STUDIA INFORMATICA 2010
Volume 31 Number 2A (89)

Agnieszka NOWAK – BRZEZIŃSKA, Tomasz JACH, Tomasz XIĘSKI
Uniwersytet Śląski, Wydział Informatyki i Nauki o Materiałach, Instytut Informatyki

WYBÓR ALGORYTMU GRUPOWANIA A EFEKTYWNOŚĆ WYSZUKIWANIA DOKUMENTÓW

Streszczenie. Praca przedstawia wyniki wstępnych eksperymentów dotyczących grupowania dokumentów tekstowych przy użyciu k-optymalizacyjnych, hierarchicznych oraz gęstościowych algorytmów analizy skupień. Eksperymenty wykonane dla rzeczywistych zbiorów dokumentów (a właściwie ich charakterystyk) potwierdzają fakt, że wybór algorytmu grupowania ma ogromny wpływ na efektywność (kompletność i dokładność) wyszukiwania informacji w strukturze skupień dokumentów.

Słowa kluczowe: grupowanie dokumentów tekstowych, kompletność, dokładność, algorytmy gęstościowe

CHOOSING THE CLUSTERING ALGORITHM AND SEARCHING CLUSTERS OF DOCUMENTS EFFICIENCY

Summary. The article presents the results of efficiency of searching relevant documents In the document clusters structure. The structure depends on the choosed clustering algorithm. In the experiments we used nonhierarchical, hierarchical and den sity based clustering algortihms.

Keywords: documents clustering, recall, precision, density based algorithm

1. Wprowadzenie

Problem grupowania dokumentów tekstowych (reprezentacja dokumentów w analizowanym zbiorze jest dość specyficzna, mając formę krótkiej charakterystyki dokumentu w postaci wektora słów kluczowych) nie jest trywialny. Dotyka on bowiem efektywności wyszukiwania dokumentów relewantnych względem zapytania użytkownika w strukturze grup dokumentów. Celem badań towarzyszących niniejszej pracy była analiza różnych algorytmów grupowania dla dużego zbioru dokumentów ale takich, które pozwolą nie tylko zbudować

optymalne struktury grup dokumentów, ale także efektywnie wyszukiwać z takich złożonych struktur informacje. W pracy [4] autorzy przedstawili wyniki eksperymentów, wykonanych na tym samym zbiorze dokumentów, ale grupowanych przy użyciu klasycznych algorytmów analizy skupień: k-optymalizacyjnych jak i hierarchicznych. Z tych pierwszych analizowano efektywność grupowania dokumentów tekstowych przy użyciu algorytmu k-medoidów, zaś wśród algorytmów hierarchicznych podstawą analizy był algorytm aglomeracyjny – AHC. Nawiązując krótko do wyników tej pracy powiemy, że jeśli opis dokumentu (owa krótka, złożona z kilku słów kluczowych, charakterystyka dokumentu) jest zbyt krótki i słowa opisujące dokument są źle dobrane, a ponadto niewłaściwa jest użyta metryka podobieństwa bądź odległości, będąca podstawą algorytmów grupowania, wówczas wyniki grupowania nigdy nie będą satysfakcjonujące. Znacznie lepsze parametry efektywności wyszukiwania informacji z takich złożonych struktur, ale także i parametry oceny jakości tworzonych grup dokumentów, uzyskano przy wykorzystaniu algorytmów hierarchicznych. Należy jednak zauważyć, iż wciąż wyniki te nie były optymalne. Zastosowanie algorytmów grupujących na dużych zbiorach danych pociąga za sobą ściśle określone wymagania stawiane tym algorytmom. Są to: wymagana minimalna wiedza na temat dziedziny przedmiotowej, by określić parametry wejściowe dla algorytmu, odkrywanie skupień o dowolnych kształtach, relatywnie duża szybkość działania (niska złożoność obliczeniowa) w przypadku dużych zbiorów danych oraz niewrażliwość na pytania złożone z rozłącznych słów kluczowych. Klasyczne algorytmy analizy skupień niestety najczęściej nie są w stanie realizować wszystkich tym wymogów, co najczęściej prowadzi do stosowania innych algorytmów. Przedmiotem analiz w kontekście grupowania i wyszukiwania informacji w dokumentach tekstowych stały się algorytmy (szeroko ostatnio rozwijane dla danych numerycznych) gęstościowe, cieszące się dość dużą popularnością.

2. Algorytmy gęstościowe

Algorytmy gęstościowe (ang. *density-based algorithms*) są jednymi z najefektywniejszych algorytmów analizy skupień. Podstawowe cechy, które odróżniają je od innych algorytmów, to: możliwość odnajdywania skupień o dowolnych kształtach, odporność na szum informacyjny oraz stosunkowo mała złożoność obliczeniowa. Powodem zainteresowania autorów niniejszej pracy algorytmami tego typu jest jednak przede wszystkim nadzieja na rozwiązanie problemów z efektywnością wyszukiwania informacji w strukturze grup utworzonych przez wcześniej analizowane algorytmy.

2.1. Analiza gęstości dokumentów

Gęstość dokumentów odpowiada naturalnie pojmowanemu podobieństwu między nimi. Nie można w tym momencie nie wspomnieć o znanej i dobrze przyjmowanej metodzie organizacji obiektów (również dokumentów) w postaci tzw. *mapy samoorganizującej się* [2]. Dwa punkty umieszczone w bliskiej odległości od siebie na mapie świadczą o dużym podobieństwie między dokumentami (przez te punkty reprezentowanymi). Większe odległości tych samych punktów będą odpowiadały sytuacji, w której dokumenty są do siebie już mniej podobne. Niewątpliwie możemy się tu posłużyć pojęciem gęstości punktów (dokumentów) na mapie. Metody gęstościowe będą zatem miały na celu znajdowanie grup punktów (dokumentów) gęsto ułożonych. Oczywiście optymalna sytuacja jest taka, w której utworzone grupy punktów (dokumentów) będą nie tylko cechować się dużym podobieństwem wewnętrznym dokumentów (dużą gęstością punktów), ale również małym podobieństwem międzygrupowym (czyli jak największą odległością grup punktów od siebie).

