w dokumencie d_i pod warunkiem modelu M i korpusu D. Prostego oszacowanie prawdopodobieństwa wystąpienia słowa w można dokonać zgodnie z zasadą estymacji największej wiarygodności(MLE, ang. *Maximum Likelihood Estimation*):

$$P_{ML}(Y_i = w \| d, M) = \frac{tf_{w,d}}{\sum_{w \in d} tf_{w,d}} \tag{1.5}$$

Oszacowanie to jest niedokładne, ponieważ niektóre słowa nie niosą znaczenia i dokumenty mają różne długości. Dlatego Goldszmidt i Sahami rozważają zastosowania kilku technik wygładzania [Gol98, str. 3–4].

Przedstawione powyżej schematy ważenia nie są oczywiście jedynymi. Zostały one wybrane ze względu na popularność w dziedzinie, dobre wyniki empiryczne w zastosowaniach i jasno zdefiniowane fundamenty teoretyczne, na których się opierają.

1.1.3. Funkcje podobieństwa w modelu wektorowym

Po zdefiniowaniu sposobu reprezentacji dokumentu w modelu wektorowym należy zdefiniować pojęcie pozwalające na dokonywanie pomiarów bliskości pomiędzy dokumentami.

Ze względu na to, że dane są w postaci wektora, to można wyznaczać bliskość na podstawie odległości w przestrzeni metrycznej. Zazwyczaj mają one formę metryki Minkowskiego [For06]:

$$dist(d_i, d_k) = \left[\sum_{j=1}^{n} |d_{ij} - d_{kj}| \right]^{\frac{1}{r}}, \text{ dla } r \geqslant 1. \tag{1.6}$$

Najczęściej stosowane są metryki: miasto ($r = 1$, równanie 1.7) i euklidesowa ($r = 2$, równanie 1.8). Metryki te jednak nie sprawdzają się w dziedzinie dokumentów tekstowych, ponieważ często dokumenty mają różne długości[Man07].

$$dist_{r_1}(d_i, d_k) = \sum_{j=1}^{n} |d_{ij} - d_{kj}| \tag{1.7}$$

$$dist_{r_2}(d_i, d_k) = \sqrt{\sum_{j=1}^{n} (d_{ij} - d_{kj})^2} \tag{1.8}$$

Innym podejście do liczenia podobieństwa między dokumentami w MW polega na sprawdzeniu zgodności pomiędzy cechami wektora. Często używanymi miarami są: współczynnik *Dice* (równanie 1.9) i uogólniony współczynnik Jaccarda (równanie 1.10) [For06, Sch05].

$$dist_{Dice}(d_i, d_k) = 2\frac{\sum_{j=1}^{n} d_{ij}d_{kj}}{\sum_{j=1}^{n} d_{ij}^2 \sum_{j=1}^{n} d_{kj}^2} \tag{1.9}$$

$$dist_{Jacard}(d_i, d_k) = \frac{\sum_{j=1}^{n} d_{ij}d_{kj}}{\sum_{j=1}^{n} d_{ij}^2 + \sum_{j=1}^{n} d_{kj}^2 - \sum_{j=1}^{n} d_{ij}d_{kj}} \tag{1.10}$$

W dziedzinie grupowania dokumentów tekstowych najczęściej stosowaną miarą jest kosinus kąta (równanie 1.11) pomiędzy wektorami dokumentów [For06, Man07, Man01, Kon06, Sch05]. Jego podstawową zaletą jest to, że mierząc kąt ignorowana jest długość dokumentu. Metoda ta jest również wydajna w wypadku gdy wektory dokumentów są znormalizowane, nie trzeba obliczać pierwiastka iloczynu kwadratów w mianowniku równania 1.11.

$$dist_{cosine}(d_i, d_k) = \frac{\sum_{i=1}^{n} d_{ij}d_{kj}}{\sqrt{\sum_{i=1}^{n} d_{ij}^2 \sum_{i=1}^{n} d_{kj}^2}} \tag{1.11}$$

W wypadku gdy chcemy zdefiniować podobieństwo po transformacji macierzy do przestrzeni probabilistycznej (równanie 1.5) można użyć miary oszacowania pokrycia (ang. *expected overlap measure*) między dokumentami [Gol98]:

$$EO(d_i, d_j, D) = \sum_{w \in d_i \cap d_j} P(Y_i = w \| d_i, M) \cdot P(Y_j = w \| d_j, M) \tag{1.12}$$

Goldszmidt i Mosze przy wyprowadzaniu tego sposobu liczenia podobieństwa przyjęli następujące założenia: słowa występują zgodnie z rozkładem multinominalnym, występowanie słów dla dostępnej informacji dostarczanej w modelu M jest niezależne pomiędzy dokumentami i prawdopodobieństwo pozostaje niezależne w wypadku dostarczenia informacji, mówiącej że słowo w występuje w obu dokumentach.

Udowodniono [Gol98] jednak, że założenia te również są obecne w wypadku używania kosinusa kąta. W badaniach Goldszmidt i Mosze pokazali, że grupując przy użyciu szacowania pokrycia, generowany jest mniejszy błąd klasyfikacji niż dla kosinusa .

1.1.4. Model grafowy

Ograniczenia modelu wektorowego w metodach analizy skupień są powszechnie znane i zostało opracowanych wiele metod używających innego sposób reprezentacji dokumentów niż MW. Część z nich (np. algorytm najbliższego sąsiada, zob. punkt 1.2.4) przedstawia problem grupowania z punktu widzenia teorii grafów.

Można zaobserwować tutaj dwa główne podejścia: metody grupowania traktują obiekty jako węzły, a łuki grafu reprezentują powiązania między nimi. Techniki te mogą być z powodzeniem stosowane w dokumentach sieci Internet połączenia pomiędzy stronami WWW tworzą graf, który należy podzielić na rozłączne podgrafy zgodnie z przyjętą funkcją oceny. Rozszerzeniem tej reprezentacji jest przejście w dziedzinę hipergrafów — grafów, w których łuki mogą łączyć dowolną liczbę dokumentów [For06].

Drugie podejście wykorzystujące teorię grafów zostało zaproponowane w [Sch05]. Schenker et al. proponują, żeby modelować cały dokument jako graf połączeń między słowami i po wcześniejszym zdefiniowaniu wymaganych operacji na grafach stosować bezpośrednio klasyczne algorytmy analizy skupień w eksploracji danych. Podejście to zostanie szerzej opisane w punkcie 1.2.4.

1.1.5. Model przestrzeni metrycznej

W niektórych zastosowaniach algorytmów analizy skupień nie jest możliwe wykorzystanie modelu wektorowego czy grafowego, ponieważ danych nie da się zapisać w postaci numerycznej. W takich wypadkach jest jednak możliwe stworzenie formalnego modelu dla problemu, jeżeli jesteśmy w stanie zdefiniować funkcję podobieństwa pomiędzy obiektami. Używając takiej funkcji można zbudować kwadratową macierz podobieństwa i wykorzystać metody grupowania w tej przestrzeni danych[For06].

Przy założeniu, że wcześniej podane sposoby reprezentacji nie są właściwe dla dokumentów tekstowych, można stosować przestrzeń metryczną w tej dziedzinie. Budowa funkcji podobieństwa opartej na modelu grafowym, wektorowym lub innymi i przejście w przestrzeń podobieństwa pozwoli na spojrzenie na problem grupowania z innej perspektywy.

1.1.6. Redukcja wymiarów

Istotnym problemem eksploracji dużych zbiorów danych jest redukcja wymiarów, zarówno pod względem liczby obserwacji jak i ilości cech [Pal04]. Do tego problemu można podejść na dwa sposoby: poprzez selekcję cech (ang. *feature selection*) lub ekstrakcję cech (ang. *feature extraction*).

Selekcja cech w analizie skupień w przestrzeni dokumentów tekstowych polega na wyborze możliwie małego podzbioru jednostek leksykalnych, który da jak największą możliwość rozróżnienia dokumentów w korpusie. Innymi słowy należy odrzucić te terminy, które niosą najmniejszą ilość informacji o dokumencie.

Z drugiej strony ekstrakcja cech wykorzystuje całą dostępną informację w celu przejścia do nowej przestrzeni, o zredukowanej liczbie wymiarów[Pal04]. Transformacja ta może być liniowa lub nieliniowa. Przykładami transformacji liniowej jest analiza głównych składowych (PCA, ang. *Principal Components Analysis*) lub rozkład na wartości osobliwe(SVD, ang. *Singular Value Decomposition*). Odwzorowanie Sammona czy skalowanie wielowymiarowe (MDS, ang. *MultiDimensional Scaling*) są przykładami transformacji nieliniowych [Pal04, Ber03].

SVD było z powodzeniem stosowane w dziedzinie przetwarzania języka naturalnego. Początkowo zostało zastosowane do poprawienia jakości wyszukiwarek opartych o model wektorowy [Dee90]. Podejście to zostało wykorzystane jednak znacznie szerzej — do symulacji i wyjaśnienia niektórych funkcji poznawczych ludzkiego mózgu w technice zwanej analizą semantyki ukrytej(LSA, ang. *Latent Semantic Analysis*)[Lan97]. W początkowych eksperymentach z systemem WEBSOM również stosowano rozkład na wartości osobliwe.

Singular Value Decomposition

SVD jest techniką redukcji wymiarów macierzy[Ber96]. Dla macierzy wejściowej A o wymiarach t na d ($t >> d$) rozkład według wartości osobliwych można zdefiniować jako:

$$A_{txd} = T_{txn} S_{nxn} D^T_{dxn},$$

gdzie $n = \min(t, d)$. Macierze T i D są ortonormalne. Macierz S jest macierzą diagonalną, której przekątna zawiera osobliwe wartości macierzy A w kolejności malejącej.

Wiersze macierzy T odpowiadają wierszom macierzy A, kolumny macierzy T są wektorami własnymi (znormalizowane, wzajemnie prostopadłe). Dla macierzy D wiersze odpowiadają kolumnom macierzy A, a kolumny są wektorami własnymi.

Jeżeli usuniemy (zastąpimy zerami) k wartości własnych ($k < n$) z przekątnej macierzy S (zaczynając od wartości najmniejszych), to po przemnożeniu macierzy $T \cdot S \cdot D^T$ i odrzuceniu wierszy i kolumn z samymi zerami otrzymamy macierz zredukowaną, która dobrze przybliża macierz A w mniejszej ilości wymiarów.

SVD ma kilka istotnych ograniczeń. Głównym ograniczeniem jest to, że SVD rozpina nową przestrzeń w oparciu o wiersze macierzy wejściowej. Przez co początek nowego układu współrzędnych nie znajduje się w centrum rozmieszczenia danych w przestrzeni. Technika ta ma dużą złożoność obliczeniowa i pamięciową — nawet w implementacji wykorzystującej zaawansowane metody optymalizacji[Ber96].

Inne ograniczenia SVD i algorytmy, które ich nie posiadają są opisane w [Ber03, str. 103–120].

1.1.7. Segmentacja tematyczna dyskursu

Omawiając grupowanie w kolekcjach dokumentów tekstowych wg kryterium semantycznego należy przemyśleć problem spójności tematycznej w obrębie jednego dokumentu. W wypadku krótkich tekstów, takich jak hasła w encyklopedii czy streszczenia, teksty zazwyczaj są spójne. Jeżeli jednak naszym celem jest pogrupowanie dłuższych dokumentów — artykułów prasowych, książek, itp. — to warto zastanowić się nad zastosowaniem technik segmentacji tematycznej dyskursu.

Podział na *pod–tematy (podgrupy tematyczne)* można definiować na wiele sposobów. Tutaj jest on rozumiany zgodnie z intuicyjną definicją zaproponowaną w [Hea97, str. 39]:

> „The use of term subtopic here is meant to signify pieces of text „about" something and is not to be confused with topic (...) found within inividual sentences."
>
> (Użycie terminu *podgrupa tematyczna* ma na celu wskazanie fragmentów tekstu „mówiących o czymś" i nie powinno być mylone z z zakresem tematycznym pojedynczych zdań.)

Celem segmentacji tematycznej jest więc dokonanie gruboziarnistego (ang. *coarse–grained*) podziału na fragmenty traktujące w przybliżeniu o jednym zagadnieniu.

Podziału można dokonać hierarchicznie lub liniowo. W grupowaniu dokumentów w ujęciu tej pracy stosowanie technik liniowej segmentacji jest wystarczające, ponieważ za stworzenie powiązań pomiędzy segmentami jest odpowiedzialna metoda grupowania.

Podejście naiwne

Do segmentacji tematycznej tekstu można zastosować naiwną technikę polegającą na wykorzystaniu jedynie istniejących podziałów w tekście, takich jak akapity, tytuły sekcji, itp. Intuicja podpowiada, że zazwyczaj jeden temat omawiany jest w większym fragmencie tekstu niż jeden akapit. Dlatego warto wprowadzić do takiego podejścia parametr k pozwalający na kontrolę wielkości segmentu, np. minimalną liczbę zna-

ków, wyrazów czy zdań na akapit. Przykładowe działanie naiwnego podejścia zostało przedstawione na schemacie algorytmu 1.

Algorytm 1: Podejście naiwne do segmentacji tekstu.

Input: Tekst t, minimalna wielkość segmentu k
Result: Lista segmentów

```
1 buffor = UtworzBuffor;
2 repeat
3     buffor += WczytajNastępnyAkapit(t);
4     if wielkość wczytanego tekstu > k then
5         UtwórzSegment(buffor);
6         Wyczyść(buffor);
7     end
8 until nie koniec tekstu ;
```

Nie można oczekiwać, żeby wyniki takiej segmentacji były dokładne. Przy odpowiednio małych wartościach parametru kontrolującego wielkość segmentu, tekst zostanie podzielony na więcej podgrup tematycznych niż występujących w tekście, przy dużych wartościach k — niektóre podgrupy tematyczne zostaną scalone w jedną.

Z punktu widzenia grupowania, przypadek łączenia jest bardziej niekorzystny od braku podziału. Problem wielotematycznych dokumentów nie został rozwiązany, a na dodatek ilość szumu w danych się zwiększyła (ponieważ istnieje więcej punktów w przestrzeni mogących prowadzić do niewłaściwego połączenia grup). Zaskakująco — przypadek nadmiernego rozdrobnienia może mieć pozytywny wpływ na jakość grupowania — kilka segmentów traktujących o jednym temacie powinno zostać połączonych w jedną grupę. W tym wypadku jednak należy zdefiniować funkcję porównującą, która potrafi dobrze operować na dokumentach składających się z niewielkiej liczby słów, co nie jest prostym zadaniem.

Kolejnym ograniczeniem podejścia naiwnego jest fakt, że nie dla wszystkich tekstów istnieją naturalnie wyodrębnione podziały. Rozwiązanie naiwne ma następujące zalety: algorytm jest szybki, prosty w implementacji, niezależny od języka i może stanowić punkt odniesienia dla bardziej zaawansowanych metod segmentacji tekstu.

TextTiling

W wypadku, gdy naiwne podejście jest niewystarczające w danej aplikacji, należy skorzystać z bardziej wyrafinowanych algorytmów. Wyniki pionierskich badań w tej dziedzinie zostały opisane w pracach [Hea93, Hea97]. Zaproponowana tam technika została nazwana *TextTiling*. Algorytm ten działa na zasadzie przyznania *oceny leksykalnej* (ang. *lexical score*) każdej następującej po sobie parze akapitów w tekście. Po ocenie wszystkich podziałów dokonywana jest analiza wykresu ocen i na jej podstawie wyodrębniane są podgrupy tematyczne. Metoda ma kilka odmian — różnice dotyczą drugiego kroku na schemacie algorytmu 2 (sposobu wyliczania oceny leksykalnej).

Krok pierwszy TextTiling — *tokenizacja* — odnosi się tutaj do procesu podziału tekstu wejściowego na pojedyncze jednostki leksykalne. Każde słowo znajdujące się na

Algorytm 2: Schemat algorytmu TextTiling opartego na blokach tekstu.

Input: Tekst
Result: Lista segmentów
1 Tokenizacja;
2 Wyliczenie oceny leksykalnej;
3 Identyfikacja granic podgrup tematycznych;

stop–liście[7] zostaje odrzucone. Pozostałe słowa zostają sprowadzone do morfologicznej formy podstawowej (proces ten nazywany jest *lematyzacją*). W wyniku procesu tokenizacji tekst zostaje podzielony na sztuczne zdania o wielkości w. Hearst twierdzi, że dla prawdziwych zdań można otrzymać fałszywe wyniki porównania zdań o znacząco różnej długości.

Każde zdanie przedstawione jest w modelu wektorowym (zob. sekcja 1.1.1). Wartości wektora mogą być poddane transformacji $tf.idf$, chociaż często obniża ona skuteczność metody podczas dzielenia tekstów na podgrupy tematyczne. Wartości oceny leksykalnej (krok 2 algorytmu 2) można obliczyć na trzy sposoby: poprzez porównywanie *bloków* tekstu, analizę *wprowadzanego słownictwa* (ang. *vocabulary introduction*) i *łańcuchów leksykalnych* (ang. *lexical chains*).

Blok tekstu — zbiór k pseudo–zdań — jest abstrakcją akapitu. Również w tym wypadku wprowadzenie sztucznego bytu ma za zadanie wyeliminować wpływ długości akapitów występujących naturalnie w tekście na wyniki porównania. Każdemu przejściu i pomiędzy akapitami zostaje przypisana ocena zgodnie ze wzorem:

$$score(i) = \frac{\sum_t w_{t,b_1} w_{t,b_2}}{\sqrt{\sum_t w_{t,b_1}^2 \sum_T w_{t,b_2}^2}}, \tag{1.13}$$

gdzie: b_n to blok, $w_{t,b}$ to waga (wartość współrzędnej t wektora po transformacji) dla terminu t w bloku b.

Ocena leksykalną oparta na analizie wprowadzanego słownictwa polega na obserwacji pojawiających się nowych słów w interwałach czasowych. W metodzie opartej na łańcuchach leksykalnych rejestrowane są powtarzające się słowa pomiędzy zdaniami. Opis tych dwóch technik zaadoptowanych dla TextTiling znajduje się w [Hea97, p. 4.2, 4.3].

W ostatnim kroku TextTiling dokonuje podziału pomiędzy podgrupami tematycznymi. Ocena leksykalna jest nanoszona na wykres — oś OX oznacza numer przerwy między zdaniami, oś OY wartość oceny. Krzywe podobieństwa następnie zostają wygładzone. Na wykresie można zaobserwować szczyty oznaczające duże podobieństwo i doliny mówiące o zmianie tematu. Podział na podgrupy tematyczne odbywa się w miejscu występowania najgłębszych dolin.

LCSeg

W pracy [Gal03] autorzy zaproponowali podejście niezależne od języka operujące na łańcuchach leksykalnych i cechach językowych do segmentacji wieloosobowej kon-

7. Stop–lista to lista zawierająca terminy częste, nie niosące dużej ilości informacji, np. „oraz", „dobry".

wersacji. Zostało ono z powodzeniem zastosowane również do tekstów nie będących transkrypcją dialogów.