Definicja *skupienia* oparta jest tutaj na obiektach (dokumentach) wzajemnie osiągalnych lub połączonych z pewną zadaną gęstością (o parametrach *Eps* i *MinPts*). Punkt bezpośrednio osiągalny z zadaną gęstością z innego punktu to taki, który znajduje się w *Eps*-sąsiedztwie tego punktu, w którym w sumie znajduje się co najmniej *MinPts* takich punktów.

Każde dwa obiekty należące do skupienia są połączone wzajemnie z zadaną gęstością (łączność) oraz wszystkie punkty osiągalne z punktów leżących wewnątrz skupienia także należą do skupienia (maksymalność). Zatem s*ąsiedztwem Eps* obiektu *p* (oznaczanym jako *NEps*(*p*)) określimy obszar, do którego będzie należał każdy taki punkt *q* ($q \in D$ jest oczywiście punktem ze zbioru *D)*, którego odległość od punktu *p (*a więc funkcja *dist(p,q))* jest nie większa niż *Eps,* co można zapisać następująco: $NEps(p) = q \in D | dist(p,q) \le Eps$. Podejście mówiące o tym, że w sąsiedztwie *Eps* danego obiektu znajdowała się co najmniej minimalna liczba (*MinPts*) obiektów, jest jednak obarczone dużym błędem. Mianowicie, obiekty (punkty w grupie) mogą wystąpić wewnątrz skupienia, blisko jego jądra (tzw. *core points*) oraz na granicy skupienia (tzw. *border points*). Generalnie sąsiedztwo *Eps* obiektu krańcowego zawiera wyraźnie mniej obiektów, aniżeli sąsiedztwo *Eps* obiektu wewnętrznego. Zatem, wartość parametru minimalnej liczby obiektów musiałaby być relatywnie mała, by uwzględnić wszystkie punkty wchodzące w skład danego skupienia. Niestety, taka wartość nie byłaby właściwa w odniesieniu do każdej grupy, zwłaszcza w przypadku występowania wartości izolowanych. Wymaga się zatem, by dla każdego obiektu *p* w skupieniu *C* istniał taki obiekt *q* należący do *C*, żeby *p* znajdował się w sąsiedztwie *Eps q* oraz *NEps*(*q*) zawierało co najmniej *MinPts* obiektów [1, 8].

2.2. Problem wyznaczania parametrów Eps i MinPts

Aby wyznaczyć parametry *Eps* i *MinPts*, można posłużyć się prostą heurystyką (dla najmniej zagęszczonego skupienia). Jeśli *d* będzie odległością obiektu *p* do jego *k*-tego najbliższego sąsiada, *d*-sąsiedztwem obiektu *p* będzie dokładnie *k* +1 obiektów. Dla danego *k* można zdefiniować funkcję *k-dist*, która odwzorowuje każdy obiekt w bazie danych na odległość do jego *k*-tego najbliższego sąsiada. Porządkując malejąco wartości *k-dist* otrzymamy przybliżony rozkład gęstości. Wówczas, wybierając losowo obiekt *p* i ustawiając wartość parametru *Eps* na *k-dist*(*p*), a wartość parametru *MinPts* na *k*, wszystkie obiekty z mniejszą lub równą wartością *k-dist* staną się *obiektami wewnętrznymi*. Określając zatem punkt progowy (ang. *threshold point*) z maksymalną wartością *k-dist* w najmniej zagęszczonym skupieniu, otrzymamy poszukiwane wartości *Eps* i *MinPts*. Tym punktem progowym jest pierwszy punkt w pierwszej dolinie znalezionej na posortowanym wykresie *k-dist.* Wszystkie obiekty położone na lewo do punktu progowego są uznawane za *szum*, natomiast wszystkie pozostałe obiekty są przypisywane do jednej z grup.

2.3. Algorytm grupowania gęstościowego DBSCAN

Jednym z najefektywniejszych algorytmów gęstościowych jest algorytm DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*), zaproponowany w 1996 roku. Do poprawnego działania DBSCAN wymaga zdefiniowania dwóch parametrów: maksymalnego promienia sąsiedztwa (*Eps*) oraz minimalnej liczby obiektów (punktów), jaka jest wymagana do utworzenia skupienia (*MinPts*). W najlepszym przypadku musielibyśmy wiedzieć, jak dobrze określić wartości tych parametrów dla każdego możliwego skupienia. To pozwoli otrzymać wszystkie obiekty (punkty), które są gęstościowo osiągalne z zadanego zbioru obiektów (punktów), przy użyciu znanych parametrów. Niestety, nie da się w prosty sposób odgadnąć, czy wyliczyć wartości dla *Eps* i *MinPts*. Istnieje jednak prosta i efektywna heurystyka (poświęcamy temu punkt 2.2. pracy), która pozwala określić szukane parametry.

Etapy działania algorytmu DBSCAN można przedstawić w kilku punktach:

1. *Dowolny wybór punktu p*
2. *Znalezienie wszystkich punktów „osiągalnych gęstościowo" opierając się na parametrach Eps oraz MinPts*
   a) *jeśli p jest punktem „centrum", to formowany jest klaster;*
   b) *jeśli p jest punktem „granicznym" i żaden z punktów nie jest „osiągalny gęstościowo" z punktu p, wtedy algorytm DBSCAN przechodzi do następnego punktu w zbiorze danych*

3. *Proces jest kontynuowany do momentu przeanalizowania wszystkich elementów.*

Punkt *p* jest *bezpośrednio osiągalny gęstościowo* z punktu *q*, gdy *p* znajduje się w sąsiedztwie punktu *q* i *q* jest punktem rdzeniowym, czyli jego sąsiedztwo zawiera co najmniej *MinPts* punktów. Natomiast punkt *p* jest *osiągalny gęstościowo* z punktu *q*, gdy istnieje taki łańcuch punktów $p_1,...,p_n$, gdzie $p_1 = p$ a $p_n = q$, że para sąsiednich punktów jest bezpośrednio osiągalna gęstościowo. Pierwszym krokiem algorytmu jest wylosowanie obiektu *p* oraz wyznaczenie wszystkich obiektów, które są gęstościowo osiągalne z obiektu *p* (przy zadanych wartościach *Eps* i *MinPts*). Jeżeli *p* jest obiektem wewnętrznym, to krok ten skutkuje powstaniem pierwszej grupy. Jeżeli *p* jest obiektem krańcowym, to żaden obiekt nie jest gęstościowo osiągalny z *p*, więc algorytm wybiera kolejny obiekt ze zbioru danych. Proces ten jest powtarzany, aż nie zostaną przeanalizowane wszystkie obiekty ze zbioru danych wejściowych. Obiekty nie zakwalifikowane do żadnego skupienia są oznaczane jako szum informacyjny.

2.4. Zalety i wady algorytmu DBSCAN

Do największych zalet algorytmu DBSCAN (jak i wszystkich opartych na pojęciu gęstości) należy możliwość znajdywania skupień o różnych (często nieregularnych) kształtach oraz prostota jego implementacji, niska złożoność obliczeniowa i pamięciowa. Nie bez znaczenia jest także odporność na wartości izolowane (szum informacyjny). Ponadto, DBSCAN w przeciwieństwie do algorytmów k-optymalizacyjnych nie wymaga wcześniejszego określenia liczby skupień. Wadą algorytmu jest zaś trudność w klasyfikacji zbiorów obiektów o różnych gęstościach oraz konieczność określenia parametrów *MinPts* i *Eps*.

3. Efektywność mierzona kompletnością i dokładnością

W dziedzinie systemów wyszukiwania informacji miarą wydajności są parametry kompletności i dokładności odpowiedzi takich systemów na zadane przez użytkowników pytania. W klasycznym ujęciu (proponowanym m.in. w pracach [6],[10]) kompletnością nazywa się zdolność systemu do wyszukiwania dokumentów relewantnych, zaś dokładnością zdolność do niewyszukiwania dokumentów nierelewantnych. Przenosząc jednak proces wyszukiwania informacji na pytania zadawane wyszukiwarkom internetowym powiemy, że fakt, iż system nie wyszuka wszystkich możliwych dokumentów relewantnych, nie jest aż tak istotny. Nie będzie zatem tak ważny, gdy kompletność nie będzie pełna. Z dokładnością jest nieco inaczej. W obszarze Internetu, a także wszystkich innych systemach wyszukiwania informacji opartych na tzw. niepełnym przeszukiwaniu systemu, nie można liczyć na pełną dokładność. Zwykle w grupie określonej jako grupa dokumentów relewantnych, zwróconej użytkowni-

kom jako wyniki wyszukiwania, obok dokumentów relewantnych znajdą się niestety również dokumenty nierelewantne. Nie jest to oczywiście regułą i są metody pozwalające budować takie struktury, w których w efekcie nie będzie jednocześnie dokumentów relewantnych i nierelewantnych względem zadanego pytania. Jednak możemy powiedzieć, że celem będzie uzyskiwanie jak najwyższej wartości dokładności. Mówiąc o systemach opartych na tzw. niepełnym przeszukiwaniu mamy na myśli np. system budowany przez autorów pracy, którego celem jest grupowanie dokumentów tekstowych przy użyciu różnych metod analizy skupień: k-optymalizacyjnych, hierarchicznych oraz gęstościowych.

Proces wyszukiwania odpowiedzi na pytania ma na celu wyszukanie dokumentów tekstowych odpowiadających słowom kluczowym wprowadzonym jako zapytanie użytkownika. Wyszukiwanie odpowiedzi sprowadza się do przeglądnięcia, utworzonej przez wybrany algorytm analizy skupień, struktury dokumentów. Można zatem zauważyć, że za każdym razem – gdy wybierzemy inną metodę analizy skupień – inna będzie utworzona struktura dokumentów, a więc inna musi być metoda jej przeszukiwania.

Krystalizując naszą wiedzę w tym zakresie powiemy, że gdy do budowy grup dokumentów użyjemy algorytmów k-optymalizacyjnych, a konkretnie algorytmu k-medoidów, wówczas w efekcie uzyskamy k grup. Musimy mieć świadomość jednak tego, że konieczność zbudowania ustalonej z góry liczby grup może w pesymistycznym przypadku, np. rozdzielić dokumenty do siebie podobne w dwie osobne grupy, podczas gdy w rzeczywistości powinny one budować jedną spójną grupę. To może w efekcie prowadzić do niepełnej kompletności.