W pierwszej fazie algorytm dokonuje tokenizacji tekstu podobnie jak TextTiling (z tą różnicą, że zamiast lematyzacji dokonywany jest *stemming* — technika sprowadzająca wyrazy do rdzenia morfologicznego).

Po fazie wstępnego przetwarzania zostają utworzone łańcuchy leksykalne i wyliczona *ocena spójności leksykalnej* (ang. *lexical cohesion score*) na granicy zdań. Dla ważenia łańcuchów leksykalnych wykorzystano zmodyfikowaną wagę $tf.idf$ (tak, aby łańcuchy krótsze z większą ilością powtórzeń słów miały większą wartość). Ostatni krok oceny polega na obliczeniu kosinusa kąta pomiędzy dwoma blokami w tekście.

Podobnie jak w TextTiling algorytm sprawdza lokalne minima wykresu oceny. W LCSeg dla każdego lokalnego minimum zostaje przypisane prawdopodobieństwo wystąpienia podziału zgodnie ze wzorem:

$$p(m_i) = \frac{1}{2}[LCF(l) + LCF(r) - 2LCF(m)], \tag{1.14}$$

gdzie m_i to lokalne minimum, l i r to maksima na lewo i prawo od m_i, a $LCF(x)$ to wartość oceny spójności leksykalnej w punkcie x. Wartość $p(m_i)$ opisuje gwałtowność zmiany spójności, przy czym wartości większe (bliższe 1) oznaczają większą zmianę.

Z wykresu zostają odrzucone punktu których $p(m_i) < p_{limit}$. Dla pozostałych punktów obliczana jest średnia μ i standardowe odchylenie σ. Za miejsca podziału przyjmowane są punkty dla których spełniona jest zależność: $p(m_i) > \mu - \alpha \cdot \sigma$. W przytoczonych zależnościach p_{limit} i α są parametrami użytkownika.

Inne podejścia

Opisane powyżej podejścia do segmentacji tematycznej dyskursu nie są jedynymi. Ponte i Croft proponują technikę wykorzystującą rozwijanie zapytań(ang. *query expansion*)[Pon97]. Podejście oparte na budowie statystycznego modelu językowego zostało zaproponowane w pracy[Bee99]. Choi et al. wykorzystali analizę semantyki ukrytej do wykrywania podziału między podgrupami tematycznymi[Cho01].

1.2. Przegląd metod grupowania

Metody grupowania zostały zebrane w kilka kategorii pod względem ogólnego mechanizmu działania. Są to:

- metody płaskie — dzielą zbiór danych na części bez wskazywania połączeń pomiędzy grupami,
- metody hierarchiczne — tworzą hierarchię grup,
- metody oparte na gęstości — definiują problem grupowania używając pojęcia gęstości,
- metody grafowe — wykorzystują teorię grafów do budowy grup,
- metody bazujące na naturze — opierają się na mechanizmach, które można zaobserwować w naturze,
- inne — niepasujące do żadnej z powyższych kategorii.

Często wyodrębnia się inny podział: na *metody twarde* (ang. *crisp, hard*), które charakteryzują się tym, że jeden obiekt może należeć tylko do jednej grupy i *miękkie* (ang. *fuzzy, soft*), pozwalające na przynależność jednego obiektu do wielu grup.

1.2.1. Metody płaskie

Płaskie metody grupowania dzielą zbiór danych na części bez wskazywania połączeń pomiędzy grupami. Z punktu widzenia celu grupowania w niniejszej pracy — budowy szkieletu dla tezaurusa hierarchicznego — nie są one przydatne. Stanowią jednak dobre wprowadzenie do innych, bardziej zaawansowanych technik, jednocześnie dając znakomity punkt odniesienia do badań pod względem jakości i wydajności.

Klasycznymi płaskimi metodami grupowania są: k–średnich(ang. *k–means*) i k–mediana(ang. *k–means*). Działanie metody k–średnich zostało przedstawiona na schemacie algorytmu 3. Centroid to obiekt będący średnią z elementów należących do grupy — najczęściej jest to obiekt tworzony sztucznie. Standardowo jako funkcji odległości używa się odległości euklidesowej lub $L_1 = \Sigma_l |x_l - y_l|$. W modelu wektorowym lepiej jest jednak użyć odległości bazującej na kosinusie kąta pomiędzy wektorami[For06, Man01].

Algorytm 3: Schemat algorytmu k–średnich

Input: liczba grup k, zbiór dokumentów
Result: grupy
1 Wylosuj k dokumentów, które będą początkowymi centroidami;
2 **repeat**
3 Oblicz centroidy dla grup;
4 Przypisz dokumenty do najbliższych centroidów;
5 **until** *chociaż jeden centroid się zmienił* ;

Jako wejście algorytm przyjmuje liczbę dodatnią k, która określa pożądaną liczbę grup. W pierwszym kroku działania należy wylosować k dokumentów. Będą one początkowymi centroidami. Następnie do każdego centroidu zostają przypisane wszystkie najbliższe mu dokumenty, tworząc grupę. Dla każdej grupy można wyznaczyć punkt środkowy — w ten sposób powstaje nowy centroid. Uaktualnianie środków grup i przypisywanie do nich nowych dokumentów jest powtarzane dopóki chociaż jeden centroid został zmieniony.

Metoda k–średnich wprowadza szereg problemów, które należy rozwiązać.

- W jaki sposób traktować remisy (sytuacje, gdy dokumenty są w takiej samej odległości od więcej niż jednego środka grupy)?
- Należy z góry wiedzieć ile chcemy mieć grup. Często ta informacja nie jest dostępna, szczególnie w dziedzinie dokumentów tekstowych.
- Losowy wybór punktów początkowych może prowadzić do złych wyników.
- Metoda często znajduje tylko ekstrema lokalne.
- Metoda potrafi wykryć jedynie grupy o sferycznych kształtach.

- Metoda nie jest odporna na szum i punkty odstające[8].

Zaletami metody są: prostota i złożoność obliczeniowa $O(n)$ [Kon06, Man01]. Pomimo swoich wad metoda jest często stosowana [Sch05, Man01, Kor05, Kon06]. Istnieje kilka algorytmów, które rozszerzają oryginalną idee k–średnich. Np. w algorytmie globalnym (ang. *Global K–Means*) został wprowadzony deterministyczny wybór początkowych centroidów [Lik01].

Centroid jest obiektem sztucznym, szczególnie jest to nienaturalne dla dokumentów tekstowych — środek grupy, dokument będący średnią z innych dokumentów, nie istnieje. Dlatego zostaje wprowadzone pojęcie mediany(ang. *medoid*) [Maz05]. Jest to obiekt(dokument) znajdujący się najbliżej środka grupy. Jednym z przykładowych algorytmów opartych na idei k–median jest PAM przedstawiony na schemacie algorytmu 4

Algorytm 4: Schemat algorytmu PAM(ang. *Partition Around Medoids*) na podstawie [Ng94, Maz05]

Input: liczba grup k, zbiór dokumentów
Result: grupy

```
1 Wylosuj k dokumentów, które będą początkowymi medianami;
2 D_s = ∞;
3 repeat
4     Przypisz dokumenty do najbliższych median;
5     Dla każdej możliwej zmiany mediany oblicz nową wartość funkcji kosztu D_n;
6     if ∃D_n < D_s then
7         D_s = min_n D_n;
8     end
9 until D_s uległo zmianie ;
```

Poza bardziej naturalną reprezentacją mediana powoduje, że algorytm jest bardziej odporny na szum, ale jest jednocześnie bardziej kosztowny obliczeniowo[For06]. Kolejność danych nie ma wpływu na wyniki metody[Ng94].

Istnieje cała rodzina metod opartych na idei k–średnich. CLARA (ang. *Clustering LARge Applications*) nie szuka median dla całego zbioru danych, tylko stosuje PAM do losowo wybranej próbki danych. Motywowane jest to tym, że jeżeli próbka jest dobrze wybrana, to mediany dla niej będą przybliżać mediany dla całego zbioru. W celu poprawienia jakości grupowania, losowanie danych jest powtarzane kilka razy i wybierane jest najlepsze rozwiązanie. CLARANS(ang. *Clustering Large Applications based on RANdomized Search*)[Ng94] rozszerza ideę CLARA o ograniczenie zamiany obiektów z medianami jedynie do najbliższych sąsiadów.

8. Ng i Han definiują punkt odstający (ang. *outlier*) jako „punkt danych bardzo odstający od pozostałych" (ang. *data points that are very far away from the rest of the data points*) [Ng94]. Szum natomiast jest rozumiany jako losowość dystrybucji słów [For06] lub jako cecha, która zwiększa poziom błędu po uwzględnieniu jej w modelu [Man07].

1.2.2. Metody hierarchiczne

Zasada stojąca ze metodami hierarchicznymi jest prosta: dla danej funkcji celu[9] wszystkie obiekty mogą zostać umieszczone w sposób deterministyczny w węzłach binarnego drzewa, które wskaże całkowite uporządkowanie obiektów. W liściach znajdują się pojedyncze obiekty. Drzewo takie jest nazywane dendrogramem.[For06].

Drzewo grup może być stworzone na dwa sposoby: od dołu(ang. *bottom–up*) poprzez rozpoczęcie z grupami zawierającymi pojedyncze obiekty, w każdym kroku algorytm łączy najbardziej bliskie grupy. Podejście od góry(ang. *top–down*) startuje z jedną grupą zawierającą wszystkie obiekty, która jest dzielona w taki sposób, żeby maksymalizować podobieństwo wewnątrz grup.

Metoda od dołu nazwana jest *aglomeracyjną* (HAC, ang. *Hierarchical Agglomerative Clustering*). Drugie podejście nazywane jest *deglomeracyjnym* lub *dzielącym* [Man01, Man07].

W porównaniu z metodami płaskimi, metody hierarchiczne generują grupy o lepszej jakości, dostarczają większej ilości informacji, jednak są bardziej złożone obliczeniowo [Man01].

Aglomeracyjne

Aglomeracyjne grupowanie hierarchiczne jest jedną z najdokładniej zbadanych metod grupowania[For06]. Manning et al. opisują trzy główne kryteria łączenia obiektów w grupy podczas budowania dendrogramu. Są to podejścia oparte na pojedynczym połączeniu (ang. *Single–Linkage*), całkowitym połączeniu (ang. *Complete–Linkage*) i połączeniu średnio–grupowym (ang. *Group–Average*). Ogólny strategia działania HAC została przedstawiona na schemacie algorytmu 5.

Algorytm 5: Schemat aglomeracyjnego algorytmu hierarchicznego

Input: zbiór dokumentów, funkcja odległości pomiędzy grupami
Result: grupy

```
1 Dla każdego dokumentu utwórz grupę;
2 repeat
3     ⟨p, q⟩ = NajbliższaParaGrup();
4     Połącz⟨p, q⟩;
5 until uzyskano grupę zawierającą wszystkie elementy lub osiągnięto inne
  kryterium stopu ;
```

W metodzie pojedynczego połączenia grupy są łączone na podstawie podobieństwa wyliczonego pomiędzy dwoma najbliższymi obiektami z grup. Takie podejście gwarantuje dużą lokalną spójność, jednak w efekcie ma tendencje do generowania podłużnych grup (ang. *elongated clusters*). Single–link jest metodą pokrewną do znajdowania minimalnego drzewa rozpinającego (MST, ang. *Minimum Spanning Tree*) zbioru punktów.

Metoda całkowitego połączenia łączy grupy w oparciu o podobieństwo wyliczone pomiędzy dwoma najdalszymi obiektami w grupach — skupiając się na globalnej jakości grup. Dzięki temu zostaje rozwiązany problem nadmiernie wydłużonych grup — zostają wygenerowane spójne, zwarte grupy. Podejście to jednak jest bardzo kosztowne

9. Funkcja celu mówi o jakości dokonywanego podziału.

obliczeniowo — jego złożoność to $O(n^3)$ [Man01]. Zaletą algorytmu jest duża odporność w obliczu szumu[For06].

Połączenie średnio–grupowe wylicza średnią odległość pomiędzy grupami. To podejście jest traktowane jako kompromis pomiędzy single–link a complete–link. Z jednej strony unika podłużnych grup generowanych przy łączeniu w oparciu o pojedyncze połączenie, z drugiej — przy złożoności obliczeniowej rzędu $O(n^2)$ — jest bardziej wydajne, niż przy liczeniu całkowitego połączenia [Man01].

Istnieje wiele metod algorytmów hierarchicznych, które są bardziej wydajne od podejść klasycznych. Ciekawy pomysł został zaprezentowany w pracy [Zha96]. Algorytm BIRCH (ang. *Balanced Iterative Reducing and Clustering using Hierarchies*) został zaprojektowany do pracy z dużymi zbiorami danych, które nie mieszczą się w pamięci operacyjnej. Metoda ta podczas jednokrotnego przeglądu danych buduje zrównoważone drzewo cech podsumowujących grupy[10] (CF Tree, ang. *Cluster Feature Tree*).

ROCK

Bardzo popularnymi[11] algorytmami hierarchicznymi są CURE (ang. *Clustering Using REpresentatives*) [Guh98] i ROCK (ang. *RObust Clustering using linKs*) [Guh00] zaproponowane przez Guha et al. CURE bazuje na zasadzie najbliższego sąsiada, ale używa reprezentantów zamiast wszystkich obiektów w grupie. Reprezentanci tworzą zbiór o ustalonej liczności, który budowany jest wg następującej zasady: pierwszy znajduje się w największej odległości od środka grupy, drugi najdalej od pierwszego, itd [For06]. Zostało pokazane, że CURE potrafi wykryć grupy o nietypowych kształtach, które nie są odnajdowane przez inne metody.

Zarówno ROCK jak i CURE mają podobny schemat działania. ROCK jednak został zaprojektowany z myślą o danych nienumerycznych. Dlatego nie używa reprezentantów do grupowania, tylko wprowadza pojęcie połączenia(ang. *link*). Metoda ta zostanie omówiona dokładniej. Aby to zrobić należy wprowadzić kilka pojęć.

Sąsiedztwem punktu p jest taki zbiór punktów, który jest do p podobny. Niech $sim(p_i, p_j)$ będzie *funkcją podobieństwa*, która jest znormalizowana, mówi o bliskości punktów p_i i p_j i przyjmuje wartości od 0 do 1 (większe wartości oznaczają większe podobieństwo). Dla danego progu $\theta \in [0, 1]$ punkty p_i i p_j są sąsiadami wtedy i tylko wtedy, gdy:

$$sim(p_i, p_j) > \theta \tag{1.15}$$

Większe wartości parametru θ nakładają silniejsze ograniczenie na zbiory sąsiedztwa. Maksimum jest osiągane, gdy $\theta = 1$ — sąsiedztwo jest ograniczone do punktów identycznych. Dla wartości $\theta = 0$, każda para punktów jest uznawana za sąsiadów.

Guha et al. wnioskują, że dwa punkty mogą być do siebie podobne, jednak należeć do różnych klas w naturalnie stworzonych grupach. Dlatego operowanie jedynie na bliskości pomiędzy punktami w grupowaniu nie dostarcza wystarczającej informacji w wypadku, gdy grupy nie są dobrze oddzielone. W takiej sytuacji, pomimo podobieństwa pary punktów jest mało prawdopodobnym, żeby punkty te miały dużą liczbę wspólnych sąsiadów. Obserwacja ta prowadzi do wyprowadzenia pojęcia połączenia[Guh00].

10. Cecha jest trójką $(N, \vec{LS}, SS)$, gdzie N to liczba punktów w grupie, $\vec{LS}$ jest sumą wszystkich punktów a SS jest sumą kwadratów punktów, czyli $SS = \sum_{i=1}^{N} \vec{X}^2$.
11. Wyszukiwarka Google Scholar znajduje ponad 1200 artykułów które cytują prace Guha et al.

Połączeniem pomiędzy punktami p_i i p_j jest liczba mówiąca o ilości wspólnych sąsiadów jakie mają punkty p_i i p_j. Liczba ta jest oznaczana jako $link(p_i, p_j)$. Inaczej mówiąc $link(p_i, p_j)$ jest liczbą ścieżek o długości 2 w nieskierowanym grafie sąsiedztwa.

Funkcja celu E_l dla metody ROCK ma za zadanie zmaksymalizować liczbę połączeń w jednej grupie jednocześnie nie dopuszczając rozwiązania, które będzie nadmiernie łączyć grupy:

$$E_l = \sum_{i=1}^{k} n_i * \sum_{p_q, p_r \in C_i} \frac{link(p_q, p_r)}{n_i^{1+2f(\theta)}} \tag{1.16}$$

Wartość tej funkcji należy zmaksymalizować dla k grup. W równaniu 1.16 C_i oznacza i–tą grupę o wielkości n_i. Wartość $n_i^{1+2f(\theta)}$ jest oszacowaniem liczby połączeń dla grupy o wielkości C_i. Funkcja $f(\theta)$ jest zależna od zbioru danych i od rodzaju grup jakie metoda ma znaleźć. Jest ona jedynie oszacowaniem — i nawet dla bardzo niedokładnego wyboru $f(\theta)$ ROCK znajduje dobre grupy, ponieważ błąd wpływa na wszystkie grupy, a nie tylko na ich część. Przykładem może być funkcja o postaci $f(\theta) = \frac{1-\theta}{1+\theta}$, która została zastosowana z powodzeniem w dziedzinie przetwarzania języka naturalnego[Son06].

Aby zmaksymalizować funkcję celu E_l można wprowadzić funkcję oceniającą trafność scalenia dwóch grup:

$$g(C_i, C_j) = \frac{link\,[C_i, C_j]}{(n_i + n_j)^{1+2f(\theta)} - n_i^{1+2f(\theta)} - n_j^{1+2f(\theta)}}, \text{ gdzie:} \tag{1.17}$$

$$link\,[C_i, C_j] = \sum_{p_q \in C_i, p_r \in C_j} link(p_q, p_r) \tag{1.18}$$

Równanie 1.18 mówi o liczbie połączeń pomiędzy dwoma grupami C_i i C_j. Funkcja $g(C_i, C_j)$ z równania 1.17 ocenia trafność decyzji o scaleniu tych grup.

Działanie algorytmu ROCK jest przedstawione na schemacie algorytmu 6.