W przypadku algorytmów hierarchicznych przedmiotem analiz był algorytm AHC, przy czym w odróżnieniu od klasycznego podejścia budującego dendrogram w postaci drzewa binarnego i jego przeszukiwania typowego dla drzew binarnych autorzy pracy zaproponowali inne podejście, przeszukujące drzewo do określonego poziomu. Celem jest znalezienie grupy odpowiednio relewantnej do podanego przez użytkownika pytania, co najczęściej sprowadza się do wyszukania grupy dokumentów odpowiednio pasujących do słów kluczowych pytania. Może się zdarzyć więc tak, że w znalezionej grupie są zarówno dokumenty relewantne, jak i te nierelewantne. Wszystko zależy od podobieństwa dokumentów wewnątrz danej grupy.

Ostatnim z analizowanych algorytmów był algorytm z grupy algorytmów gęstościowych, a konkretnie DBSCAN. Wyższość tego algorytmu nad pozostałymi podlegającymi analizie polega na tym, że buduje on naturalne grupy, tj. łączy ze sobą takie dokumenty, które spełniają odpowiednie kryterium podobieństwa i gęstości. Póki więc analizowany dokument jest podobny do dokumentów w danej grupie, jest do niej dołączany. Sposób rozłożenia dokumentów w grupach zależy wówczas od tego, jak podobne są one do siebie, może się zatem zdarzyć tak, że utworzonych będzie wiele grup mało licznych bądź z kolei mało grup, ale bardzo licznych.

4. System grupowania dokumentów i wyszukiwania informacji

Inspiracją do stworzenia systemu grupującego i wyszukującego dokumenty relewantne był rozpowszechniony i uznany system SMART Saltona, w którym dokumenty grupowano na podstawie ich podobieństwa między sobą [7]. Powstałą w ten sposób strukturę, na czele z reprezentantami grup, przeszukiwano w dużo krótszym czasie w stosunku do przeszukiwania liniowego całego zbioru dokumentów. W początkowym etapie do budowy grup dokumentów użyto metod k-optymalizacyjnych oraz hierarchicznych. Niektóre eksperymenty wykonane w ramach tych metod opublikowano m.in. w pracach [4] oraz [3] i [9]. Ponieważ metody te dla wyszukiwania dokumentów relewantnych względem pytań zadawanych przez użytkowników nie zdawały rezultatu, w drugim etapie prac zajęto się metodami opartymi na gęstości dokumentów. Pojawiła się bowiem szansa, że właściwość metod gęstościowych dążąca do budowy naturalnych skupień obiektów, rozwiąże problemy specyficzne dla metod k-optymalizacyjnych i hierarchicznych.

4.1. Baza dokumentów

Biorąc pod uwagę wszystkie przedstawione wcześniej aspekty jako sposób reprezentacji wiedzy wybrano reprezentację za pomocą cech nominalnych. Dany dokument (w chwili obecnej baza danych zawiera 360 dokumentów opisanych za pomocą 407 słów kluczowych) reprezentowany był w systemie jako wektor cech charakterystycznych dla konkretnego dokumentu (przypomnijmy że dokumentami były prace dyplomowe uczelni z kilku lat, z zakresu informatyki), takie jak: identyfikator pracy oraz identyfikatory poszczególnych słów kluczowych z odpowiadającego im słownika terminów. Wybraną ostatecznie metryką podobieństwa dla algorytmu k-optymalizacyjnego była miara Simple Matching Coefficient (SMC). Podobieństwo danego dokumentu do innego zostało określone zatem jako stosunek liczby cech wspólnych (liczby wspólnych słów kluczowych) do liczby wszystkich cech, jakimi opisane są te dokumenty. Takie podejście do mierzenia podobieństwa obiektów uwzględniało najlepiej (spośród przetestowanych miar) różnice między obiektami zakodowanymi za pomocą opisanej reprezentacji wiedzy [9].

4.2. Dotychczasowe wyniki eksperymentów

Głównym celem przeprowadzonych badań było porównanie hierarchicznych algorytmów analizy skupień (reprezentowanych przez algorytm AHC) z *k*-optymalizacyjnymi (których reprezentantem jest k-medoidów) oraz analiza ich efektywności na zaprezentowanym specyficznym zestawie danych. W przypadku algorytmu *k*-medoidów liczba grup była ustawiona na stałą wartość równą czterdzieści.

Porównanie efektywności obu algorytmów odbyło się na podstawie współczynników kompletności i dokładności wyszukiwania. Jeśli założymy, że dany dokument uznamy za relewantny względem zadanego zapytania, gdy zawiera co najmniej jedno podane (przez użytkownika) słowo kluczowe, wówczas będziemy mogli zdefiniować pojęcia kompletności oraz dokładności odpowiedzi. Kompletność rozumiana będzie wtedy jako stosunek liczby relewantnych, wyszukanych dokumentów, do wszystkich relewantnych do zapytania dokumentów zawartych w bazie danych, zaś dokładność jako stosunek liczby wyszukanych, relewantnych dokumentów do wszystkich wyszukanych prac.