Algorytm 6: Schemat algorytmu ROCK

Input: zbiór dokumentów, funkcja podobieństwa, pożądana liczba grup
Result: grupy

1 Wybór losowej próbki danych;
2 links = CreateLinks();
3 ROCK(links);
4 Klasyfikacja pozostałych danych.

Algorytm zaczyna od wylosowania próbki danych. Próbka z jednej strony powinna być reprezentatywna dla zbioru danych i jak największa. Z drugiej — powinna się zmieścić w całości w pamięci razem z innymi potrzebnymi procedurami. Drugim krokiem algorytmu jest stworzenie macierzy połączeń pomiędzy dokumentami. W podejściu naiwnym jest to problem podobny do mnożenia macierzy, ma on złożoność $O(n^3)$. Dlatego w [Guh00] proponują bardziej wydajny sposób liczenia połączeń(zob. schemat algorytmu 7). Idea jest prosta: po wyliczeniu sąsiedztwa dla każdego punktu i, każda para w liście sąsiadów punktu i tworzy jedno połączenie (i–ty punkt łączy parę

Algorytm 7: CreateLinks() — funkcja obliczająca w wydajny sposób połączenia między dokumentami

Input: zbiór dokumentów D
Result: grupy

```
1  Oblicz lista_sasiedztwa[i] dla każdego punktu w D;
2  Wyzeruj link[i, j] dla każdego i,j;
3  for i = 1 to n do
4      N = lista_sasiedztwa[i];
5      for j = 1 to ||N|| − 1 do
6          for l = j + 1 to ||N|| do
7              link[N[j], N[l] ] = link[N[j],N[l]] + 1;
8          end
9      end
10 end
```

sąsiadów). Złożoność tego algorytmu wynosi $O(n^2 m_a)$, dla średniej liczby sąsiadów m_a(dowód został przedstawiony w [Guh00, str. 356]).

Algorytm 8 przedstawia grupowanie w ROCK. Na początku zostają zbudowane *lokalne kopce* (ang. *local heaps*) $q[i]$ dla każdej grupy i, czyli struktury zawierające wartości funkcji g (równanie 1.17) i numery grup, z którymi i–te skupienie jest połączone. Elementy lokalnego kopca są posortowane malejąco zgodnie z funkcją g. Dzięki temu pierwsza grupa kopca $q[i]$ oznacza najlepszego kandydata do scalenia z grupą i.

Następnie zostaje zbudowany *globalny kopiec* (ang. *global heap*) Q, czyli struktura która przechowuje wszystkie grupy posortowane malejąco zgodnie z wartością funkcji g pierwszego elementu lokalnego kopca dla danej grupy. Dzięki temu pierwszy element kopca Q zawiera informacje o grupie, która będzie w następnym kroku algorytmu łączona.

W pętli while, rozpoczynającej sie w kroku 3 algorytmu 8, zostają wyodrębnione grupy. Działa ona dopóki na kopcu nie pozostanie wymagana liczba grup, lub gdy liczba połączeń pomiędzy wszystkimi grupami spadnie do zera. W każdej iteracji dwie najlepsze grupy zostają scalone (4–7), a następnie ich lokalne kopce zostają uaktualnione, zostaje stworzony kopiec lokalny dla nowopowstałej grupy. Te czynności odbywają się w pętli for (8–13). W kolejnym kroku globalny kopiec Q zostaje odświeżony, a pamięć zajmowana przez już niepotrzebne kopce lokalne zostaje zwolniona.

Ostatnim krokiem ROCK jest oznaczenie pozostałych danych, spoza próbki losowej. Odbywa się to w dwóch krokach: najpierw zostaje wybrany zbiór L_i zawierający ułamek punktów dla każdej grupy i. Następnie pozostałe dane są wczytane z dysku i każdy punkt p jest przypisywany do grupy i, tak że p ma największą liczbę sąsiadów z i.

Metody hierarchiczne dzielące

Dzielące metody hierarchiczne(HDC, ang. *Hierarchical Devisiv Clustering*) rozpoczynają działanie traktując wejściowy zbiór danych jako jedną grupę. Algorytmy te charakteryzuje duża złożoność pamięciowa i obliczeniowa. W [Maz05] przedstawiono ogólny schemat działania algorytmów grupowania z klasy HDC (zob. algorytm 9).

Jako schemat działania dla metod HDC można użyć jednego ze schematów opi-

Algorytm 8: Hierarchiczne grupowanie w ROCK (punkt 3 algorytmu 6).

Input: zbiór dokumentów D, zbiór połączeń links, pożądana liczba grup k
Result: grupy

```
1  Dla każdego dokumentu d zbuduj lokalny kopiec q[d];
2  Zbuduj globalny kopiec Q;
3  while ||Q|| > k do
4      u = max(Q);
5      v = max(q[u]);
6      usuń(Q, v);
7      m = scal(u, v);
8      for x ∈ q[u] ∪ q[v] do
9          link[x, w] = link[x, u] + link[x, v];
10         usuń(q[x], u); usuń(q[x], v);
11         wstaw(q[x], w); wstaw(q[w], x);
12         uaktualnij(Q, x, q[x]);
13     end
14     wstaw(Q, w);
15     zwolnij(q[u]);zwolnij(q[v]);
16 end
```

Algorytm 9: Ogólny schemat działania algorytmów z klasy HDC wg [Maz05]

```
1  Stwórz jedną grupę zawierającą wszystkie dokumenty;
2  Wygeneruj wszystkie możliwe podziały bieżących grup;
3  Zgodnie z funkcją oceny wybierz najlepszy podział;
4  if Każdy obiekt znajduje się w innej grupie then
5      STOP;
6  else
7      GOTO 2;
```

sanych dla metod HAC(single–link, complete–link, group–average). Metody te nie są jednak często stosowane[Maz05, Man01]. Forster w pracy [For06, str. 40] zauważa jednak, że w ostatnim czasie coraz więcej badań jest prowadzonych nad nimi.

1.2.3. Metody oparte na gęstości

Interesujące podejście do grupowania zostało przedstawione w pracy [Est96]. Autorzy patrzą na problem grupowania z zupełnie innej perspektywy — opierają definicję grupy na pojęciu gęstości w przestrzeni.

Aby zrozumieć ideę działania zaproponowanego algorytmu o nazwie DBSCAN należy przytoczyć kilka definicji. Pod pojęciem ϵ*–sąsiedztwa* punktu p jest rozumiany taki zbiór punktów, do których odległość z p jest nie większa od ustalonego progu ϵ.

Punkty tworzące jedną grupę można podzielić na dwie klasy: *punktów rdzenia* (ang. *core points*) i *punktów granicznych* (ang. *border points*). Obserwacja ta prowadzi do wprowadzenia trzech pojęć: *bezpośredniej osiągalności gęstościowej* (ang. *directly density–reachable*), *osiągalności gęstościowej* (ang. *density–reachable*) i *połączenia gęstościowego* (ang. *density-connected*).

Punkt p jest bezpośrednio osiągalny gęstościowo z punktu q, gdy p znajduje się w ϵ–sąsiedztwie punktu q i q jest punktem rdzeniowym, czyli jego ϵ–sąsiedztwo zawiera co najmniej $MinPts$ punktów. Natomiast punkt p jest osiągalny gęstościowo z punktu q, gdy istnieje taki łańcuch punktów $p_1, \ldots, p_n$, gdzie $p_1 = p$ a $p_n = q$, że para sąsiednich punktów jest bezpośrednio osiągalna gęstościowo. Punkt p jest połączony gęstościowo z punktem q, wtedy i tylko wtedy gdy istnieje taki punkt o, że zarówno p jak i q są osiągalne gęstościowo z o.

Po wprowadzeniu powyższych definicji można określić czym jest grupa w notacji gęstości. Grupa C jest niepustym zbiorem punktów, takich że dla każdej pary punktów p i q zachodzą dwa warunki: oba punkty są połączone gęstościowo i jeżeli $p \in C$ i p jest osiągalny gęstościowo z punktu q, to $q \in C$.

Algorytm 10: Ogólny schemat DBSCAN

Input: zbiór punktów, ϵ, $MinPts$
Result: grupy

```
1 while Istniej jakiś punkt nierozpatrzony do
2     Wybierz dowolny nierozpatrzony punkt p;
3     Q ← Wszystkie punkty osiągalne gęstościowo z p;
4     if card(Q) = 0 then
5         Punkt p jest punktem granicznym;
6     else
7         Stwórz nową grupę;
8 end
```

Idea działania DBSCAN została przedstawiona na schemacie algorytmu 10. Algorytm ten jest wydajny dla dużych baz danych. Jego czas działania to $O(n \log n)$. Potrafi wykryć grupy o dowolnych kształtach w przeciwieństwie do metod klasycznych, które wykrywają tylko grupy wypukłe.

Istnieje cała rodzina algorytmów rozwijających ideę zaproponowaną w [Est96], np: GDBSCAN (ang. *Generalized DBSCAN*) [San98] uogólnia algorytm DBSCAN o m.in. stosowanie innych atrybutów niż przestrzenne. DBCLASD (ang. *Distribution Based Clustering of LArge Spatial Databases*) [Xu98] nie wymaga podawania żadnych parametrów. OPTICS (ang. *Ordering Points to Identify the Clustering Structure*) [Ank99] nie generuje bezpośrednio grup, tylko wskazuje uporządkowanie danych w bazie wg całego zbioru globalnych parametrów.

Algorytmy oparte na gęstości są rzadko stosowane w dziedzinie dokumentów tekstowych [For06]. Wyniki eksperymentów omówione w [Dee06] sugerują, że dokumenty naturalnie tworzą grupy wypukłe. Zostały one przeprowadzone jednak tylko na jednym korpusie tekstów dla języka angielskiego.

1.2.4. Metody grafowe

Schenker et al. sugerują, żeby modelować cały dokument jako graf połączeń między słowami. Proponują oni następujące sposoby reprezentacji dokumentu[Sch05, str. 35–37]:

- *Reprezentacja standardowa* (ang. *standard representation*) — dla każdego słowa jest tworzony węzeł, przy czym jedno słowo występuje tylko raz w grafie dla dokumentu. Dokument jest podzielony na sekcje: tytuł (wraz z metadanymi), odnośniki (tekst w odnośnikach), tekst (cały widoczny tekst, włącznie z odnośnikami). Jeżeli dwa słowa występują bezpośrednio po sobie w obrębie jednej sekcji to jest tworzony łuk skierowany od pierwszego do drugiego z nich. Łuk jest oznaczony zgodnie z miejscem występowania jako tytuł(TI), powiązanie(L) lub tekst(TX) . Po zbudowaniu grafu przeprowadzany jest *stemming* (technika polegająca na sprowadzeniu słów do rdzenia morfologicznego) i węzły są zwijane do najczęściej występującej formy.
- *Reprezentacja prosta* (ang. *simple representation*) — analogiczna do reprezentacji standardowej z tą różnicą, że przetwarzany jest tylko tekst widoczny na stronie i do łuków nie są przypisywane etykiety.
- *Reprezentacja n–odległości* (ang. *n–distance representation*) — łuki grafu są tworzone nie tylko dla słów występujących bezpośrednio po sobie, ale również dla n słów do przodu (n jest parametrem dostarczanym przez użytkownika). Połączenie między słowami jest tworzone tylko, gdy nie zostaną napotkane predefiniowane znaki interpunkcyjne. Łuk jest etykietowany odległością pomiędzy słowami.
- *Reprezentacja prostej n–odległości* (ang. *n–simple distance*) — analogiczna reprezentacja do n–odległości, z tą różnicą, że łuki nie są etykietowane odległością. Graf mówi tylko o tym, że pomiędzy słowami występuje połączenie, ale nie mówi jak jest ono silne.
- *Reprezentacja bezwzględnej częstości* (ang. *absolute frequency*) — podobna do reprezentacji prostej — węzły są tworzone dla słów występujących bezpośrednio po sobie, nie są uwzględniane informacje strukturalne. Do węzła przypisywana jest ilość wystąpień słowa w dokumencie, do łuku — częstość wystąpienia dwóch słów po sobie.
- *Reprezentacja względnej częstości* (ang. *relative frequency*) — analogicznie do reprezentacji bezwzględnej częstości, przy czym liczności wystąpień słowa (etykiety węzłów) są normalizowane przez maksimum z częstości wszystkich węzłów, a licz-

ności powiązań między słowami (etykiety łuków) przez maksimum liczności wszystkich powiązań.

W tej reprezentacji należy określić czym jest podobieństwo pomiędzy dokumentami. Schenker et al. definiują kilka miar, m.in. opartą na maksymalnym wspólnym podgrafie, odległości edycyjnej (ile operacji należy wykonać aby przekształcić jeden graf w drugi), itp.

Po zdefiniowaniu sposobu reprezentacji dokumentów i funkcji podobieństwa pomiędzy nimi można wykorzystać standardowe metody grupowania. W pracy [Sch05] zostają dokładnie zbadane: algorytmy podziałowe k–średnich i rozmytego c–średnich[12] oraz metody aglomeracyjnego hierarchicznego grupowania zgodnie ze schematami single–link i complete–link.

Dla algorytmu k–średnich na jednym testowanym zbiorze danych podejście grafowe dało dużo (rzędu 10%) lepsze rezultaty, na innym — podejście klasyczne okazało się nieznacznie (o ok. 1%) lepsze. Klasyczna metoda HAC pojedynczego połączenia okazała się lepsza dla testowanych kolekcji w [Sch05] — jednak dla obu sposobów reprezentacji była znacząco gorsza od innych badanych metod grupowania. Dla pozostałych algorytmów grupowania podejście grafowe generowało grupy o wiele bardziej odpowiadające klasyfikacji dokonanej przez ludzi.

Lepsze wyniki grupowania bazującego na reprezentacji dokumentu jako grafu można wytłumaczyć tym, że ten model zachowuje dużo więcej informacji o tekście (kolejność słów, struktura tekstu, częstości słów), niż model wektorowy (jedynie częstości wystąpienia słów).

Wyznaczanie podobieństwa pomiędzy grafami jest kosztowne obliczeniowo. Wydawać by się mogło, że czas wykonania będzie głównym ograniczeniem tej techniki. Tak jest rzeczywiście — w wypadku algorytmu k–średnich czas grupowania grafów dla 2340 dokumentów to ponad 550 minut[13]. Metoda klasyczna ukończyła to samo zadanie w niecałe 215 minut.

W wypadku gdy zostanie ograniczona liczba węzłów w grafie do 10 (dla 10 najczęstszych słów na dokument), to czas działania algorytmu spadł do 173 minut, przy czym jakość grupowania pogorszyła się jedynie nieznacznie [Sch05, str. 82]. Bardziej imponujący wynik metody grafowej — 4.5 godziny krótszy czas wykonania w stosunku do metody w modelu wektorowym —jest w wypadku zastosowania rozwinięcia globalnego algorytmu k–średnich.

1.2.5. Metody bazujące na naturze

W literaturze można spotkać sposoby naśladujące naturalne procesy w celu grupowania [Kon06]. Przy użyciu algorytmów genetyczne można przeszukać przestrzeń możliwych rozwiązań (wszystkich istniejących podziałów na grupy) i w sensownym czasie uzyskać sub–optymalne rozwiązanie. Symulowane wyżarzanie poprzez naśladowanie procesu z dziedziny metalurgii stara się uniknąć zatrzymania przeszukiwania w lokalnym optimum. Sztuczne sieci neuronowe pozwalają na wykrycie skomplikowanych,

12. Algorytm podobny w idei działania do k–średnich, z tą różnicą, że jeden obiekt może należeć do kilku grup.
13. Testy czasowe zostały wykonane na komputerze Sun UltraSPARC–II o mocy obliczeniowej 296 MHz i 1024 MB pamięci operacyjnej.

nawet hierarchicznych, wzorców w zbiorze danych — ze względu na tą własność jedna rodzina sieci zostanie omówiona szerzej.

Sieci Kohonena

Samoorganizująca się sieć odwzorowań Kohonena (SOM, ang. *Self-Organizing Map*) (czasami nazywaną też *mapą Kohonena*) została opracowana przez Teuvo Kohonena na przełomie lat siedemdziesiątych i osiemdziesiątych ubiegłego wieku. Sieci SOM są stosowane w wielu dziedzinach — od klasyfikacji po kompresję danych [Oso96].Sieci Kohonena działają w oparciu ostrategię *zwycięzca bierze wszystko* (ang. *winner takes all*). Ich podstawowymi zaletami są: nienadzorowany tryb uczenia, szybkość działania, zdolność do klasyfikacji danych wcześniej nie znanych i prostota (zarówno niski poziom skomplikowania koncepcji map Kohonena jak i ich implementacji).

Pojedynczy neuron (węzeł sieci) zawiera co najmniej dwa elementy: wektor położenia i wektor wag. Liczba wymiarów wektora wag odpowiada wymiarowi wektorów danych uczących. Ilość współrzędnych wektora położenia jest zależna od topologii mapy Kohonena.

Sieć składa się z neuronów połączonych bezpośrednio z wejściami[14] ułożonych w wiele warstw. Między węzłami sieci nie istnieją fizyczne połączenia, a topologia sieci jest zazwyczaj opisywana położeniem neuronów w przestrzeni. Najczęściej stosowane są sieci jednowymiarowe (neurony ułożone w linii), dwuwymiarowe (neurony ułożone w węzłach siatki) i trójwymiarowe. Ograniczenie się do maksymalnie trzech wymiarów daje możliwość prostej wizualizacji klasyfikowanych danych.

Algorytm 11: Ogólny schemat uczenia sieci SOM.

Input: Liczba epok e, liczba neuronów, parametry działania sieci P

```
1 Utwórz sieć o zadanych parametrach;
2 for i = 0; i < e do
3     BMU = Znajdź zwycięzcę;
4     Uaktualnij BMU i jego sąsiadów;
5     Uaktualnij(P, i);
6 end
```

Algorytm 11 przedstawia sposób uczenia sieci SOM. Jego ideę można wyjaśnić w następujący sposób: na początku treningu, podczas podawania kolejnych wektorów uczących modyfikowana jest cała sieć — neurony są rozkładane w całej przestrzeni danych. W miarę treningu neurony specjalizują do rozpoznawania konkretnych klas danych. Ze względu na to, że modyfikacji ulegają też wagi węzłów sąsiadujących ze zwycięzcą (BMU, ang. *Best Matching Unit*), to zostaje uzyskany efekt grupowania neuronów rozpoznających podobne wzorce blisko siebie.

Najważniejszymi parametrami sieci są: liczba neuronów, współczynnik sąsiedztwa i promień uczenia. Współczynnik uczenia i promień sąsiedztwa powinny się zmniejszać w czasie — od wartości stosunkowo dużych na początkowym etapie uczenia, do małych pod koniec uczenia. Sąsiedztwo jest liczone względem położenia neuronów w sieci, przy czym położenie samych neuronów nie zmienia się w czasie. Aktualizacja wag odbywa

14. Lub z warstwą wejściową, która przygotowuje dane, np. normalizuje wektory uczące.

się w taki sposób aby węzeł wygrywający i jego sąsiedzi rozpoznawali jeszcze lepiej podany wektor uczący. Im dalej od BMU, tym wagi są słabiej modyfikowane.

Sieci Kohonena zostały z powodzeniem zastosowane w systemie WEBSOM [Koh00] do stworzenia mapy dla ogromnej liczby dokumentów.

Pointcot et al. rozwijają idee mapy o wprowadzenie możliwości prostego dwupoziomowego grupowania hierarchicznego [Poi98]. Dla każdego neuronu w sieci zostaje stworzona mapa pomocnicza. Pozwala to na dokładniejsze nawigowanie w miejscach skupiających dużo dokumentów głównej sieci.

Podobną koncepcje zaproponowali Rauber et al. Zauważają oni, że istotnym ograniczeniem map Kohonena jest konieczność doboru liczby neuronów i płaska struktura sieci. Proponują rozszerzenie SOM do postaci rozrastającej się struktury hierarchicznej. Opracowana sieć została nazwana *rosnącą hierarchiczną siecią samoorganizującą się* (GHSOM, ang. *Growing Hierarchical Self-Organizing Map*) [Rau02, Dit00].

W GHSOM, po zakończeniu procesu nauki, obliczany jest średni błąd kwantyzacji (MQE, ang. *Mean Quantization Error*) dla każdego neuronu meq_n sieci — średnia odległość zaklasyfikowanych przez dany neuron obiektów od wektora wag dla danego neuronu. Następnie obliczany jest średni błąd kwantyzacji całej sieci MQE_i i porównywany z błędem sieci–rodzica[15] MQE_{i-1}.

W wypadku gdy $MQE_i \geqslant \tau_1 \cdot MQE_{i-1}$ następuje rozbudowa sieci o wiersz lub kolumnę w miejscu największego błędu. Po czym proces uczenia jest powtarzany. Jeżeli natomiast nie jest spełniony ten warunek, to dla każdego neuronu sprawdzane jest czy $meq_n \geqslant \tau_2 \cdot MQE_{i-1}$. Jeżeli ten warunek jest spełniony to zostaje zbudowana nowa sieć, dla wektorów uczących dla których neuron n był zwycięzcą.

SOM nie jest jedyną strukturą samoorganizującą się. System BEATCA [Klo06] wykorzystuje gaz neuronowy (GNG, ang. *Growing Neural Gas*) do stworzenia struktury dokumentów. Rozwijając idee GNG Hung i Wermter zaproponowali dynamicznie adaptatywną hybrydową sieć samoorganizującą (DASH, ang. *Dynamic Adapative Hybrid Self–Organizing*) [Hun05]. Została ona użyta do hierarchicznego pogrupowania dokumentów w środowisku niestacjonarnym (tj. takim, w którym nowe obiekty pojawiają się w czasie). Sieć ta nie tylko dopasowuje automatycznie swój rozmiar do danych, ale również parametry sterujące.

1.2.6. Inne metody

W przedstawionym powyżej przeglądzie nie zostały opisane wszystkie istniejące metody grupowania, ani nawet wszystkie rodzaje metod grupowania. Między innymi pominięto metody grupowania oparte na zbiorach rozmytych, na kratach, w oparciu o modelowania, itd.

Zostały wybrane metody, które zostały zastosowane (lub prezentują koncpecję, którą można łatwo zastosować do danych wielowymiarowych) w dziedzinie dokumentów tekstowych, z naciskiem na metody pozwalające na stworzenie hierarchii.

Kompletny przegląd różnych metod grupowania, nie tylko pod kątem organizacji kolekcji dokumentów testowych, można znaleźć w pracach: [Jai99, For06, Maz05, Wei06, Ber03, Kor05, Pal04, Sch05, Wu04]

15. Średni błąd kwantyzacji pierwszej sieci porównywany jest ze średnią odległością wszystkich wektorów uczących od centroidu dla całego zbioru uczącego.

1.3. Wydobywanie słów reprezentatywnych z dokumentów tekstowych

Dla stworzonej hierarchii grup dokumentów tekstowych, z każdej grupy należy wydobyć *słowa reprezentatywne* – słowa, które przypisane do odpowiednich węzłów hierarchii stworzyłyby tezaurusa.

Istnieją dwie główne metody wydobywania słów kluczowych: nadzorowane [Tur02, Tur00] i nienadzorowane [Mat04]. Pierwsze z nich polegają na ekstrakcji wiedzy z odpowiednio dużego, oznaczonego korpusu. Ze względu na brak dostępu do takich zasobów i przyjęte założenia metody nadzorowane zostaną pominięte w opisie.

Metody nienadzorowane natomiast starają się uchwycić pewne własności statystyczne tekstu i na ich podstawie ocenić, które terminy mogą posłużyć za słowa kluczowe dla danego dokumentu.

1.3.1. Metody lokalne

Metody lokalne w procesie wydobywania słów kluczowych posługują się tylko informacjami obejmującymi grupę dokumentów. Matsuo i Ishizuka zaproponowali algorytm wydobywania słów kluczowych z pojedynczego dokumentu, przy braku dostępu do globalnych statystyk całego korpusu tekstów [Mat04].

Ich podejście składa się z kilku następujących po sobie krokach. Początkowo wszystkie słowa w dokumencie są sprowadzane do rdzenia morfologicznego i usuwane są słowa ze stop–listy. Następnie dla 30% najczęstszych terminów zostaje zbudowana macierz ich koincydencji w zdaniach.

W celu polepszenia jakości wydobywanych słów, zostają one pogrupowane dwoma technikami. Pierwsza z nich podąża za obserwacją, mówiącą że jeżeli dwa słowa mają podobne rozkłady, to należą do jednej grupy. Podobieństwo rozkładów wyraża dywergencja Jensena–Shanona:

$$J(w_1, w_2) = \log 2 + \frac{1}{2} \sum_{w' \in C} \{h(P(w'|w_1) + P(w'|w_2)) - h(P(w'|w_1) - P(w'|w_2))\}, \tag{1.19}$$

gdzie: w_i — wyraz, C grupa, $h(x) = x \log(x)$, $P(w'|w_1) = freq(w', w_1)/freq(w_1)$, $freq$ to funkcja zwracająca częstość. Grupę tworzą wyrazy, których $J(w_1, w_2) > 0.95 \cdot \log 2$.

Druga technika grupowania polega na sprawdzaniu współwystępowaniu pary słów z innymi słowami — zostają utworzone grupy z wyrazów, których informacja wzajemna $M(w_1, w_2) > \log 2$ (równanie 1.20).

$$M(w_1, w_2) = \log \frac{P(w_1, w_2)}{P(w_1)P(w_2)} = \log \frac{N_{total} \cdot freq(w_1, w_2)}{freq(w_1) \cdot freq(w_2)} \tag{1.20}$$

Na podstawie współwystępowania słów z grupami zostaje obliczone oczekiwane wartość pojawienia się danego słowa w dokumencie. Po czym badane jest odchylenie od wartości zaobserwowanej. Do oceny odchylenia Matsuo i Ishizuka używają testu χ^2. Najczęstsze terminy są posortowane malejąco zgodnie z wynikiem testu i pierwsze n% z nich daje zbiór słów kluczowych dla tego dokumentu.

1.3.2. Metody globalne

Metody globalne w procesie wydobywania słów kluczowych posługują się informacjami zarówno dostępnymi na poziomie grupy dokumentów, jak i całego korpusu. Naiwne podejście może wykorzystać posortować słowa zgodnie z wagą $tf.idf$ i używać np. najlepszych 10 jako słów kluczowych.

Ciekawe rozwiązanie zostało zaproponowane w pracy [IP04]. Do wydobycia słów zostają zastosowane jednocześnie różne miary stosowane w dziedzinie wydobywania informacji.

Każdy wektor dokumentu jest ważony zgodnie ze schematem $tf.idf$. Dla każdego słowa występującego w grupie dokumentów zostaje przypisana waga w, będąca minimalną wartością $tf.idf$ dla danego słowa w każdym dokumencie. Prezentuje to podejście bardzo rygorystyczne - słowo musi występować naraz we wszystkich dokumentach grupy, aby było brane pod uwagę jako potencjalne słowo kluczowe.

Spośród wszystkich słów zostają dorzucone te, których częstość kolekcji df jest mniejsza niż df_{min} i większa niż df_{max}. Motywowane jest to tym, że słowa występujące zbyt często i zbyt rzadko nie są dobrymi dyskryminatorami. Salton podaje, że granicami tymi są $df_{min} = 1\%$, a $df_{max} = 10\%$ (za [IP04]).

Dla pozostałych słów zostaje wyznaczona miara wskaźnika ważności terminu (cv, ang. *cue validity*). Miara ta bada jak dane słowo jest charakterystyczne dla grupy dokumentów, $cv = \frac{tf_{grupy}}{tf}$.

Ostateczna waga dla terminu zostaje wyznaczona zgodnie ze wzorem:

$$w_t = \alpha \cdot min_{tf.idf_t} + \beta \cdot cv_t, \tag{1.21}$$

gdzie α i β są parametrami pozwalającymi na kontrolę wpływu poszczególnych części na wagę terminu.

Rozdział 2

Wydobywanie semantyki leksykalnej w oparciu o analizę korpusu

Inne podejście do budowy tezaurusa hierarchicznego, od przestawionego w rozdziale 1, polega na wydobywaniu relacji leksykalnych pomiędzy słowami niejako „wprost" z tekstu. Metody do wydobywania relacji bezpośrednio z tekstu opierają się *hipotezie dystrybucyjnej*. Jej autorem jest Z. Harris. Głosi ona (za [Kar01]):

> The meaning of entities, and the meaning of grammatical relations among them, is related to the restriction on combinations of these entities relative to other entities.
>
> (Znaczenie wyrażeń i relacji gramatycznych pomiędzy nimi jest powiązane z ograniczeniami nałożonymi na dopuszczalne kombinacje tych wyrażeń.)

Kalgren i Shalgren formalizują podejście do hipotezy dystrybucyjnej w ujęciu obliczeniowym [Kar01]. Twierdzą oni, że znaczenie M słowa w jest zdeterminowane jego dystrybucją D w tekście T. Dystrybucja słowa w tekście może być zdefiniowany jako suma wszystkich kontekstów C, w których słowo w występuje, czyli:

$$D(w,T) = \sum \{C \| w \in C\} \text{ determinuje } M(w) \tag{2.1}$$

Oznacza to, że „znaczenie słów może zostać wydobyte z tekstu i zapisane w systemie komputerowym"[Kar01].

W dalszej części rozdziału zostaną opisane metody wydobywania podobieństwa semantycznego słów z tekstu. Zadanie to można sformalizować jako próbę automatycznego wydobycia *funkcji podobieństwa semantycznego* (dalej FPS) (ang. *Semantic Similarity Function*) [Pia07a, Pia07c]. FPS odwzorowuje parę *jednostek leksykalnych* (pojedynczych słów lub wyrażeń składających się z wielu słów) na liczbę rzeczywistą. Weed i Weir wskazują jako przykład zastosowania takiej funkcji próby automatycznego zbudowania tezaurusa [Wee05, str. 440]. W pierwszej kolejności jednak zostanie opisane podejście pozwalające na ekstrakcję relacji hiponimii wprost z tekstu.

2.1. Wzorce składniowe

Hearst zaproponowała podejście do wydobywanie relacji hiponimii oparte na *wzorcach leksykalno–składniowych* (ang. *lexico–syntatic patterns*). Wyniki badań, których celem było stworzenie techniki możliwej do zastosowania do różnych rodzajów tekstów i nie wymagającej dużej ilości zakodowanej wiedzy, zostały opisane w raporcie [Hea92].

Metoda oparta na wzorcach składniowych jest motywowana prostą obserwacją. Hearst rozważa zdanie w języku angielskim:

> The bow lute, such as the Bambara ndang, is plucked and has an individual curved neck for each string.
>
> (Lutnia smyczkowa, taka jak Bambara ndang, jest szarpana i ma osobną wygiętą szyjkę dla każdej ze strun.[1])

Z powyższego zdania można wywnioskować, że „Bambara ndang" to rodzaj lutni smyczkowej. Zjawisko to jest bardzo ciekawe, ponieważ nie jest to definicja encyklopedyczna lutni „Bambara ndang" i osoba czytająca to zdanie może powiązać relacją terminy „lutnia smyczkowa" i „Bambara ndang" nawet w wypadku gdy nie wiadomo dokładnie czym jest „lutnia smyczkowa".

Obserwację tą można zapisać w postaci [Hea92] równania 2.2[2]. Każdy taki wzorzec pociąga za sobą zależność wyrażoną równaniem 2.3.

$$NP_0 \text{ such as } \{NP_1, NP_2, \ldots, (\text{ and } | \text{ or})\}\, NP_n \tag{2.2}$$

$$\forall_{i=1}^{n-1} hiponim(NP_i, NP_0) \tag{2.3}$$

Innymi słowy, gdy napotkamy w tekście fragment pasujący do wzorca składniowego 2.2, to możemy wnioskować, że pomiędzy częściami tego wzorca występuje relacja hiponimii. W przytoczonym wcześniej przykładzie oznacza to, że „Bambara ndang" jest hiponimem lutni smyczkowej.

Aby wzorzec składniowy był przydany musi on spełniać trzy kryteria: być częsty, dokładny i łatwy w rozpoznaniu. Gdy wzorzec będzie częsty w dowolnym tekście, to będzie można odkryć wiele słów spełniających daną relację. Im bardziej wzorzec będzie dokładny, tym wydobyte instancje relacji będą obarczone mniejszym błędem. Dzięki łatwości w rozpoznaniu, nie będzie wymagane użycie zaawansowanych narzędzi w przetwarzaniu.

Algorytm 12 przedstawia metodę odkrywania nowych wzorców. Początkowo zostają zaobserwowane interesujące wzorce w tekście. Następnie należy zgromadzić wiedzę o ich otoczeniu. Z wiedzy tej należy wydobyć schemat relacji, o ile taki istnieje. W końcowej fazie można zebrać kolejne przykłady zgodnie z wydobytym schematem.

Hearst wypróbowała tą metodę dla wzorca „such as" na korpusie encyklopedii Groliera. Spośród 8,6 miliona słów zostało znalezione ponad 7 tysięcy zdań zawierających zwrot „such as", w tym 152 spełniające nałożone na relacje ograniczania. Wynik eksperymentu charakteryzuje się dużą dokładnością. Na 152 wzorce przypada 226 unikalnych

1. Tłumaczenie z [Pia06b].
2. NP to skrót ot *nount phrase*, czyli frazy rzeczownikowej.

Algorytm 12: Schemat procesu odkrywania wzorców leksykalno–syntaktycznych [Hea92].

1 Wybór typu relacji R;
2 Zgromadzenie przykładów spełniających R;
3 Wyszukanie miejsc w występowania C_i relacji R w korpusie;
4 Wyszukanie wspólnych cech dla C_i i stworzenie hipotezy;
5 Wydobywanie słów spełniających relację R;
6 GOTO 2;

słów, z czego 180 istnieje w WordNecie. W tym 61 z 106 potencjalnych relacji (obydwa słowa są w WordNecie) istniało już w WordNecie.

Metoda oparta na wzorcach składniowych bardzo trudno zastosować dla języka polskiego — język polski jest językiem fleksyjnym, charakteryzującym się swobodnym szykiem szykiem zdania. Dlatego bez zastosowania skomplikowanych narzędzi lingwistycznych, takich jak analizator morfologiczny, tagger czy parser, trudno będzie odszukać potencjalne wzorce.

2.2. Word Space

Schütze zaproponował metodę zwaną Word Space [Sch93, Sch98] do rozstrzygania sensów słów (ang. *word sense disambiguation*). Metoda ta polega na zbieraniu informacji o współwystępowaniu słów w oknie o zadanej szerokości w macierz M.

W takiej reprezentacji zaciera się granica pomiędzy sensami. Dlatego należy stworzyć też *wektory kontekstowe* (ang. *context vectors*), które są centroidem (lub sumą) wektorów słów występujących w kontekstach. Wektory kontekstowe są następnie grupowane. Dla grup kontekstów wyliczane są centroidy — tworzą one tzw. *wektory sensów* (ang. *sense vectors*).

Po utworzeniu wektorów słów, wektorów kontekstowych i wektorów sensów możliwe jest rozstrzyganie pomiędzy sensami danego słowa w zadanym kontekście. Algorytm 13 przedstawia postępowania w metodzie Word Space.

Algorytm 13: Schemat rozstrzygania sensu słowa w technice Word Space [Sch98].

Input: Kontekst t wieloznacznego słowa v
Output: Sens słowa v
1 Przyporządkowanie t do odpowiedniego wektora kontekstowego $\vec{c}$;
2 Wydobycie wszystkich sensów $\vec{s_j}$ słowa s;
3 Przypisanie t do sensu j, którego wektor sensu $\vec{s_j}$ jest najbliższy $\vec{c}$;

2.3. Analiza semantyki ukrytej

Landauer i Dumais [Lan97] zaproponowali technikę *analizy opartej na semantyce ukrytej* (LSA, ang. *Latent Sematic Analysis*). LSA jest teorią i matematyczną metodą

wydobywania wiedzy o kontekstowym znaczeniu słów z dużych korpusów tekstów. Początkowo została zastosowana do poprawienia jakości wyszukiwarek opartych o model wektorowy[3] [Dee90]. Już w pierwszych pracach nad LSA Deerwester zauważył, że analiza oparta na semantyce ukrytej pomaga w rozwiązaniu problemów synonimii (kilka wyrazów mają to samo znaczenie w określonym kontekście, np. barwa a kolor) i polisemii (jedno słowo może mieć wiele znaczeń, np. „zamek") w systemach wydobywania informacji. Podejście to zostało wykorzystane jednak znacznie szerzej — do symulacji i wyjaśnienia niektórych funkcji poznawczych ludzkiego mózgu [Lan97].

Pierwszym krokiem LSA jest stworzenie macierzy zbierającej statystyki o występowaniu słów w dokumentach. Macierz jest następnie poddawana transformacji logent(zob. sekcja 1.1.2), czyli każda komórka macierzy jest logarytmowana i dzielona przez entropię wiersza. Ostatnim krokiem przygotowawczym analizy opartej na semantyce ukrytej jest redukcja wymiarów macierzy techniką SVD [4].

Skuteczność wydobywania semantyki leksykalnej LSA pokazano na przykładzie wykrywania synonimii, w części dotyczącej znajomośći słownictwa testu TOEFL (ang. *Test of English as a Foreign Language*). Zadanie postawione przed komputerem składa się z wielu pytań — dla każdego problemowego wyrazu należy wybrać jeden z czterech przedstawionych, będący synonimem do wyrazu problemowego.

Dla języka angielskiego LSA zostało wytrenowane na dużym korpusie akademickiej wersji encyklopedii Groliera[5] (4,5 miliona słów zawartych w ponad 30 tysiącach artykułów). LSA osiągnęło średni wynik 65% — co jest porównywalne ze średnim wynikiem obcokrajowców starających się dostać na studia w uniwersytatech w USA[Lan97].

2.4. Inne funkcje podobieństwa semantycznego

LSA jako FPS używa kosinusa kąta pomiędzy zredukowanymi wektorami słów. Można jednak spotkać się z użyciem innych funkcji - w [Gre93] badana jest między innymi miara Jaccarda. Dagan et al. porównują kilka FPS opartych na teorii prawdopodobieństwa [Dag97].

Turney do porównywania słów używa miary zaczerpniętej z dziedziny teorii informacji — *punktowej informacji wzajemnej* (PMI, ang. *Pointwise Mutual Information*) [Tur01]. Dla każdej pary słów buduje on zapytanie do wyszukiwarki AltaVista, po czym ocenia zwrócone wyniki zapytania. Rozwiązanie testuje również na teście synonimii z egzaminu TOEFL, w którym uzyskuje 73,75% poprawnych odpowiedzi. W dalszych badaniach Turney et al. połączyli kilka technik i uzyskując 97,5% w teście synonimii TOEFL praktycznie go rozwiązali [Tur03].