Przeprowadzone w pierwszym etapie badania pozwoliły wywnioskować, iż algorytmy hierarchiczne lepiej sprawują się na dostępnych danych wejściowych aniżeli algorytmy k-optymalizacyjne. Specyfika danych wejściowych uwydatnia podstawowe wady algorytmu k-medoidów, jak konieczność uprzedniego podania liczby grup, na jakie chcemy podzielić zbiór danych, czy też spora zależność końcowych wyników procesu grupowania od warunków początkowych. Algorytmy hierarchiczne pozbawione są tych wad, co wyjaśniły znaczne różnice w parametrach kompletności i dokładności. Dodatkowo można stwierdzić, że *k*-optymalizacyjny algorytm grupujący *k*-medoidów dla zaprezentowanego zestawu specyficznych danych tekstowych nie osiągał zadowalających i oczekiwanych wyników. Sytuacji nie poprawiała nawet zmiana w sposobie inicjalizacji początkowych reprezentantów skupień. Porównanie z algorytmem hierarchicznym wykazało jego przewagę w jakości otrzymywanych zgrupowań, co wskazuje, że to algorytmy z tej grupy lepiej sprawdzą się przy zadaniu grupowania w stosunku do tak specyficznych danych. Kolejnym wnioskiem płynącym z pracy [4] był fakt, iż bez odpowiedniego standardu opisu każdej pracy dyplomowej, a co za tym idzie – stworzenia minimalnego zbioru słów kluczowych opisujących dokumenty zawarte w bazie, komfort użytkowania systemu znacząco spadnie (gdy użytkownik będzie miał do wyboru zbyt dużą liczbę słów kluczowych, nie będzie mógł podjąć decyzji co do wyboru właściwych), a sam system zamiast wspomagać decyzję może ją utrudnić. Z tego względu podjęto próby poszukiwania nowych metod opisu prac pod kątem redukcji słów kluczowych oraz znalezienia innych metod grupowania dokumentów.

5. Eksperymenty

Eksperymenty miały na celu analizę efektywności algorytmów grupowania gęstościowego w odniesieniu do dokumentów tekstowych. Przedmiotem badań było zachowanie się algorytmu dla różnych przypadków pytań zadawanych do systemów opartych na gęstościowych grupach dokumentów. Chciano zweryfikować poziom efektywności mierzonej w przypadku systemów wyszukiwania informacji: parametrami kompletności oraz dokładności. Nie

bez znaczenia, jak wiadomo, dla tych parametrów jest procent dokumentów relewantnych względem danego pytania w systemie. W systemach o dużej liczbie dokumentów, gdzie liczba dokumentów relewantnych jest stosunkowo niewielka – w skrajnym przypadku może to być np. tylko jeden dokument relewantny, efektywnym systemem byłby ten, który potrafiłby znaleźć ów jeden relewantny dokument. Jednakże, gdy algorytm grupowania dokumentów będzie źle dobrany, może zdarzyć się tak, że ten relewantny dokument nie zostanie wyszukany. Oczywiste jest, że gdy liczba dokumentów relewantnych w stosunku do liczby wszystkich dokumentów w zbiorze jest bardzo liczna (powiedzmy, że nawet bliska połowie), wówczas szanse na to, że na zadane pytanie zostanie w odpowiedzi znaleziony dokument relewantny, są zawsze większe – można nawet powiedzieć spore. Reasumując powiemy, że wartość dokładności jest zależna nie tylko od wybranego algorytmu, ale również od liczby faktycznych dokumentów relewantnych w danym systemie. Inaczej jest w przypadku kompletności. Jak wiadomo, system będzie wówczas kompletny, gdy w odpowiedzi na zadane pytanie wskaże wszystkie obiekty (dokumenty) relewantne. Wiadomo jednak, że często specyfika algorytmu powoduje, że nie dostaniemy całej grupy w odpowiedzi, a np. jedynie jej fragment. Wówczas z pełną dokładnością uzyskamy niepełną kompletność.

W kontekście wyszukiwania informacji relewantnej możemy powiedzieć, że głównym celem jest pełna dokładność, nawet kosztem pełnej kompletności. W przypadku algorytmów k-optymalizacyjnych, jak chociażby k-średnich czy k-medoidów, ze względu na duży szum informacyjny istnieje spore prawdopodobieństwo, że algorytm nie będzie znajdował wszystkich dokumentów relewantnych, ale jednocześnie też spore jest prawdopodobieństwo, że w odpowiedzi wskaże zawsze jakiś dokument relewantny. Z kolei algorytmy hierarchiczne, np. aglomeracyjne, jak AHC – w klasycznej wersji, przeszukiwanie drzewa binarnego skończy się znalezieniem jednego dokumentu liścia (zamiast większej liczby dokumentów relewantnych), jednak, co zazwyczaj ważniejsze – dokument ten będzie relewantny. Algorytmy gęstościowe powinny znajdować znacznie więcej dokumentów relewantnych, a więc powinny w ogólnym przypadku pozwalać na zwiększenie parametrów kompletności przy podtrzymaniu wysokiej dokładności.

5.1. Plan eksperymentów

W celu potwierdzenia tych przypuszczeń eksperyment będzie polegał na porównaniu wartości parametrów kompletności oraz dokładności dla trzech różnych grup algorytmów: *k*-optymalizacyjnych (*k*-medoidów), hierarchicznych (AHC) oraz gęstościowych (DBSCAN) i dla różnych przypadków pytań, a raczej odpowiedzi na nie. Mianowicie, dla każdego z wymienionych algorytmów eksperyment będzie dotyczył pytań, na które:

a) w bazie danych dokumentów składowany jest tylko jeden dokument relewantny,

b) liczba dokumentów relewantnych względem zapytania jest duża (powiedzmy, że większa niż 60 %),
c) liczba dokumentów relewantnych względem zapytania jest stosunkowo niewielka (powiedzmy, że nie większa niż 1 %).

Warte podkreślenia w tym momencie jest to, że każda z metod grupowania zastosowana w ramach prac narzuca w pewnym sensie sposób późniejszego przeglądu tak utworzonej struktury dokumentów oraz wyszukiwania z nich informacji.