Weeds i Weir podkreślają, że różne dziedziny wymagają odmiennych modeli podobieństwa. Zaproponowali oni stworzenie elastycznej infrastruktury pozwalającej na obliczanie podobieństwa semantycznego dla różnych funkcji podobieństwa semantycznego w terminologii wydobywania koincydencji (ang. *co–occurrence retrieval*). Pozwala to na zbadanie różnych funkcji podobieństwa, ich ocenę i porównanie ich między sobą [Wee05, Wee03].

3. W tym ujęciu technika ta jest nazywana indeksowaniem opartym na semantyce ukrytej (LSI, ang. *Latent Semantic Indexing*).
4. Opis SVD znajduje się w sekcji 1.1.6.
5. Grolier's Academic American Encyclopedia

Rozdział 3

SuperMatrix — system pozwalający na badania metod wydobywania leksykalnych relacji semantycznych dla języka polskiego

W niniejszym rozdziale zostanie opisany stworzony system, dzięki któremu możliwe będzie przebadanie metod grupowania dokumentów i metod wydobywania leksykalnych relacji semantycznych opartych o hipotezę dystrybucyjną pod kątem ich przydatności w automatycznej ekstrakcji sieci semantycznych dla języka polskiego.

Podczas projektowania systemu przyjęto następujące założenia: system ma budowę modułową, jest prosty w utrzymaniu i ma udostępniać możliwość rozszerzania go o dodatkowe algorytmy, potrafi przetwarzać dużą ilości wielowymiarowych danych, a kod ma być przenośny. Ze względu na wydajność do implementacji większości systemu użyto języka c++. Jedynie aplikacja dostarczająca interfejsu graficznego do przeglądania wyników grupowania została napisana w języku Java.

3.1. Koncepcja budowy systemu

System składa się z szeregu luźno powiązanych ze sobą modułów funkcjonujących pod jedną wspólną nazwą *SuperMatrix*. Elementem je wiążącym jest pakiet *Matrices*. Definiuje on interfejs dla macierzy, w których mogą być przechowywane dokumenty i słowa w modelu wektorowym.

Pakiet *Corpus*[1] zawiera zbiór narzędzi pozwalających czytać tekst m.in. w postaci nieoznakowanej, jak i w formacie Korpusu IPI PAN [Prz04]. Pakiet ten w wypadku czytania zwykłego tekstu korzysta z analizatora morfologicznego Morfeusz [Wol06].

Pakiet *SmartComparator* określa interfejsy wykorzystywane przez metody selekcji cech, transformacji komórek macierzy i obliczania podobieństwa pomiędzy wierszami macierzy.

Pakiety *ClusteringBase* i *Clustering* odpowiedzialne są za grupowanie dokumentów. Pierwszy z nich definiuje interfejsy dla metod grupowania, drugi implementuje algorytmy grupowania, metody wydobywania słów reprezentatywnych i oceny jakości grupowania.

1. Autorem pierwszej wersji pakietu jest Grzegorz Godlewski, powstała ona na potrzeby stworzenia tagera TaKIPI [Pia06c].

Pakiet *JOSKIPI* implementuje *Język Opisu Stanu Korpusu IPI PAN*. Powstał on na potrzeby tagera TaKIPI [Pia06a, Pia06c][2]. Pozwala on na tworzenie wyrażeń w języku JOSKIPI, za pomocą których można testować spełnianie różnych gramatycznych relacji występujących w tekście (zob. 4.2.2).

Pakiet *SVD* dostosowuje bibliotekę SVDLIBC [Ber96] na potrzeby SuperMatrix. Biblioteka ta służy do redukcji wymiarów macierzy techniką rozkładu na czynniki osobliwe (ang. *Singular Value Decomposidtion*, zob. punkt 1.1.6).

3.2. Obsługa macierzy

Macierz jest podstawową strukturą danych, na której opiera się cały SuperMatrix. System musi spełniać bardzo wiele różnych funkcji. Dlatego macierze muszą być bardzo elastyczne. W zależności od zastosowania mogą one być odpowiedzialne między innymi za:

- obsługę macierzy gęstych i rzadkich,
- transformacje komórek macierzy,
- wydobywanie funkcji podobieństwa semantycznego z tekstu,
- grupowanie dokumentów,
- przetwarzanie macierzy zarówno dla małej jak i dużej liczby wymiarów[3].

Spełnienia powyższych wymagań nie jest w stanie zapewnić jedna metoda realizacji. Dlatego interfejs macierzy implementuje szereg klas, różniące się sposobem przechowywania macierzy w pamięci.

1. Macierz można przechowywać w pamięci w postaci rozwiniętej. Realizacja w tym modelu najlepiej sprawdza się dla niewielkich macierzy rzadkich. Wadą takiego rozwiązania jest duża ilość wymaganej pamięci, zaletą — szybkość działania.
2. Dla zbyt dużych macierzy, które nie zmieszczą się w dostępnej pamięci operacyjnej komputera, przewidziano możliwość przechowywania danych w bazie danych. Ze względów wydajnościowych wybrano obiektowy system zarządzania bazą danych *Oracle Berkeley DB*.
   W celu polepszenia wydajności dla poszczególnych zastosowań istnieją trzy sposoby przechowywania macierzy w bazie danych: całymi wierszami, całymi kolumnami i pojedynczymi komórkami w bazie. Macierz wierszowa pozwala na wykonywanie szybkich operacji na wierszach macierzy, jednak zajmuje dużo pamięci dyskowej i dostęp do kolumn macierzy jest bardzo nieefektywny. Analogiczne wady i zalety ma macierz kolumnowa. Macierz przechowywana w bazie pojedynczymi komórkami przyjmuje strategię pośrednią.
3. Rzadkie macierze można przechowywać w pamięci w formie skompresowanej. W zależności od zastosowań macierz można przechowywać w formacie skompresowanej kolumny (CCS, ang. *Compressed Column Storage*)[4] lub skompresowanego wiersza (CRS, ang. *Compressed Row Storage*) [Duf89, Ber96]. Formaty te przecho-

2. Autorem tego pakietu jest również Grzegorz Godlewski.
3. W ujęciu SuperMatrix suma wymiarów małej macierzy nie przekracza kilkunastu tysięcy. Największa przetworzona do tej pory macierz (rzadka) miała rozmiary 8 tyś na ok. 200 tyś.
4. Format ten też jest również nazwany formatem Harwella–Boeinga

wują jedynie niezerowe wartości komórek indeksowane po wierszach (kolumnach) wraz z niewielką ilością informacji dodatkowej.

Macierz można przechowywać na dysku w postaciach: CCS, CRS lub w bazie danych Berkeley DB.

3.3. Budowa Macierzy

TheArchitect jest narzędziem pozwalającym na wydobywanie macierzy koincydencji z korpusu. Możliwe jest tworzenie czterech typów macierzy.

1. »Terminy na dokumenty«. Macierze takie używane są w wydobywaniu funkcji podobieństwa semantycznego działającej na zasadzie takiej jak LSA. Zazwyczaj takie macierze są bardzo duże (por. 4.2.1), dlatego przechowywane są w bazie danych. Narzędzie czyta całe dokumenty tekstowe i zlicza statystyki wystąpień terminów w dokumentach.
2. Podstawowym zastosowaniem macierzy »dokumenty na terminy« to grupowanie dokumentów. Macierze te są bardzo rzadkie, wykonywane na nich jest bardzo dużo czasochłonnych operacji, dlatego warto je przechowywać w pamięci w formacie CRS. Narzędzie czyta całe dokumenty tekstowe i zlicza statystyki wystąpień terminów w dokumentach.
3. »Terminy na terminy« — macierze budowane w oparciu o kontekst oknowy (por. punkt 2.2). Nad tekstem jest przesuwane okno o wielkości k terminów i zbierane są statystki koincydencji terminu znajdującego się w centrum okna z pozostałymi terminami.
4. »Terminy na operatory JOSKIPI«. Macierze tworzone w podobnej idei do macierz z punktu 3. Wiersze macierzy odpowiadają terminom, dla których budowana jest funkcja podobieństwa semantycznego. Kontekstami są operatory JOSKIPI sprawdzające odpowiednie ograniczenia gramatyczne. Ze względu na budowę JOSKIPI wielkość okna odpowiada wielkości zdania [Pia06a].

3.4. Przetwarzanie macierzy

Pakiet *SmartComparator* pozwala na definiowanie wielu funkcji do porównywania wierszy macierzy. Odbywa się według następującego schematu:

1. Globalny wybór cech macierzy. Selekcja ta jest oparta na statystykach możliwych do wyliczenia dla całej macierzy, np. entropii wierszy.
2. Transformacja komórek macierzy wg różnych schematów ważenia (zob. 1.1.2).
3. Wybór lokalnych cech na podstawie porównania dwóch wierszy.
4. Wyliczenie wartości podobieństwa pomiędzy parą wierszy.

Dzięki przyjęciu takiego sposobu postępowania możliwe jest odtworzenie bardzo wielu funkcji podobieństwa zaproponowanych w literaturze (por. rozdział 2 i punkt 4.2).

Rozdział 4

Przeprowadzone badania

Wszystkie badania zostały przeprowadzone na dokumentach wchodzących w skład *Korpusu IPI PAN*[1] w wersji 2.0 (dalej KIPI) [Prz04]. Zawiera on ponad 254 miliony słów w 357 384 dokumentach o różnorodnym charakterze, m.in. prawniczym, politycznym, publicystycznym i beletrystycznym. W skład korpusu wchodzą m.in. sprawozdania z pracy Sejmu i Senatu, artykuły prasowe Dziennika Polskiego, książki.

Tekst w korpusie został oznaczony przy użyciu analizatora morfologicznego Morfeusz [Wol06]. Morfeusz przypisuje każdemu słowu listę struktur mówiących o możliwych interpretacjach danego słowa. Następnie został użyty tagger TaKIPI [Pia06c, Pia06d] — narzędzie rozstrzygające, która interpretacja jest właściwa w danym kontekście.

Do badań zostały wydzielone podzbiory KIPI:

- $KIPI_{full}$ — cały korpus,
- DZP_{98} — zbiór 25 486 artykułów z „Dziennika Polskiego" z 1998 roku,
- DZP_{04} — zbiór 7 776 artykułów z „Dziennika Polskiego" z okresu od stycznia do kwietnia 2004 roku,
- DZP_{98-01} — zbiór 185 066 artykułów z „Dziennika Polskiego" z lat 1998 — 2001,
- DZP_{sport} — zbiór 7 516 artykułów z „Dziennika Polskiego" z lat 2003–2004 zawierający artykuły w kategorii „Sport",
- $DZP_{gospodarka}$ — zbiór 1 603 artykułów z „Dziennika Polskiego" z lat 2003–2004 zawierający artykuły w kategorii „Gospodarka".

Podział taki jest motywowany następującymi czynnikami. Funkcja podobieństwa semantycznego wymaga jak największej ilości danych, dlatego cały korpus został użyty do jej nauki. Do testów grupowania natomiast wybrano jedynie artykuły prasowe, żeby zminimalizować wpływy długich dokumentów na algorytm grupowania. Wybrano zbiory o małych licznościach, aby łatwiejsza była ręczna ocena jakości grupowania.

Zbiór DZP_{98} zawiera teksty podzielone na kilka ogólnych kategorii: *Kraków*,*Gospodarka*, *Sport*, *Magazyn*, *Kraj*, *Świat*. Natomiast artykuły w DZP_{04} zostały również podzielone na regiony, m.in. *Bochnia*, *Podhale*, *Oświęcim* (razem z ogólnymi klasami zostały wydzielone 24 kategorie).

Dwa tematyczne podzbiory, DZP_{sport} i $DZP_{gospodarka}$, zostały wyodrębnione w celu zbadania przydatności metod grupowania w konstrukcji tezaurusa dziedzinowego.

1. Korpus dostępny jest pod adresem: http://korpus.pl/

W dalszej części rozdziału przedstawione zostały wykonane eksperymenty. Zgodnie z celem pracy zostały one podzielone na dwie części: w pierwszej kolejności zostały opisane badania metod grupowania dokumentów, po nich następuje analiza wyników z badań przeprowadzonych nad metodami opartymi o hipotezę dystrybucyjną pod kątem przydatności w wydobywaniu leksykalnych relacji semantycznych.

4.1. Metody grupowania

Oceny jakości metod grupowania można dokonać na trzy zasadnicze sposoby: w oparciu o kryterium wewnętrzne (ang. *internal criteria*), w oparciu o kryterium zewnętrzne (ang. *external criteria*), i ręcznie (ang. *end–user criteria*) [For06].

Pierwszy z tych sposobów polega na analizie grup przy użyciu miary pozwalającej na zbadanie pewnych własności mierzalnych rozwiązania. Przykładową metodą jest indeks Dunna [Sch05] (równanie 4.1, d_{min} to minimalna odległość pomiędzy dwoma obiektami w różnych grupach, d_{max} to maksymalna odległość pomiędzy dwoma obiektami w jednej grupie). Sprawdza on jak zwarte i odseparowane są grupy dla najgorszego przypadku.

$$D_I = \frac{d_{min}}{d_{max}} \tag{4.1}$$

Metod oceny w oparciu o kryteria wewnętrzne należy stosować bardzo ostrożnie do porównania algorytmów grupowania. Należy dokładnie określić punkt odniesienia, co nie jest proste [For06]. Metody te najczęściej są one stosowane są w wypadku, gdy nie istnieje wzorcowa kolekcja oznaczona ręcznie. Innym problemem jest interpretacja wyników, szczególnie w dziedzinie danych wielowymiarowych.

Wynik oceny w oparciu o kryterium zewnętrzne mówi jak dobrze zaproponowany algorytm przybliża rozwiązanie stworzone przez człowieka. Istnieje wiele miar mówiących o jakości grupowania. *Czystość* (ang. *purity*) mierzy jak jednorodne są grupy:

$$Purity(L, C) = \frac{1}{N} \sum_k \max_j |c_k \cap l_j|, \tag{4.2}$$

gdzie $C = \{c_1, c_2, \ldots, c_k\}$ to zbiór grup, a $L = \{l_1, l_2, \ldots, l_j\}$ to zbiór klas wzorcowych. $Purity(L, C)$ przyjmuje wartości od 0 (najgorszy przypadek), do 1 (najlepsza jakość grupowania). Miara ta ma jedno ograniczenie — preferuje rozwiązania o bardzo dużej liczbie grup. Przypisując każdy dokument do innej grupy wartość czystości jest maksymalna [Man07].

Indeks Randa (równanie 4.3) mierzy trafność w ujęciu podejmowania decyzji dla każdej pary dokumentów [Man07, Sch05]. W ujęciu grupowania decyzje można podzielić na cztery kategorie.

- Decyzja jest prawdziwie pozytywna (TP, ang. *True Positive*), jeżeli para dokumentów znajduje się razem w grupie wzorcowej i wygenerowanej.
- Decyzja jest prawdziwie negatywna (TN, ang. *True Negative*), jeżeli para dokumentów nie znajduje się razem ani w grupie wzorcowej ani w wygenerowanej.
- Decyzja fałszywie negatywna (FN, ang. *False Negative*) polega na rozdzieleniu pary dokumentów z kolekcji wzorcowej pomiędzy różne grupy.

- Decyzja fałszywie pozytywna (FP, ang. *False Positive*) polega na umieszczeniu takiej pary dokumentów w jednej grupie, która nie znajdowała się w żadnej grupie wzorcowej.

$$R_I = \frac{TP + TN}{TP + FP + FN + TN} \tag{4.3}$$

Ograniczeniem indeksu Randa jest, to że na równo traktuje decyzje fałszywie pozytywne i fałszywie negatywne [Man07]. Dlatego warto zastosować miary znane z dziedziny wydobywania informacji [For06]: dokładność (równanie 4.4), kompletność (równanie 4.5) i średnią harmoniczną z dokładności i kompletność (równanie 4.6). Im mniejszy parametr β średniej harmonicznej, tym bardziej jest karane połączenie dwóch dokumentów nie będących w jednej grupie wzorcowej.

$$P = \frac{TP}{TP + FP} \tag{4.4}$$

$$R = \frac{TP}{TP + FN} \tag{4.5}$$

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R} \tag{4.6}$$

Podczas oceny w oparciu o kryterium zewnętrzne należy wziąć pod uwagę jakość podziału wzorcowego. Z jednej strony proces tworzenie wzorca jest podatny na błędy, z drugiej — algorytm może znaleźć lepsze rozwiązanie niż przyjęte przez autorów wzorca. Z tych względów warto oceniać wyniki ręcznie mając na uwadze cel, któremu służy grupowanie.

Ze względu na przytoczone ograniczenia metod oceny w oparciu o kryteria wewnętrzne i dostęp do kolekcji oznaczonej ręcznie algorytmy grupowania zostały ocenione przy użyciu metod ręcznych i opartych o kryteria zewnętrzne.

4.1.1. Ocena w oparciu o kolekcje wzorcowe

Eksperymenty zostały przeprowadzone dla dwóch metod grupowania: ROCK i GHSOM. Ze względu na cel grupowania — stworzenie szkieletu hierarchii dla tezaurusa — wprowadzono do nich pewne modyfikacje. Dla algorytmu ROCK został zmieniony warunek zatrzymania algorytmu — grupowanie się kończy, gdy już żadna grupa nie może zostać połączona. Jako funkcji podobieństwa dla ROCK użyto przeskalowanego kosinusa kąta ($sim(a, b) = (1 + cos(a, b))/2$). Nie zostało też wymuszone przypisanie każdego dokumentu do grupy — ROCK łączy grupy jedynie w momencie, gdy pomiędzy nimi występuje co najmniej jedno połączenie. Dla GHSOM zrezygnowano z możliwości rozrastania się mapy w szerz (parametr $\tau_1 = 0$). Dzięki temu można było porównać jakość grupowania w zależność od ustalonej liczby neuronów.

Ze względu na rodzaj przyjętych miar oceniających jakość grupowania, porównywane zostały tylko grupy na szczycie hierarchii dla ROCK i pierwsza warstwa mapy GHSOM.

ROCK

Dla algorytmu ROCK przeprowadzono dwa rodzaje eksperymentów: wpływ parametru θ na jakość grupowania i wpływ sposobu ważenia.

θ	$Purity$	R_I	P	R	$F_{\beta=0.1}$	$F_{\beta=1}$	$F_{\beta=2}$	k
0,65	0,15	0,09	0,08	0,99	0,09	0,16	0,32	3
0,70	0,22	0,29	0,09	0,78	0,09	0,17	0,31	135
0,75	0,64	0,83	0,29	0,33	0,29	0,31	0,33	165
0,80	0,79	0,84	0,52	0,24	0,52	0,33	0,27	73

Tabela 4.1. Ocena działania algorytmu ROCK dla korpusu DZP_{04}. Komórki macierzy zostały przetransformowane wagą *logent*, k to liczba korzeni.

Tabela 4.1 przedstawia wyniki ROCK dla korpusu DZP_{04}, przy użyciu wagi *logent* do transformacji komórek macierzy. Wraz ze zwiększaniem się parametru θ rośnie precyzja, dokładność i czystość grup, a maleje kompletność. Dla $\theta = 0,64$ otrzymujemy jedynie 3 grupy w korzeniu — algorytm nadmiernie łączy grupy — średnia liczba sąsiadów dla dokumentu wynosi 50,3 (w porównaniu do 0,4 dla $\theta = 0,8$).

Do podobnych wniosków można dojść przyglądając się wynikom przy użyciu wagi $tf.idf$ (tabela 4.2) — z tym, że daje ona nieco gorsze wyniki ogólnie. Należy zwrócić uwagę też na różną liczbę grup w korzeniu hierarchii dla różnych schematów ważenia. Użyte miary mogą zawyżać ocenę dla większej ilości grup. Jednak dla $\theta = 0.75$, pomimo tego że waga *logent* dała mniejszą liczbę grup w korzeniu hierarchii, to wyniki są lepsze.

θ	$Purity$	R_I	P	R	$F_{\beta=0.1}$	$F_{\beta=1}$	$F_{\beta=2}$	k
0,65	0,15	0,09	0,08	0,99	0,09	0,16	0,32	3
0,70	0,16	0,13	0,09	0,94	0,1	0,17	0,31	50
0,75	0,4	0,81	0,16	0,25	0,16	0,19	0,22	209
0,80	0,64	0,89	0,46	0,09	0,44	0,16	0,11	170

Tabela 4.2. Ocena działania algorytmu ROCK dla korpusu DZP_{04}. Komórki macierzy zostały przetransformowane wagą $tf.idf$, k to liczba korzeni.

Wartość zarówno precyzji jak i kompletności jest niepokojąco niska. Dlatego dokładnie został przeanalizowany korpus DZP_{04}. Okazało się, że wiele grup tematycznie podobnych zostało przypisanych do różnych kategorii. Szczególnie widoczne to było dla artykułów sportowych — znajdują się one bardzo często w kategoriach regionalnych. Dlatego przeprowadzono test na korpusie DZP_{98}, który nie został podzielony na tak dużo kategorii. Wyniki oceny grupowania tego zbioru dokumentów są dużo lepsze (tabele 4.3 i 4.4). Dla $\theta = 0,75$ uzyskano precyzję na poziomie 93%. Również tutaj wyniki dla wagi $tf.idf$ są gorsze pod względem precyzji.

W tabelach 4.5 i 4.6 zostały pokazane wyniki eksperymentów dla inaczej podzielonego korpusu DZP_{04}. Trzy wyraźne kategorie — sport, gospodarka i świat — pozostały niezmienione, pozostałe kategorie zostały połączone w jedną. Wyniki oceny precyzji poprawiły się nieznacznie kosztem kompletności, jednak czystość zwiększyła się w istotny sposób. Potwierdza to wcześniejsze spostrzeżenie — ROCK nie jest w stanie wykryć

θ	$Purity$	R_I	P	R	$F_{\beta=0.1}$	$F_{\beta=1}$	$F_{\beta=2}$	k
0,65	0,3	0,25	0,23	0,96	0,23	0,38	0,6	113
0,70	0,59	0,56	0,31	0,58	0,31	0,41	0,5	608
0,75	0,91	0,75	0,93	0,11	0,87	0,2	0,14	553
0,80	0,96	0,68	0,82	0,05	0,82	0,09	0,06	344

Tabela 4.3. Ocena działania algorytmu ROCK dla korpusu DZP_{98}. Komórki macierzy zostały przetransformowane wagą *logent*, k to liczba korzeni.

θ	$Purity$	R_I	P	R	$F_{\beta=0.1}$	$F_{\beta=1}$	$F_{\beta=2}$	k
0,65	0,3	0,24	0,23	0,97	0,23	0,38	0,6	68
0,70	0,41	0,41	0,24	0,68	0,24	0,35	0,5	489
0,75	0,76	0,75	0,44	0,75	0,43	0,17	0,12	793
0,80	0,89	0,75	0,84	0,02	0,6	0,04	0,02	698

Tabela 4.4. Ocena działania algorytmu ROCK dla korpusu DZP_{98}. Komórki macierzy zostały przetransformowane wagą $tf.idf$, k to liczba korzeni.

subtelnych różnic pomiędzy dokumentami w kategoriach regionalnych, a artykułami w kategoriach bardziej ogólnych.

θ	$Purity$	R_I	P	R	$F_{\beta=0.1}$	$F_{\beta=1}$	$F_{\beta=2}$
0,65	0,88	0,78	0,78	0,99	0,78	0,87	0,94
0,70	0,86	0,61	0,73	0,74	0,73	0,74	0,74
0,75	0,86	0,3	0,57	0,1	0,54	0,18	0,13
0,80	0,85	0,3	0,57	0,06	0,52	0,01	0,07

Tabela 4.5. Ocena działania algorytmu ROCK dla korpusu DZP_{04}. Komórki macierzy zostały przetransformowane wagą *logent*. Wyniki dla podziału na klasy: Sport, Gospodarka, Świat, Pozostałe.

Algorytm nie grupuje wszystkich dokumentów — dzięki temu uzyskana jest większa precyzja i czystość. Jest to własność bardzo ważna podczas tworzenia szkieletu dla hierarchii słów. Dokumenty źle zaklasyfikowane mogą mieć bardzo duży wpływ na pogorszenie się jakości wydobywania słów służących do opisu grup. Z tego punktu widzenia lepszy jest brak decyzji o przypisaniu dokumentu do grupy, niż klasyfikacja chociaż w małym stopniu błędna.

Ręczna analiza wyników pokazała, że ROCK dla parametru $\theta \geqslant 0.7$ tworzy bardzo sensowne hierarchie grup dokumentów. Po dokonaniu dokładna inspekcji wyników dla mniejszego korpusu DZP_{04} można dojść do wniosku, że ani razu nie nastąpiło połączenie głównych tematów, tj. nie istnieje grupa, w której znalazłyby się dokumenty mówiące np. zarówno o sporcie jak i polityce. Algorytm potrafi wyodrębnia więcej kategorii niż istniejących w podziale wzorcowym, np. wyraźny jest rozdział pomiędzy poszczególnymi dyscyplinami sportowymi (m.in. dobrze wyodrębniona są grupy mówiąca o skokach narciarskich, koszykówce, czy piłce nożnej).

θ	$Purity$	R_I	P	R	$F_{\beta=0.1}$	$F_{\beta=1}$	$F_{\beta=2}$
0,65	0,88	0,79	0,79	0,99	0,79	0,88	0,94
0,70	0,88	0,74	0,78	0,93	0,78	0,85	0,9
0,75	0,97	0,27	0,64	0,11	0,61	0,2	0,14
0,80	0,89	0,23	0,67	0,02	0,47	0,03	0,02

Tabela 4.6. Ocena działania algorytmu ROCK dla korpusu DZP_{04}. Komórki macierzy zostały przetransformowane wagą $tf.idf$. Wyniki dla podziału na klasy: Sport, Gospodarka, Świat, Pozostałe.

Utworzona hierarchia dla wszystkich korpusów testowych algorytmem ROCK miała jedną wadę — nadmierne rozrastanie się hierarchii, podczas łączenia małych grup w większe. Często można zaobserwować zjawisko tworzenia jednej grupy w kilku krokach algorytmu poprzez dołączanie do niej pojedynczych dokumentów. Z tego względu należy wprowadzić do ROCK mechanizm pruningu, pozwalający na odcięcie takich gałęzi.

SOM

Pierwszym eksperymentem z siecią GHSOM było zbadanie wpływu liczby neuronów na jakość grupowania — testowano sieci o strukturze kwadratowej składającej się z od 4 do 49 neuronów (tabela 4.7). Ze względu na losową inicjalizację wag sieci, grupowanie powtórzono pięć razy i wzięto średnią z wyników. Interesującym jest to, iż pomiędzy dwoma przebiegami różnice w ocenie były nieznaczne. Oznacza to, że algorytm szybko zbiega dla dużej ilości dokumentów do sklasyfikowania podanych w jednej epoce.

Struktura	$Purity$	R_I	P	R	$F_{\beta=0.1}$	$F_{\beta=1}$	$F_{\beta=2}$
7x7	0,88	0,22	0,83	0,03	0,65	0,05	0,03
6x6	0,88	0,23	0,83	0,04	0,7	0,08	0,05
5x5	0,88	0,23	0,79	0,05	0,69	0,09	0,06
4x4	0,88	0,25	0,82	0,07	0,75	0,14	0,09
3x3	0,88	0,29	0,8	0,14	0,77	0,23	0,16
2x2	0,88	0,38	0,81	0,28	0,8	0,4	0,3

Tabela 4.7. Ocena działania algorytmu GHSOM w zależności od struktury sieci dla korpusu DZP_{04}. Komórki macierzy zostały przetransformowane wagą $tf.idf$. Wyniki dla podziału na klasy: Sport, Gospodarka, Świat, Pozostałe.

Spostrzeżenie to potwierdza także przeprowadzony eksperyment z wpływem czasu uczenia na wyniki. Zostały ocenione sieci wytrenowane w czasie od 5 do 15 epok. Wyniki nie uległy statystycznie znaczącej zmianie. Również zastosowanie wagi *logent* nie wpłynęło istotnie na wyniki.

Obserwacje te nie oznaczają, że testowane parametry nie mają wpływu na pracę sieci. Przyjęta metoda oceny nie jest w stanie wychwycić niektórych własności sieci Kohonena. Ręczna analiza wyników grupowania pokazała, że dla większej liczby epok uczących dokumenty zostają rozłożone bardziej równomiernie na całej mapie. Przy

zastosowaniu wagi *logent* można zauważyć bardziej wyraźne „wyspy" — skupiska dokumentów podobnych — niż dla wagi $tf.idf$. Z punktu widzenia jednak przyjętego celu grupowania, własności przestrzenne sieci nie są czynnikiem najważniejszym.

Ręczna analiza wyników grupowania pokazała, że czasami na wyższych poziomach hierarchii w grupach znajdują się dokumenty z kilku kategorii tematycznych. Zauważalne jest szczególnie w przypadku mniejszych sieci, jednak zjawisko to występuje nawet dla sieci większych. Jest to cecha niepożądana, mogąca mieć istotny wpływ na jakość wydobytych słów kluczowych. Dzieje się tak dlatego, że GHSOM w przeciwieństwie do ROCK klasyfikuje wszystkie dokumenty.

Sama struktura hierarchii dla parametru $\tau_2 = 0.0035$ [2] była bardziej zrównoważona niż dla ROCK. Dla większych sieci miała maksymalnie 4 poziomy głębokości, dla sieci średnich rozmiarów (9 — 25 neuronów) osiągała maksymalnie 6 poziomów. Jest to porównywalne ze strukturą hiponimii Słowosieci (8 poziomów hierarchii bez korzenia).

Algorytm ROCK pod względem czasu działania[3] jest kilkakrotnie szybszy od GHSOM, jednak wymaga o wiele więcej pamięci operacyjnej. Wydajność algorytmów nie jest tutaj jednak najważniejszym kryterium — ważne, żeby wyniki uzyskać w rozsądnym czasie (poniżej kilku dni). Testowane korpusy ROCK grupował w czasie do trzech godzin, GHSOM do około sześciu godzin.

4.1.2. Analiza przydatności metod grupowania w wydobywaniu leksykalnych relacji semantycznych

Po stworzeniu hierarchii grup dokumentów tekstowych, z każdej grupy należy wydobyć słowa reprezentatywne. W tym celu zastosowano połączenie technik lokalnych z globalnymi do wydobywania słów. Metodę ważenia zaproponowaną w [IP04] połączono z techniką opartą na teście istotności statystycznej χ^2 z pracy [Mat04]. Po odrzuceniu słów t nie spełniających kryterium $df_{min} < df_t < df_{max}$, każdemu słowu zostaje przypisana waga zgodnie ze wzorem 4.7.

$$w_t = \alpha \cdot min_{tf.idf_t} + \beta \cdot cv_t + \gamma \cdot \chi_t^2, \tag{4.7}$$

gdzie: $min_{tf.idf_t}$ jest minimalną wartością wagi $tf.idf$ w grupie dokumentów dla słowa t, $cv = \frac{tf_{grupy}}{tf}$, a χ_t^2 to waga obliczona zgodnie z algorytmem zaproponowanym przez Matsuo i Ishizuka (patrz punkt 1.3.1). Algorytm ten został zaprojektowany z myślą o wyznaczaniu wag dla słów w obrębie jednego dokumentu, dlatego dokumenty w grupie zostają połączone, przed wyliczeniem wartości χ_t^2. Parametry α, β i γ kontrolują siłę oddziaływania poszczególnych wag składowych.

Okazało się, że kryterium $min_{tf.idf_t}$ odrzucające słowa nie występujące w każdym dokumencie, jest zbyt silne, szczególnie dla grup znajdujących się bliżej korzenia. Dlatego zostało ono złagodzone — słowo musi wystąpić w 90% dokumentów.

Jakość rozwiązania została następnie oceniona o Słowosieć, tj. została zbadana dokładność odwzorowania struktury hiponimii Słowosieci przez rozwiązanie automatyczne. Wynik oceny pokazał słabość zaproponowanego podejścia — dokładność dla

2. Wartość sugerowana w [Rau02].
3. Eksperymenty były przeprowadzane na dwóch różnych komputerach dwurdzeniowych. Pierwszy z nich został wyposażony w procesor Athlon 64 X2 o mocy obliczeniowej 2,21 GHz i 4 GB pamięci operacyjnej. Drugi z nich został wyposażony w procesor Core 2 Duo o mocy obliczeniowej 2,13 GHz i 2 GB RAM.

najlepszej metody nie przekraczała jednego procenta (udało się odwzorować jedynie 86 relacji w najlepszym podejściu). Próba budowy tezaurusa o tematyce sportowej (korpus DZP_{sport}) i gospodarczej (korpus $DZP_{gospodarka}$) w nieznacznym stopniu poprawiły wyniki.

Podczas ręcznej analizy powstałej hierarchii słów zostały zauważone dwa problemy w stosowanym podejściu. Po pierwsze — kontekst o wielkości dokumentu jest zbyt szeroki. Zastosowano więc naiwne podejście do podziału dokumentów na mniejsze, spójne tematycznie fragmenty (punkt 1.1.7). Nie poprawiło to jakości stworzonego tezaurusa, a pogorszyło jakość grupowania — użyte w algorytmach grupowania funkcje podobieństwa nie rozróżniają dobrze zbyt krótkich dokumentów.

Drugim problemem jest sposób ważenia słów. Ma on korzenie w dziedzinie wydobywania słów kluczowych na potrzeby systemów wydobywania i wyszukiwania informacji. Wydobyte słowa bardzo dobrze opisują stworzone grupy, np. grupę dokumentów mówiących o interwencyjnym skupie zboża i żniwach na terenie Małopolski opisują słowa (kolejność podana zgodnie z wyliczoną wagą): »zboże, pszenica, tona, rolnik, agencja«. Na niższych poziomach hierarchii dochodzą słowa »skup, interwencja, magazyn«. Między tymi grupami słów istnieje relacja semantyczna, jednak jest ona bardzo odległa.

Zaletą tego podejścia jest natomiast potencjalna możliwość wykrywania polisemii słów (ile znaczeń może mieć dane słowo). Te same słowa występują w różnych częściach powstałej hierarchii co w przyszłości może zostać wykorzystane do rozwiązania problemu polisemii w metodach wydobywających relację leksykalne wprost z tekstu (rozdział 2 i wyniki badań z punktu 4.2).

4.2. Wydobywanie semantyki leksykalnej w oparciu o hipotezę dystrybucyjną

Jako, że dla języka polskiego nie istnieje odpowiednik testu synonimii egzaminu TOEFL (a sam test TOEFL dla języka angielskiego jest już uznany a problem rozwiązany [Tur03]) należało skorzystać z innego sposobu oceny.

Freitag et al.[Fre05] zaproponowali metodę automatycznego generowania testów na zasadach TOEFL'a w oparciu o WordNet. Tworzenie testu zaczyna się od wyboru słów kandydujących z korpusu. Do listy wchodzą słowa występujące w korpusie określoną ilość razy i nie znajdujące się na stop–liście.

Dla każdej uporządkowanej pary słów z listy kandydatów tworzona jest para (pytanie, odpowiedź), jeżeli istnieje pomiędzy nimi relacja synonimii w WordNet'ie. Odrzucane są pary zbyt podobne na poziomie znakowym[4]. Takie pytania są trywialne dla człowieka, a w wypadku metod automatycznych zawyżałyby oceny metod stosujących pewne uproszczenia.

Dla każdej tak stworzonej pary pytanie–odpowiedź są losowane trzy wyrazy nie będące w relacji synonimii z przetwarzaną parą. Tak wygenerowany test nosi nazwę *WordNet–Based Synonymy Test* (WBST).

Listy słów kandydujących dla WBST w [Fre05] zostały wydobyte z korpusu North American News, który składa się z 350 milionów słów. Na liście słów znalazły się tylko wyrazy występujące co najmniej tysiąc razy w korpusie. Uzyskano 23 570 pytań, w

4. Np.: group i grouping.

tym 9 887 dla rzeczowników, 7 398 dla czasowników, 5 824 dla przymiotników i 461 dla przysłówków. Średni wynik siedmiu osób rozwiązującej ten test to 88.4%. Przykładowe pytania dla WBST:

```
technology:
  A. engineering B. difference
  C. department  D. west

stadium:
  A. miss B. hockey
  C. wife D. bowl

string:
  A. giant B. ballet
  C. chain D. hat

trial:
  A. run   B. one-third
  C. drove D. form
```

Test WBST jest trudniejszy od TOEFL'a. Metoda osiągająca w TOEFL'u 82%, w WBST osiągnęła jedynie 67.6%. Metoda optymalna zaproponowana w [Fre05] osiągnęła 75.8% poprawnych odpowiedzi.

Dla języka polskiego listę słów kandydujących uzyskano z całego KIPI. Użyto również słów występujących w korpusie co najmniej tysiąc razy. Ze względu na budowę Słowosieci (występuje w niej dużo synsetów jednoelementowych) rozszerzono test o uwzględnianie hiperonimii — jeżeli dla danego pytania nie istnieje drugie słowo z listy kandydatów w bezpośredniej relacji synonimii w Słowosieci, to sprawdzano czy wśród kandydatów znajduje się wyraz bardziej ogólny. W ten sposób uzyskano 4782 pytania dla rzeczowników[5].

W celu sprawdzenia poprawności metody, stworzono test składający się z wylosowanych 79 pytań i poproszono dwudziestu czterech Polaków o jego rozwiązanie. Średni uzyskany przez nich wynik wyniósł 89.29%. Przy czym zgodność między rozwiązującymi test, mierzona współczynnikiem kappa Cohena [Coh60] wynosi od 0.19 do 0.47.