Pamiętajmy, że w przypadku algorytmu *k*-medoidów struktura, która zostanie utworzona, to k grup dokumentów opisanych pewnym zbiorem słów kluczowych charakterystycznych dla danej grupy. Jako że metoda ta bardziej kładzie nacisk na utworzenie wymaganej liczby k grup niż to, by dokumenty relewantne względem siebie były na pewno ulokowane w jednej i tej samej grupie, nie możemy oczekiwać, że parametry efektywności będą zadowalające w każdym z analizowanych przypadków. Algorytm w wyniku zwróci użytkownikowi do przeglądu grupę wg niego najbardziej relewantną, ale nie ma żadnej pewności, że wszystkie umieszczone w niej dokumenty będą faktycznie relewantne. To samo się tyczy algorytmów hierarchicznych.

W efekcie grupowania algorytmem AHC utworzone zostanie drzewo dokumentów, którego przeszukiwanie będzie się sprowadzać do przeglądu drzewa (a raczej reprezentantów węzłów) od korzenia w dół i wyborze grupy, której podobieństwo do pytania użytkownika jest odpowiednio duże. Wówczas także mamy do czynienia z przypadkiem, w którym w znalezionej przez system grupie prócz dokumentów relewantnych będą jednak również te nierelewantne.

Wreszcie docieramy do metod gęstościowych. W tym przypadku struktura utworzona dzięki zastosowania algorytmu DBSCAN charakteryzuje się tym, iż powstało *k* grup, przy czym liczba k nie jest tu określana przez użytkownika, lecz wynika z natury utworzonych skupień dokumentów oraz wybranych wartości parametrów. Przeszukiwanie takiej struktury sprowadza się do znalezienia grupy najbardziej relewantnej. Wyższość tego typu algorytmów nad algorytmami k-optymalizacyjnymi polega na tym, że algorytmy gęstościowe dają nam większą szansę na to, że wszystkie podobne do siebie dokumenty będą umieszczone w tej samej grupie (jako że są ze sobą gęsto umieszczone). Problem z algorytmami k-optymalizacyjnymi polega na tym, że czasem mając dwa dokumenty podobne w pewnym stopniu do dwóch różnych grup (a w zasadzie ich reprezentantów) wybierze grupę o większym stopniu podobieństwa i w ten sposób pozwoli na budowanie dwóch grup oddzielnie, zamiast spróbować tworzyć jedno skupienie o rozszerzonej gęstości. Gdybyśmy w takiej sytuacji zastosowali algorytmy gęstościowe, teoretycznie wszystkie dokumenty podobne do siebie w jakimś (dopuszczalnym) stopniu (a więc takie, które spełniają zadany próg gęstości) powinny zostać ulokowane w jednej grupie. Dzięki temu przeszukując potem strukturę tak utworzonych grup

zazwyczaj powinniśmy znajdować jedną taką grupę, która z wyraźnie wysokim prawdopodobieństwem pozwoli znaleźć informacje zawarte w pytaniach użytkownika (a więc w zadanych słowach kluczowych). Algorytmy te dają zatem szansę na podwyższenie parametrów kompletności oraz dokładności odpowiedzi systemu w stosunku do wyników dostarczanych przez algorytmy hierarchiczne czy k-optymalizacyjne.

5.2. Wyniki eksperymentów

Oczywiście w trakcie prac nad analizą poszczególnych implementowanych algorytmów grupowania wykonano wiele eksperymentów, jednak na potrzeby niniejszej pracy 4 wydają się szczególnie istotne.

Tabela 1

Wyniki dla przypadku I testowego

	k-medoidów	AHC	DBSCAN
Liczba słów kluczowych w pytaniu	3	3	3
Liczba wszystkich dokumentów	360	360	360
Liczba dokumentów relewantnych	212	212	212
Liczba dokumentów jakie system zwrócił w odpowiedzi	95	56	182
Kompletność	0,4150943	0.2688680	0,5141509
Dokładność	0,9263158	1	0,5989011

Tabela 2

Wyniki dla przypadku II testowego

	k-medoidów	AHC	DBSCAN
Liczba słów kluczowych w pytaniu	3	3	3
Liczba wszystkich dokumentów	360	360	360
Liczba dokumentów relewantnych	18	18	18
Liczba dokumentów jakie system zwrócił w odpowiedzi	0	56	5
Kompletność	0	0.5000000	0,2777778
Dokładność	0	0.1578950	1

Tabela 3

Wyniki dla przypadku III testowego

	k-medoidów	AHC	DBSCAN
Liczba słów kluczowych w pytaniu	3	3	3
Liczba wszystkich dokumentów	360	360	360
Liczba dokumentów relewantnych	1	1	1
Liczba dokumentów jakie system zwrócił w odpowiedzi	1	56	1
Kompletność	1	1	1
Dokładność	1	0.0175439	1

Przypadek opisany w tabeli 1 dotyczył pytania użytkownika, który szukał odpowiedzi na pewien charakterystyczny zbiór słów kluczowych (3 słowa kluczowe). Ważny jest fakt, że spośród 360 dokumentów podlegających analizie aż 212 było dokumentami relewantnymi. Analizując zachowanie się poszczególnych metod grupowania, różne liczebnie były wyniki

wyszukiwania odpowiedzi na zadawane pytania. Otóż system oparty na grupowaniu metodą *k*-medoidów zwrócił w odpowiedzi 95 dokumentów, system oparty na algorytmie AHC zwrócił dokumentów najmniej, bo 56, zaś system bazujący na algorytmie DBSCAN wyszukał 182 dokumenty. Nie to jest jednak istotne. Najbardziej istotne jest oczywiście to, ile z tych dokumentów wyszukanych przez system było naprawdę dokumentami relewantnymi. Informacji takiej dostarczają wartości parametrów kompletności i dokładności. Dla algorytmu k-medoidów uzyskano kompletność równą 0,4150943 i dokładność równą 0,9263158. Trzeba przyznać, że są one obiecujące, gdyż tak otrzymana wartość dokładności (prawie optymalna) świadczy o tym, że wśród wyszukanych dokumentów nie było zbyt wielu dokumentów nierelewantnych. Kompletność na poziomie 0,4150943 wynika z faktu, że skoro było tak dużo dokumentów relewantnych, a system zwrócił w odpowiedzi o wiele mniejszy zbiór, to wartość ta nie mogła być wyższa. W przypadku algorytmu AHC co prawda kompletność jest stosunkowo niska (na poziomie 0.2688680), ale – co ważniejsze – dokładność jest pełna (równa 1). Niska kompletność wynika z tego, że system w odpowiedzi zwraca stosunkowo niewiele wyników, bo 56, ale na 212 dokumentów relewantnych wszystkie one są relewantne. To tłumaczy pełną dokładność. Interesujący jest przypadek algorytmu DBSCAN. Otóż, parametry efektywności nie są ani za niskie, ani za wysokie. Niepełność wartości kompletności można się spodziewać widząc, że liczba dokumentów zwróconych jako odpowiedź jest mniejsza niż liczba dokumentów relewantnych. Zastanawiać może dokładność na poziomie 60%, co oznacza, że widocznie system wśród dokumentów wyszukanych ulokował sporą część, bo bliską 40%, liczbę dokumentów nierelewantnych. Na uwagę zasługuję również przypadek 2 (opisany w tab. 2), w którym dokumentów relewantnych względem zadanego pytania było 18, co stanowi niewielki procent całości bazy dokumentów. Otóż, w przypadku algorytmu *k*-medoidów system nie znalazł w wyniku żadnego dokumentu, co tłumaczy zerowe wartości kompletności oraz dokładności. Z kolei algorytm AHC wyszukał 56 dokumentów, podczas gdy wiemy, że relewantnych było tylko 18. Oczywista jest z tego względu niska wartość dokładności. Kompletność na poziomie 0,5 oznacza, że system wyszukał tylko połowę z 18 dokumentów relewantnych. Najbardziej zadowalające wartości efektywności dostarcza algorytm gęstościowy, który pozwala otrzymać pełną dokładność i niską kompletność (na poziomie 28%), wynikającą z tego, że system zwrócił w odpowiedzi tylko 5 dokumentów z 18 relewantnych, ale na szczęście wszystkie były relewantne. Jeszcze bardziej ciekawy jest przypadek przedstawiony w tabeli 3, gdzie odpowiedzią na zadane pytanie był tylko jeden dokument. System, w którym dokumenty grupujemy algorytmem *k*-medoidów, jak i ten oparty na algorytmach gęstościowych, szczęśliwie znalazł ów jeden dokument, co przełożyło się na optymalne wartości parametrów kompletności i dokładności. System oparty na metodzie AHC otrzymał pełną kompletność (wśród 56 dokumentów wyszukanych był ten jeden relewantny), lecz – czego można się było

spodziewać – niską dokładność (na poziomie 0.017), wynikającą z faktu, że skoro system w odpowiedzi zwrócił aż 56 dokumentów, a tylko jeden był relewantny, to reszta była nierelewantna. Specyficzny jest przypadek ostatni, dany tabelą 4. Do systemu zadano bardzo ogólne pytanie w postaci jednego słowa kluczowego. Istotne jest to, że wśród 360 dokumentów tylko jeden był relewantny. I taki przypadek niestety pokazał niedoskonałość metod grupowania bądź metod wyszukiwania informacji w strukturach skupień dokumentów. Niestety, algorytmy *k*-medoidów i DBSCAN nie znalazły dokumentu relewantnego, zapewne ze względu na fakt, że przy budowie reprezentanta grupy nie wzięto pod uwagę tego słowa kluczowego, o które pytał użytkownik. Wówczas system nie miał możliwości odnalezienia tego dokumentu. W przypadku algorytmu AHC, gdy budowana jest struktura hierarchiczna w postaci drzewa binarnego, możliwe jest zastosowanie efektywnych technik przeszukiwania tego typu struktur w stosunkowo krótkim czasie i, co widać w wynikach, system ma szansę na pewnym poziomie w drzewie odnaleźć relewantne dokumenty. Kompletność w sensie zdolności do znalezienia dokumentów relewantnych jest równa 1, bowiem wśród 54 dokumentów zwróconych jako odpowiedź systemu był dokument relewantny. Z kolei dokładność rozumiana jako zdolność do niewyszukiwania dokumentów relewantnych rzecz jasna w tym przypadku będzie niska, bo na 54 zwrócone dokumenty, tylko jeden był relewantny, co daje wartość dokładności równą 0.0181818.