Oto przykładowe wygenerowane pytania testu WBST:

```
grupa:
  A. depresja B. kompromis
  C. skóra    D. zespół

aspirant:
  A. kandydat B. kolegium
  C. porada   D. rekord

przykrość:
  A. bilans     B. cnota
```

5. Ograniczono się jedynie do rzeczowników, ponieważ są one we wstępnej wersji Słowosieci najlepiej opisane.

```
  C. literatura D. żal

ludność:
  A. most B. półka
  C. szum D. wspólnota
```

4.2.1. LSA dla języka polskiego

W przeprowadzonych eksperymentach macierz została zbudowana w oparciu o korpus DZP_{98-01} (185 066 artykułów z „Dziennika Polskiego" z lat 1998 — 2001).

W oryginalnym LSA słowa nie są poddawane wcześniejszej obróbce, wszystkie operacje są wykonywane na łańcuchach znakowych. Dla języka polskiego jednak liczba różnych form słów jest zbyt duża — rzeczowniki mogą występować w 14 formach. Dla wybranych rzeczowników dałoby to aż 64 554 wiersze macierzy. Dlatego należało sprowadzić wyrazy do form bazowych.

Dla zaproponowanego rozszerzonego testu WBST, LSA dla języka polskiego osiągnęło maksymalnie 58.07% (zob. tabela 4.8). Jest to mniej niż wynik oryginalnego LSA [Lan97], jednak test WBST jest trudniejszy od TOEFL[Fre05].

Redukcja wymiaru	50	300	500	750	1000
Dokładność[%]	53.74	57.76	**58.07**	58.06	58

Tabela 4.8. Wynik LSA w WBST

4.2.2. Kontekst oknowy — podejście naiwne lingwistycznie

W celu pokonania większości ograniczeń LSA można zastosować technikę Word Space zaproponowaną przez Schütze[Sch93, Sch98]. Polega ona na przemieszczaniu okna nad tekstem i zbieraniu statystyk współwystępowania wyrazów. Podejście to daje możliwość rozbudowy metody o wiedzę lingwistyczną. Jednym z możliwych pomysłów przy wyszukiwaniu relacji pomiędzy rzeczownikami jest wykorzystanie przymiotników jako cech opisujących rzeczowniki.

Podczas przemieszczania okna o rozmiarze $\pm k$ nad tekstem została utworzona macierz współwystępowania rzeczowników z przymiotnikami. Początkowo został przeprowadzony eksperyment dla 4 157 przymiotników znajdujących się w Słowosieci. Następnie lista przymiotników została rozszerzona do wszystkich[6] 15 768 przymiotników z KIPI.

Po utworzeniu macierzy wartości komórek zostały poddane transformacji entropijnej — tak samo jak w LSA. Jako miary podobieństwa wyrazów użyto kosinusa kąta pomiędzy wierszami macierzy. Najlepszy wynik uzyskano dla rozszerzonej listy przymiotników dla okna o wielkości $k = 2$ — 75.94% (por. tabela 4.9). W tablicy P_p oznacza użycie podstawowej listy 4 157 przymiotników, P_r — rozszerzonej lista 15 768 przymiotników.

Zbadano również wpływ braku transformacji wartości komórek macierzy (wiersz *liczność*) — wyniki są o wiele gorsze, jednak porównywalne z uzyskanymi w LSA. Można

6. Użyto stop–listy do odrzucenia niektórych nieniosących dużej ilości informacji wyrazów.

zaobserwować, że w miarę zwiększania rozmiaru okna zmniejsza się dokładność. Dzieje się tak ponieważ większe okno znajduje więcej niewłaściwych powiązań przymiotników do rzeczowników.

Uzyskane wyniki są porównywalne z przedstawionymi przez Freitag et al. w [Fre05].

	$\mathbf{P_p}(k=3)$	$\mathbf{P_r}(2)$	$\mathbf{P_r}(3)$	$\mathbf{P_r}(4)$	$\mathbf{P_r}(5)$	$\mathbf{P_r}(6)$	$\mathbf{P_r}(10)$
entropia	68.11	**75.94**	70.73	68.27	66.8	65.86	63.25
liczność	50.84	59.35	56.39	56.29	56.09	55.94	55.89

Tabela 4.9. Dokładność [%] dla porównania wyrazów kosinusem kąta z macierzy współwystępowania rzeczowników i przymiotników dla różnych rozmiarów okna.

4.2.3. Kontekst oknowy — podejście wykorzystujące JOSKIPI

Opisany wyżej eksperyment ma jedną podstawową wadę — jedynym parametrem sprawdzającym czy dany przymiotnik jest powiązany z rzeczownikiem jest wielkość okna. Dla dużego okna zostanie odnotowanych wiele niepoprawnych związków. Dla małego okna — część poprawnych związków występujących w korpusie zostanie pominięta.

Dlatego należało wzbogacić metodę o kolejne informacje lingwistyczne, które umożliwiałyby sprawdzanie uzgodnienia przymiotników z rzeczownikami. Na potrzeby TaKIPI — tagera[7] dla języka polskiego — został stworzony Język Opisu Stanu Korpusu IPI PAN (JOSKIPI) [Pia06a]. Pozwala on na tworzenie operatorów za pomocą których można testować spełnianie różnych gramatycznych relacji występujących w tekście, np.:

- `equal(pos[0],pos[-2])` — sprawdzana czy wyrazy na pozycji 0 i -2 mają takie same części mowy,
- `case[0] ? equal(pos[0],subst)` — pobranie wartości przypadka jedynie dla wyrazów z kategorii gramatycznej subst,
- `agr(-4,0,cas, gnd, nmb)` — sprawdzenie czy wyrazy na pozycji -4 i 0 mogą być uzgodnione pod względem przypadka, rodzaju i liczby.

W celu poprawienia jakości metody należało opracować operator JOSKIPI, który sprawdzałby uzgodnienie przymiotnika z rzeczownikiem pod względem przypadku liczby i rodzaju. Dodatkowo operator powinien odrzucać pewne przypadki szczególne: występowanie niedozwolonych modyfikatorów, słów i konstrukcji pomiędzy rzeczownikiem a przymiotnikiem. Użyto operator zaproponowany w pracy [Pia07a]:

```
or(
and(
llook(-1,-5,$A,and(
     inter(pos[$A],{adj,pact,ppas}),
        inter(base[$A],{"^Zmienna0"}),
        agrpp(0,$A,{nmb,gnd,cas},3)
        )),//llook
    or(
```

7. Narzędzie do automatycznego ujednoznaczniania morfo–syntaktycznego.

```
        only($A,-1,$Ad, inter(pos[$Ad],{adj,adv,qub,pcon,
        pact,ppas,num,numcol})),
         and(
          not(
               llook(-1,$A,$V,and(
                in(pos[$V],{fin,praet,inf,impt}),
                 not(
                   inter(base[$V],{"być"})
                      )
                )) //llook
           ),//not
           in(cas[0],{nom,acc,dat,loc,inst,voc}),
           not(
              llook(-1,$A,$S,and(
                 inter(pos[$S],{subst,ger,depr}),
                 in(cas[$S],{nom,acc,dat,loc,inst,voc}),
                 not(
                  llook($S,$A,$P,equal(pos[$P],{prep}))
                  ),//not
              )),//llook
             ),//not
           )//and
      )//or
),//and
  and(
    rlook(1,5,$A,and(
      inter(pos[$A],{adj,pact,ppas}),
      inter(base[$A],{"^Zmienna0"}),
      agrpp(0,$A,{nmb,gnd,cas},3)
      )),//rlook
    or(
      only(1,$A,$Ad, inter(pos[$Ad],{adj,adv,qub,pcon
       ,pact,ppas,num,numcol})),
       and(
         not(
           rlook(1,$A,$P,equal(pos[$P],{prep}))
         ),//not
         not(
           rlook(1,$A,$V,in(pos[$V],{fin,praet,inf,impt})),
             not(
               inter(base[$V],{"być"})
             )
         ),//not
         in(cas[0],{nom,acc,dat,loc,inst,voc}),
         not(
           rlook(1,$A,$S,and(
```

```
                inter(pos[$S],{subst,ger,depr}),
                in(cas[$S],{nom,acc,dat,loc,inst,voc}),
                )),//rlook
            ),//not
           )//and
         )//or
      )//and
 )//or
```

Operator `llook` szuka słowa odległego o 5 pozycji spełniającego kilka ograniczeń: musi to być uzgodniony z rzeczownikiem przymiotnik o określonej formie podstawowej. Analogicznie działa operator `rlook`. W miejsce `Zmienna0` zostaje wstawiony przymiotnik w formie podstawowej. Stworzenie pełnego operatora zajęło dwa dni robocze.

	JOSKIPI+$\mathbf{P_p}$ k=5	JOSKIPI+$\mathbf{P_r}$ k=5
entropia	80.35	**81.04**
liczność	62.61	62.49

Tabela 4.10. Dokładność [%] dla porównania wyrazów kosinusem kąta z macierzy współwystępowania rzeczowników i przymiotników po wrowadzeniu operatorów JOSKIPI.

Po wprowadzeniu operatorów JOSKIPI do eksperymenty wynik zgodnie z oczekiwaniami jeszcze się poprawił — w teście WBST ma on 81.04% (por. tabela 4.10).

4.2.4. Eksperymenty z funkcjami podobieństwa

Do tej pory stosowaną miarą podobieństwa był kosinus kąta między wektorami słów. Manning i Schütze twierdzą, że lepiej użyć miar opartych na porównywaniu rozkładów prawdopodobieństw, ponieważ wystąpienia słów w korpusie zazwyczaj nie są zgodne z rozkładem normalnym (a takie założenie przyjmują miary euklidesowe) [Man01].

Zostały przeprowadzone eksperymenty dla dwóch miar opartych na porównywaniu rozkładów prawdopodobieństwa zapropnowane w [Dag97, Man01]: promień informacji (IRad, ang. *Information Radius*) przedstawiony na równaniu 4.8 i średnia L_1 (ang. L_1 *norm*) — równanie 4.9.

$$IRad(p||q) = D(p||\frac{p+q}{2}) + D(q||\frac{p+q}{2}), \tag{4.8}$$

gdzie $D(p||q) = \sum_i p_i \log \frac{p_i}{q_i}$ to dywergencja Kullbacka–Leiblera (inaczej entropia względna) [Man01].

$$L_1(p||q) = \sum_i |p_i - q_i| \tag{4.9}$$

Wektory współwystępowania słów należało zamienić na wektory $\vec{V}$ opisujące (empiryczne) rozkłady prawdopodobieństw $P(A|N)$ modyfikacji rzeczownika N przez czasownik a (równanie 4.10).

$$\vec{V_N}(a) = \frac{\vec{V_n}(a)}{\sum_i \vec{V_N}(i)} \tag{4.10}$$

Miary $IRad$ i L_1 badają jak odległe są rozkłady, należało sprowadzić je do miar podobieństwa (równania 4.11 i 4.12). Wartość parametru β zostały dobrane empirycznie. Dla Sim_{IRad} — $\beta = 10$ dla Sim_{L_1} — $\beta = 4$.

$$Sim_{IRad} = 10^{-\beta IRad(p||q)} \tag{4.11}$$

$$Sim_{L_1} = (2 - L_1(p, q))^{\beta} \tag{4.12}$$

Wyniki testu WBST dla tych FPS zostały zaprezentowane w tablicy 4.11. Najlepszy uzyskany wynik jest nieznacznie gorszy od miary opartej na kosinusie kąta. Wprowadzanie wygładzania rozkładu prawdopodobieństwa opartego na prawie LaPlace'a znacznie obniżyło dokładność funkcji podobieństwa — różnice pomiędzy atrybutami zostały zatarte.

	$P(A\|N)$	Wygładzany $P(A\|N)$
IRad	**80.16**	41.73
L_1	76.98	46.76

Tabela 4.11. Dokładność [%] probabilistycznych funkcji podobieństwa semantycznego.

4.2.5. Ręczna ocena jakości funkcji podobieństwa semantycznego

Testy synonimii oparte na Słowosieci dały możliwość oceny automatycznej i obiektywnej różnych FPS dla języka polskiego. Przydatność tego podejścia w zastosowaniach do automatycznej budowy i rozbudowy tezaurusa zostanie zbadana manualnie. W tym celu zostały wygenerowane grupy n najbardziej podobnych wyrazów dla metody wyjściowej — LSA i dla metody uzyskującej najlepsze wyniki w teście WBST.

adwokat	**jod**	**buźka**	**absencja**	**opakowanie**	**żołnierz**
sąd	uczulenie	pytanie	kontuzja	wytwórca	armia
rozprawa	organizm	odpowiedź	rezerwowy	produkt	brygada
instancja	toksyna	decyzja	dubler	receptura	wojsko
mecenas	izotop	prezydium	rywal	wyrób	dowódca
wyrok	bakteria	zarząd	kolano	konserwant	korpus

Tabela 4.12. Przykładowe wyniki porównywania wyrazów dla LSA.

W tablicy 4.12 zostały pokazane grupy najbliższych wyrazów zgodnie z funkcją podobieństwa opartą na analizie semantyki ukrytej. Zostały wybrane wyrazy dające dobre listy podobieństwa (»żołnierz«, »jod«, »adwokat«) i złe (»buźka«, »absencja«, »opakowanie«).

Wyrazy znajdujące się na dobrych listach nie tworzą jedynie relacji synonimii. Synonimem »adwokata« może być »mecenas«, jednak »sąd« czy »rozprawa« tworzą z »adwokatem« bardziej rozmytą relacji podobieństwa. Inna natomiast sytuacja występuje dla słowa »żołnierz« — na liście występują dwie relacje: meronimii (»wojsko« składa się z »żołnierzy«) i hiponimii (»dowódca« to rodzaj »żołnierza«).

Najbardziej podobne słowa do »absencji« pokazują błędy LSA. W „Dziennikach Polskich" występuje dużo artykułów sportowych. W relacjach ze spotkań piłkarskich »absencja« często jest używana w kontekście nieobecności zawodnika na boisku z powodu kontuzji. Po redukcji wymiarów wyrazy te znalazły się blisko siebie w nowym układzie współrzędnych. Została zatarta granica funkcji słowa »absencja«, natomiast uwypuklony jego kontekst. Stąd na liście pojawił się »rezerwowy«, »dubler« i »kolano«. Podobna sytuacja mogła zajść dla słowa »buźka« — z tym, że tutaj wyrazy tworzą trudną do zidentyfikowania grupę. Jest to istotnym ograniczeniem LSA, ponieważ w wypadku »absencji« można znaleźć przyczynę i poprawić jakość FPS (np. poprzez dodanie do korpusu dokumentów mówiących o nieobecności w innych kontekstach niż sportowym), to dla »buźki« dostarczenie „poprawnych" kontekstów nie jest łatwe.

adwokat	**jod**	**buźka**	**absencja**	**opakowanie**	**żołnierz**
prawnik	izotop	buzia	nieobecność	pojemnik	oficer
architekt	pierwiastek	bliźni	przerwa	pudełko	generał
dziennikarz	substancja	tata	pauza	butelka	obywatel
aktor	odpad	córcia	pobyt	puszka	armia
naukowiec	rodnik	humanitarność	zwolnienie	okno	wojsko

Tabela 4.13. Przykładowe wyniki porównywania wyrazów dla funkcji podobieństwa używającej wyrażeń JOSKIPI.

Listy podobieństw dla tych samych wyrazów zostały wydobyte przy użyciu FPS opartej na wyrażeniach JOSKIPI (tablica 4.13). Wyrazy najbardziej podobne do słowa »żołnierz« tworzą podobne z nim relacje jak w przypadku LSA. Dla wyrazu »adwokat« sytuacja się trochę poprawia. Najbardziej podobnym do niego jest »prawnik« (relacja hiponimii), kolejne wyrazy — nazwy zawodów – pełnią podobną funkcję semantyczną w tekście co »adwokat«.

Wyraźna poprawa sytuacji zachodzi dla wyrazu »opakowanie« — LSA stworzyło listę wyrazów występujących razem w szerokim kontekście, natomiast FPS oparta na JOSKIPI wykryła wyrazy będące w relacji synonimii (»pojemnik«, »pudełko«) i hiponimii (»butelka«, »puszka«). Co ciekawe w Słowosieci znajduje się synset składający się z wyrazów: »opakowanie, pojemnik, paczka, pudełko«. Dla wyrazu »jod« i »absencja« zachodzi podobna sytuacja. Pierwszy wyraz na liście podobnych do »buźka« tworzy relację synonimii (w Słowosieci występuje synset składający się z wyrazów »buzia« i »buźka«), kolejne wyrazy również są nietrafione.

Funkcja podobieństwa semantycznego nie radzi sobie z problemem *polisemii* (niektóre słowa mają więcej niż jedno znaczenie, jednak jest ono powiązane) i *homonimii* (wiele niepowiązanych ze sobą znaczeń). Dla przykładu na liście najbardziej podobnych do »agent« pojawiają się dwa znaczenia: »szpieg«, »żołnierz« i »pośrednik«, »urzędnik«, »przedstawiciel«. Podobna sytuacja występuje dla słowa »apart«: »organ«, »ad-

ministracja« i »urządzenie«, »maszyna«. Może też nastąpić inna sytuacja, np. dla słowa »zamek«, FPS wskazuje jedynie znaczenie związane z budowlą (»pałac«, »warownia«, »dwór«).

Rozdział 5

Podsumowanie

Celem pracy było dokonanie analizy porównawczej metod grupowania dokumentów i metod wydobywania leksykalnych relacji semantycznych opartych o hipotezę dystrybucyjną pod kątem ich przydatności w automatycznej ekstrakcji sieci semantycznych dla języka polskiego.

Dokumenty tekstowe można pogrupować hierarchicznie w taki sposób, że stworzona struktura stanowić będzie szkielet dla sieci semantycznej. Natomiast metody oparte na hipotezie dystrybucyjnej wydobywają leksykalne relacje semantyczne pomiędzy słowami bezpośrednio z tekstu.

Przeprowadzono testy dla dwóch metod grupowania — ROCK i GHSOM. Z eksperymentów wynika, że obydwa algorytmy mają pewne wady w zastosowaniu do stworzenia szkieletu sieci semantycznej. Pierwszy z nich ma tendencję do budowy zbyt głębokich struktur, drugi częściej błędnie grupuje dokumenty. Pomimo tych ograniczeń ręczna analiza powstałych hierarchii wskazuje na ich użyteczność, szczególnie w wykrywaniu liczby różnych sensów słów — wykrywanie polisemii może stanowić jeden z kierunków dalszych badań.

Próba opisania grup słowami reprezentatywnymi w celu zbudowania tezaurusa zakończyła się niepowodzeniem. Wydobyte słowa dobrze opisują dokumenty, jednak rzadko zachodzi pomiędzy nimi relacja hiponimii. Techniki zaczerpnięte z dziedziny automatycznego streszczania mogą okazać się bardziej skuteczne od podejścia użytego w tej pracy. Zaproponowanie metody wydobywania słów z grupy dokumentów uwzględniającej zarówno informacje statystyczne o słowach, strukturę powiązań pomiędzy dokumentami jak i wiedzę lingwistyczną dotyczącą występujących między słowami relacji może stanowić ciekawy obszar badawczy.