Tabela 4

Wyniki dla przypadku IV testowego

	k-medoidów	AHC	DBSCAN
Liczba słów kluczowych w pytaniu	1	1	1
Liczba wszystkich dokumentów	360	360	360
Liczba dokumentów relewantnych	1	1	1
Liczba dokumentów jakie system zwrócił w odpowiedzi	0	54	0
Kompletność	0	1	0
Dokładność	0	0.0181818	0

6. Podsumowanie

Celem niniejszej pracy była analiza efektywności systemów wyszukiwania dokumentów relewantnych względem pytań zadawanych przez użytkownika w odniesieniu do systemów o strukturze skupień dokumentów podobnych do siebie. Przedmiotem analizy stały się trzy grupy algorytmów grupowania: *k*-optymalizacyjne (algorytm *k*-medoidów opisany szczegółowo w pracy [9]), hierarchiczne (tutaj wybrano aglomeracyjny algorytm AHC, którego szczegóły implementacyjne można znaleźć w pracy [3]) oraz gęstościowe (wybrano algorytm DBSCAN). Eksperymenty miały na celu zbadać poziom efektywności odpowiedzi systemu mierzonej standardowymi miarami kompletności oraz dokładności. Wyniki wskazują, iż nie

bez znaczenia, jak wiadomo, dla tych parametrów jest procent dokumentów relewantnych względem danego pytania w systemie.

W systemach o dużej liczbie dokumentów zazwyczaj dąży się do wyszukania dokumentów relewantnych (niekoniecznie wszystkich możliwych) i do niewyszukania dokumentów, które relewantnymi nie są. Zatem ważniejsze jest uzyskanie większej dokładności, nawet kosztem mniejszej kompletności. Okazuje się, że na ostateczny poziom wartości tych parametrów wpływ ma nie tylko wybrany algorytm grupowania, ale również w dużym stopniu liczba dokumentów relewantnych w systemie. Wiadomo bowiem, że gdy w systemie mamy więcej dokumentów relewantnych względem danego pytania, daje to większe prawdopodobieństwo, że zostanie on wyszukany.

Generalizując, analizowane przypadki (nie tylko te, które umieszczono w tabelach 1-4) wskazują, że obiecujący jest algorytm gęstościowy. W większości przypadków pozwalał uzyskiwać pełną dokładność.

BIBLIOGRAFIA

1. Ester M., Kriegel H.P., Sander J., Xu X.: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. Institute for Computer Science, University of Munich. Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96), 1996.
2. Honkela T., Kaski S., Lagus K., and Kohonen, T.: Self-organizing maps of document collections. ALMA, 1(2). Electronic Journal. http://www.diemme.it/luigi/alma.html, 1996.
3. Jach T.: Grupowanie jako metoda eksploracji wiedzy w systemach wspomagania decyzji. Analiza algorytmów hierarchicznych. Sosnowiec, 2008.
4. Nowak A., Xięski T., Jach T.: Analiza hierarchicznych i niehierarchicznych algorytmów grupowania dla dokumentów tekstowych, STUDIA INFORMATICA, Zeszyty Naukowe Politechniki Śląskiej, Volume 30, No. 2A(83), s. 245÷258.
5. Nowak A., Wakulicz-Deja A., Bachliński S.: Optimization of Speech Recognition by Clustering of Phones. Fundamenta Informaticae, Vol. 72, 2006, s. 283÷293.
6. Rijsbergen C.J.: Information retrieval. online book http://www.dcs.gla.ac.uk/Keith/Preface.html, 1979
7. Salton G.: Automatic Information Organization and Retreival. McGraw-Hill, New York, USA, 1975.
8. Sander J., Ester M., Kriegel H.P., Xu X.: Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and its Applications, 1998.

9. Xięski T.: Grupowanie jako metoda eksploracji wiedzy w systemach wspomagania decyzji. Analiza algorytmów niehierarchicznych (k-optymalizacyjnych). Sosnowiec, 2008.
10. Wakulicz-Deja A.: Podstawy systemów wyszukiwania informacji. Analiza metod. Akademicka Oficyna Wydawnicza PLJ, Warszawa, 1995.

Recenzenci: Dr hab. inż. Andrzej Chydziński, prof. Pol. Śląskiej
Dr inż. Michał Kozielski

Wpłynęło do Redakcji 31 stycznia 2010 r.

Abstract

The paper presents the results of experiments based on methods of clustering textual documents. Authors used not only classical clustering algorithms like nonhierarchical (k-medoid) and hierarchical (AHC) but also density based algorithm (DBSCAN). The experiments are connected with some previous results of researches done on retrieval information systems and textual document clustering. The subject of analysis is similarity between documents that are clustered and method of creating as natural and well constructed clusters as possibile. In authors oppinion, the quality of searching documents' clusters is high only if we use proper clustering methods which are resistant to noise in data. In the experiments different types of questions were analyzed. The recall and precision are dependent on the number of relevant documents. The more relevant documents build documents' set, the higher value of recall and precision parameter is achieved. In general, the best results are obtained when using AHC or DBSCAN algorithms. It was because this methods created well clusters of documents, therefore during the search process we were able to find one group of documents that were relevant. Because of that, during the searching process, we could find one group of documents that were relevant to the given question and we get irrelevant documents as the answer to the query. Only in such case both parameters: recall and precision can achieve their optimal values.

Adresy

Agnieszka NOWAK – BRZEZIŃSKA: Uniwersytet Śląski, Instytut Informatyki, Wydział Informatyki i Nauki o Materiałach, ul. Będzińska 39, 41-200 Sosnowiec, Gliwice, Polska, Agnieszka.nowak@us.edu.pl

Tomasz JACH: Uniwersytet Śląski, Instytut Informatyki, Wydział Informatyki i Nauki o Materiałach, ul. Będzińska 39, 41-200 Sosnowiec, Gliwice, Polska, Tomasz.jach@us.edu.pl

Tomasz XIĘSKI: Uniwersytet Śląski, Instytut Informatyki, Wydział Informatyki i Nauki o Materiałach, ul. Będzińska 39, 41-200 Sosnowiec, Gliwice, Polska, Tomasz.xieski@us.edu.pl