Punktem wyjścia do analizy metod opartych o hipotezę dystrybucyjną Harrisa pod kątem przydatności do ekstrakcji sieci semantycznej dla języka polskiego było powtórzenie eksperymentu przeprowadzonego przez Landauer et al. opisanego w pracy [Lan97]. Następnie zastosowano i wzbogacono o informacje lingwistyczne technikę *Word Space* zaproponowaną przez Schütze [Sch93, Sch98].

Do oceny *funkcji podobieństwa semantycznego* (FPS) został wykorzystany test synonimii WBST — *WordNet–Based Synonymy Test* [Fre05]. Ze względu na strukturę

Słowosieci został on rozszerzony o uwzględnianie relacji hiponimii w procesie generowania pytań.

W teście WBST otrzymano wynik 81.04%. To bardzo dobry rezultat biorąc pod uwagę, fakt że grupa 24 Polaków osiągnęła średnio 89,29%. Niestety jakości zaproponowanego podejścia nie można bezpośrednio porównywać z badaniami dla języka angielskiego (Freitag et al. uzyskali 75.8%), ponieważ dla języka polskiego została użyta wczesna wersja Słowosieci i test składa się z mniejszej o połowę liczby pytań. Dokładność zaproponowanej metody można jeszcze poprawić poprzez zastosowanie innych operatorów JOSKIPI, które dostarczałyby dodatkowych informacji potrzebnych w rozróżnianiu słów.

Funkcje podobieństwa semantycznego mogą zostać wykorzystane w automatycznym wydobywaniu sieci semantycznych z tekstu. Wcześniej jednak należy rozwiązać problem polisemii. Podczas analizy wyników zostało pokazane, że zaproponowane FPS nie są w stanie rozstrzygać pomiędzy różnymi znaczeniami słowa.

Dalsze prace nad automatyczną ekstrakcją sieci semantycznych dla języka polskiego powinny dotyczyć hybrydowego połączenia obydwóch analizowanych podejść. Metody grupowania mogą dostarczyć informacji o polisemii do FPS. Funkcja podobieństwa semantycznego jest źródłem wiedzy o relacjach występujących pomiędzy słowami, które może być wykorzystane przez metody grupowania.

Spis tablic

Spis algorytmów

Bibliografia

[Ank99] Ankerst, M., Breunig, M. M., Kriegel, H.-P., Sander, J. *Optics: ordering points to identify the clustering structure.* SIGMOD Rec., vol. 28(2), pp. 49–60, 1999. ISSN 0163-5808.

[Bee99] Beeferman, D., Berger, A., Lafferty, J. *Statistical models for text segmentation.* Mach. Learn., vol. 34(1-3), pp. 177–210, 1999. ISSN 0885-6125.

[Ber92] Berry, M. *Large scale singular value computations.* International Journal of Supercomputer Applications, vol. 6(1), pp. 13–49, 1992.

[Ber96] Berry, M. W., Do, T., O'Brien, G. W., Krishna, V., Varadhan, S. *Svdpackc (version 1.0) user guide*, 1996.

[Ber03] Berry, M. W. *Survey of Text Mining.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2003. ISBN 0387955631.

[Cho01] Choi, F., Wiemer-Hastings, P., Moore, J. *Latent semantic analysis for text segmentation.* W Proceedings of 6th EMNLP, pp. 109–117, 2001. URL: `citeseer.ist.psu.edu/choi01latent.html`.

[Coh60] Cohen, J. *A coefficient of agreement for nominal scales.* Educational and Psychological Measurement, vol. 20, pp. 3–46, 1960.

[Dag97] Dagan, I., Lee, L., Pereira, F. *Similarity-based methods for word sense disambiguation.* W Proceedings of the Thirty-Fifth Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics, (P. R. Cohen, W. Wahlster, eds.), pp. 56–63, Association for Computational Linguistics, Somerset, New Jersey, 1997. URL: `citeseer.ist.psu.edu/dagan97similaritybased.html`.

[Dee90] Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., Harshman, R. A. *Indexing by latent semantic analysis.* Journal of the American Society of Information Science, vol. 41(6), pp. 391–407, 1990. URL: `citeseer.ist.psu.edu/deerwester90indexing.html`.

[Dee06] Deepak, P., Roy, S. *Optics on text data: Experiments and test results.* Raport tech., IBM India Research Lab, 2006.

[Der07a] Derwojedowa, M., Piasecki, M., Szpakowicz, S., Zawisławska, M. *Polish WordNet on a shoestring.* W Proceedings of Biannual Conference of the Society for Computational Linguistics and Language Technology, Tübingen, April 11–13 2007, pp. 169–178, Universität Tübingen, 2007.

[Der07b] Derwojedowa, M., Zawisławska, M., Piasecki, M., Szpakowicz, S. *Relacje w polskim WordNecie (wnpl).* Raporty Serii PREPRINTY 1, Instytut Informatyki Stosowanej, Politechnika Wrocławska, 2007. URL: `http://plwordnet.pwr.wroc.pl/main/content/files/publications/relacje_v5rc02.pdf`.

[Dit00] Dittenbach, M., Merkl, D., Rauber, A. *Using Growing Hierarchical Self-Organizing Maps for Document Classification.* W Proc of the European Symposium on Artificial Neural Networks (ESANN 2000), pp. 7–12, D-Facto Publications, Bruges, Belgium, 2000. URL: `citeseer.ist.psu.edu/dittenbach00using.html`.

[Duf89] Duff, I. S., Grimes, R. G., Lewis, J. G. *Sparse matrix test problems.* ACM Trans. Math. Softw., vol. 15(1), pp. 1–14, 1989. ISSN 0098-3500.

[Est96] Ester, M., Kriegel, H.-P., Sander, J., Xu, X. *A density-based algorithm for discovering clusters in large spatial databases with noise.* W Second International Conference on Knowledge Discovery and Data Mining, (E. Simoudis, J. Han, U. Fayyad, eds.), pp. 226–231, AAAI Press, Portland, Oregon, 1996. URL: `citeseer.ist.psu.edu/chu02incremental.html`.

[Fel98] Fellbaum, C., ed.. *WordNet — An Electronic Lexical Database.* The MIT Press, 1998.

[For06] Forster, R. *Document Clustering in Large German Corpora Using Natural Language Processing.* Praca doktorska, University of Zurich, 2006.

[Fre05] Freitag, D., Blume, M., Byrnes, J., Chow, E., Kapadia, S., Rohwer, R., Wang, Z. *New experiments in distributional representations of synonymy.* W Proceedings of the 9th Conference on Computational Natural Language Learning, pp. 25–32, ACL, 2005.

[Gal03] Galley, M., McKeown, K. R., Fosler-Lussier, E., Jing, H. *Discourse segmentation of multi-party conversation".* W Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL–03), (E. Hinrichs, D. Roth, eds.), pp. 562–569, Sapporo, Japan, 2003. URL: `http://www1.cs.columbia.edu/~galley/papers/mtgseg.pdf`.

[Gol98] Goldszmidt, M., Sahami, M. *A probabilistic approach to full–text document clustering.* Raport tech., SRI International, 1998.

[Gre93] Grefenstette, G. *Evaluation techniques for automatic semantic extraction: Comparing syntactic and window-based approaches.* Raport tech., Department of Computer Science, University of Pittsburgh, 1993.

[Guh98] Guha, S., Rastogi, R., Shim, K. *Cure: an efficient clustering algorithm for large databases.* W Proceedings of the 1998 ACM SIGMOD international conference on Management of data, pp. 73–84, 1998. URL: `citeseer.ist.psu.edu/guha98cure.html`.

[Guh00] Guha, S., Rastogi, R., Shim, K. *Rock: A robust clustering algorithm for categori-*

cal attributes. Information Systems, vol. 25(5), pp. 345–366, 2000. URL: citeseer.ist.psu.edu/guha00rock.html.

[Hea92] Hearst, M. A. *Automatic acquisition of hyponyms from large text corpora*. Raport tech. S2K-92-09, 1992. URL: citeseer.ist.psu.edu/hearst92automatic.html.

[Hea93] Hearst, M. A. *Texttiling: A quantitative approach to discourse segmentation*. Raport tech., Berkeley, CA, USA, 1993.

[Hea97] Hearst, M. A. *TextTiling: Segmenting text into multi-paragraph subtopic passages*. Computational Linguistics, vol. 23(1), pp. 33–64, 1997. URL: http://www.ischool.berkeley.edu/~hearst/papers/cl-texttiling97.pdf.

[Hun05] Hung, C., Wermter, S. *A constructive and hierarchical self-organizing model in a non-stationary environment*. W Neural Networks, 2005.

[IP04] Indyka-Piasecka, A. *Modele użytkownika w internetowych systemach wyszukiwania informacji*. Praca doktorska, Politechnika Wrocławska, 2004.

[Jai99] Jain, A. K., Murty, M. N., Flynn, P. J. *Data clustering: a review*. ACM Computing Surveys, vol. 31(3), pp. 264–323, 1999. URL: citeseer.ist.psu.edu/jain99data.html.

[Kar01] Karlgren, J., Sahlgren, M. *From words to understanding*. W Foundations of Real-World Intelligence. 2001. ISBN 1575863383. URL: http://eprints.sics.se/131/01/KarlgrenSahlgren2001.pdf.

[Klo06] Klopotek, M. A., Ciesielski, K., Czerski, D., Draminski, M., Wierzchon, S. T. *Beatca: Map-based intelligent navigation in www*. W Artificial Intelligence: Methodology, Systems, and Applications, 12th International Conference, AIMSA 2006„ pp. 245–254, 2006.

[Koh00] Kohonen, T., Kaski, S., Lagus, K., Salojrvi, J., Honkela, J., Paatero, V., Saarela, A. *Self organization of a massive document collection*. IEEE Transactions on Neural Networks„ vol. 11, pp. 574–585, 2000.

[Kon06] Konchady, M. *Text Mining Application Programming*. Charles River Media, 2006.

[Kor05] Koronacki, J., Ćwik, J. *Statystyczne systemy uczące się*. Wydawnictow Naukowo-Techniczne, 2005.

[Lag00] Lagus, K. *Text Mining with WEBSOM*. Praca doktorska, Helsinki University of Technology, 2000.

[Lan97] Landauer, T., Dumais, S. *A solution to Plato's problem: The latent semantic analysis theory of acquisition*. Psychological Review, vol. 104(2), pp. 211–240, 1997.

[Lik01] Likas, A., Vlassis, N., Verbeek, J. *The global k-means clustering algorithm*. Raport tech., Computer Science Institute, University of Amsterdam, The Netherlands, 2001. IAS-UVA-01-02, URL: citeseer.ist.psu.edu/article/likas01global.html.

[Lin97] Lin, D. *Using syntactic dependency as local context to resolve word sense ambiguity*. W Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association

for Computational Linguistics (ACL-97), pp. 64–71, ACL, Madrid, Spain, 1997. URL: `http://acl.ldc.upenn.edu/P/P97/P97-1009.pdf`.

[Lin98] Lin, D. *Automatic retrieval and clustering of similar words.* W COLING 1998, pp. 768–774, ACL, 1998. URL: `http://acl.ldc.upenn.edu/P/P98/P98-2127.pdf`.

[Man01] Manning, C. D., Schütze, H. *Foundations of Statistical Natural Language Processing.* The MIT Press, 2001.

[Man07] Manning, C. D., Raghavan, P., Schütze, H. *Introduction to Information Retrieval.* Cambridge University Press, 2007. To appear.

[Mat04] Matsuo, Y., Ishizuka, M. *Keyword extraction from a single document using word co-occurrence statistical information.* International Journal on Artificial Intelligence Tools, vol. 13(1), pp. 157–169, 2004.

[Maz05] Mazur, D. *Metody grupowania i ich implementacja do eksploracji danych postaci symbolicznej.* Praca doktorska, Politechnika Śląska, 2005.

[Ng94] Ng, R. T., Han, J. *Efficient and effective clustering methods for spatial data mining.* W VLDB'94, Proceedings of 20th International Conference on Very Large Data Bases, September 12-15, 1994, Santiago de Chile, Chile, (J. B. Bocca, M. Jarke, C. Zaniolo, eds.), pp. 144–155, Morgan Kaufmann, 1994. ISBN 1-55860-153-8.

[Oso96] Osowski, S. *Sieci neuronowe w ujęciu algorytmicznym.* WNT, Warszawa, 1996.

[Pal04] Pal, S. K., Mitra, P. *Pattern Recognition Algorithms for Data Mining: Scalability, Knowledge Discovery, and Soft Granular Computing.* Chapman & Hall, Ltd., London, UK, UK, 2004. ISBN 1584884576.

[Pia] Piasecki, M. *Cele i zadania lingwistyki informatycznej.* Ukaże się w red. P. Stalmaszczyk, Metodologie jezykoznawstwa. Wspolczesne tendencje i kontrowersje. 2007.

[Pia06a] Piasecki, M. *Handmade and automatic rules for Polish tagger.* 2006.

[Pia06b] Piasecki, M. *Wykłady z przedmiotu „wydobywanie i wyszukiwanie informacji z internetu", politechnika wrocławska*, Politechnika Wrocławska, 2006.

[Pia06c] Piasecki, M., Godlewski, G. *Effective architecture of the Polish tagger.* 2006.

[Pia06d] Piasecki, M., Godlewski, G. *Reductionistic, Tree and Rule Based Tagger for Polish.* 2006.

[Pia07a] Piasecki, M., Broda, B. *Semantic similarity measure of Polish nouns based on linguistic features.* W Business Information Systems 10th International Conference, BIS 2007, Poznan, Poland, (W. Abramowicz, ed.), *Lecture Notes in Computer Science*, vol. 4439, Springer, 2007.

[Pia07b] Piasecki, M., Derwojedowa, M., Koczan, P., Przepiórkowski, A., Szpakowicz, S., Zawisławska, M. *Pólautomatyczna konstrukcja Słowosieci*, 2007. Strona domowa projektu, URL: `http://plwordnet.pwr.wroc.pl/`.

[Pia07c] Piasecki, M., Szpakowicz, S., Broda, B. *Automatic selection of heterogeneous syntactic features in semantic similarity of polish nouns.* W Proceedings of the Text, Speech and Dialogue Conference, 2007.

[Poi98] Pointcot, P., Lesteven, S., Murtagh, F. *A spatial user interface to astronomical literature.* Astronomy and Astrophysics Supplement Series, vol. 130, pp. 183–191, 1998.

[Pon97] Ponte, J. M., Croft, W. B. *Text segmentation by topic.* W ECDL '97: Proceedings of the First European Conference on Research and Advanced Technology for Digital Libraries, pp. 113–125, Springer-Verlag, London, UK, 1997. ISBN 3-540-63554-8.

[Prz04] Przepiórkowski, A. *Korpus IPI PAN. Wersja wstępna.* Instytut Podstaw Informatyki PAN, 2004.

[Rau02] Rauber, A., Merkl, D., Dittenbach, M. *The growing hierarchical self-organizing maps: exploratory analysis of high-dimensional data*, 2002. URL: `citeseer.ist.psu.edu/rauber02growing.html`.

[Rug92] Ruge, G. *Experiments on linguistically-based term associations.* Information Processing and Management, vol. 28(3), pp. 317–332, 1992.

[Sal75] Salton, G., Wong, A., Yang, C. S. *A vector space model for automatic indexing.* Communications of the ACM, vol. 18, p. 613–620, 1975.

[San98] Sander, J., Ester, M., Kriegel, H.-P., Xu, X. *Density-based clustering in spatial databases: The algorithm gdbscan and its applications.* Data Min. Knowl. Discov., vol. 2(2), pp. 169–194, 1998. ISSN 1384-5810.

[Sch93] Schütze, H. *Word space.* W Advances in Neural Information Processing Systems 5, (S. Hanson, J. Cowan, C. Giles, eds.). Morgan Kaufmann Publishers, 1993. URL: `citeseer.ist.psu.edu/schutze93word.html`.

[Sch98] Schütze, H. *Automatic word sense discrimination.* Computational Linguistics, vol. 24(1), pp. 97–123, 1998. URL: `citeseer.ist.psu.edu/schutze98automatic.html`.

[Sch05] Schenker, A., Bunke, H., Last, M., Kandel, A. *Graph-Theoretic Techniques for Web Content Mining.* World Scientific Publishing Co. Pte. Ltd., 2005.

[Son06] Song, S., Li, C. *Improved rock for text clustering using asymmetric proximity.* W SOFSEM 2006: Theory and Practice of Computer Science, *Lecture Notes in Computer Science*, vol. 3831, 2006.

[Tur00] Turney, P. D. *Learning algorithms for keyphrase extraction.* Information Retrieval, vol. 2(4), pp. 303–336, 2000. URL: `citeseer.ist.psu.edu/turney00learning.html`.

[Tur01] Turney, P. *Mining the web for synonyms: Pmi-ir versus lsa on toefl.* W Proceedings of the Twelfth European Conference on Machine Learning, pp. 491–502, Springer-Verlag, Berlin, 2001.

[Tur02] Turney, P. *Mining the web for lexical knowledge to improve keyphrase extraction: Learning from labeled and unlabeled data*, 2002. URL: `citeseer.ist.psu.edu/turney02mining.html`.

[Tur03] Turney, P., Littman, M., Bigham, J., Shnayder, V. *Combining independent modules to solve multiple-choice synonym and analogy problems.* W Proceedings of

the International Conference on Recent Advances in Natural Language Processing, 2003.

[Vos02] Vossen, P. *EuroWordNet general document version 3.* Raport tech., University of Amsterdam, 2002.

[Wee03] Weeds, J., Weir, D. *A general framework for distributional similarity.* W Proceedings of the Conference on Empirical Methods in Natural Language Processing, 2003. URL: `http://www.informatics.sussex.ac.uk/users/davidw/papers/emnlp03.pdf`.

[Wee05] Weeds, J., Weir, D. *Co-occurrence retrieval: A flexible framework for lexical distributional similarity.* Computational Linguistics, vol. 31(4), pp. 439–475, 2005.

[Wei06] Weiss, D. *Descriptive Clustering as a Method for Exploring Text Collections.* Praca doktorska, Politechnika Poznańska, 2006.

[Wid04] Widdows, D. *Geometry and Meaning.* CSLI Publications, 2004.

[Wol06] Woliński, M. *Morfeusz — a practical tool for the morphological analysis of Polish.* 2006.

[Wu04] Wu, W., Xiong, H., Shekhar, S., eds.. *Clustering and Information Retrieval.* Kulwer Academic Publishers, 2004.

[Xu98] Xu, X., Ester, M., Kriegel, H.-P., Sander, J. *A distribution-based clustering algorithm for mining in large spatial databases.* W ICDE '98: Proceedings of the Fourteenth International Conference on Data Engineering, pp. 324–331, IEEE Computer Society, Washington, DC, USA, 1998. ISBN 0-8186-8289-2.

[Zha96] Zhang, T., Ramakrishnan, R., Livny, M. *Birch: an efficient data clustering method for very large databases.* W In Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, pp. 103–114, 1996. URL: `citeseer.ist.psu.edu/zhang96birch.html`.