Universitatea Babeş–Bolyai, Cluj-Napoca
Facultatea de Matematică şi Informatică

Învăţare semisupervizată folosind kerneluri

Teză de doctorat
Rezumat

Conducător ştiinţific:
Zoltán KÁSA

Doctorand:
Zalán-Péter BODÓ

Cluj-Napoca
2009

Cuprinsul rezumatului

Cuprinsul tezei de doctorat

Cuvinte cheie: instruire automată, învăţare semisupervizată, kerneluri şi metode kernel, kerneluri cluster, kerneluri semisupervizate

1 Introducere

Din anul 1956 inteligenţa artificială (*Artificial Intelligence – AI*) a devenit un domeniu intens studiat al informaticii, scopul ei fiind construcţia maşinilor inteligente. Totuşi inteligenţa nu este un concept bine definit şi nici unul uşor de definit – iar aici ne referim la definiţia informală a inteligenţei. O tentativă de a demonstra inteligenţa automat – astfel o încercare de a o defini – a fost bine-cunoscutul test Turing: o fiinţă umană poartă o conversaţie într-o limbă naturală cu o altă fiinţă umană şi cu o maşină, ambele încercând să pară a fi fiinţe umane, iar sarcina este de a determina care este fiinţa umană şi care este maşina. Dacă acest lucru nu poate fi judecat în mod fiabil, maşina "trece examenul". Astfel putem afirma că scopul AI este de a construi maşini care se comportă asemenea fiinţelor umane.

După Russel and Norvig [1995] sistemele AI pot fi organizate în patru grupuri: sisteme care gândesc într-un mod similar cu fiinţele umane, sisteme care pot acţiona ca şi fiinţele umane, sisteme care pot gândi raţional şi sisteme care pot acţiona în mod raţional. Învăţarea automată (*Machine Learning – ML*) este subdomeniul AI, care doreşte să modeleze cele mai importante activităţi ale creierului uman: clasificarea, diferenţierea şi predicţia; maşinile instruibile aparţin primei categorii descrise mai sus. În ultimii 20 de ani ştiinţele cognitive şi informatica s-au îndepărtat unul de altul şi noi domenii mult mai înguste s-au format în cadrul AI. Unele tehnologii au devenit indispensabile, şi în unele privinţe au depăşit ceea ce pot face fiinţele umane: să ne gândim doar la sistemele de regăsire a informaţiei, unde milioane de documente sunt scanate pentru a prezenta utilizatorului cele care conţin informaţii relevante, iar în prezent acurateţea atinsă de către aceste sisteme este comparabilă cu judecata umană.

Două dintre cele mai importante subdomenii ale ML sunt învăţarea supervizată şi cea nesupervizată: în cazul învăţării supervizate primim exemple împreună cu instrucţiuni de instruire pe care le numim etichete; învăţarea nesupervizată este mai dificilă din cauza că nu sunt furnizate instrucţiuni adiţionale. Sarcina obiş-

nuită în cazul învăţării nesupervizate este determinarea clusterelor prin gruparea punctelor similare sau găsirea densităţii probabilistice a datelor.

Prezenta teză se concentrează pe învăţarea semisupervizată (*Semi-Supervised Learning – SSL*): din cauza că adnotarea umană a exemplelor de instruire în general solicită expertiză în domeniu, aceasta este costisitoare şi consumatoare de timp. O soluţie este utilizarea numai a unei mici proporţii a datelor etichetate pentru a îmbunătăţi performanţa algoritmului de instruire. De exemplu să presupunem că într-o problemă de clasificare a documentelor cuvântul "profesor" se dovedeşte a fi un bun mijloc de predicţie a exemplelor pozitive pe baza datelor etichetate. Apoi, dacă datele neetichetate arată faptul că termenul "profesor" şi "universitate" sunt corelate, prin utilizarea ambelor cuvinte se poate aştepta creşterea acurateţei clasificatorului.

1.1 Învăţare semisupervizată şi metode kernel

Funcţiile kernel sau kerneluri returnează similarităţile între exemple. În cazul metodelor kernel nu lucrăm cu datele în mod direct, ci cu ajutorul unei matrice numită matricea kernel, ceea ce conţine similarităţile între exemple. Aceasta la rândul ei este analoagă clasificării umane, unde similaritatea între exemple joacă un rol important [Estes, 1994]. Kerneluri sunt instrumente pentru extensia neliniară a metodelor lineare: dacă un algoritm poate fi scris în termenii produselor scalare, putem în mod simplu schimba matricea de produse scalare cu o funcţie/matrice arbitrară pozitiv semidefinită care să conţină produsele scalare ale datelor în aşa-numitul spaţiu al caracteristicilor (*feature space*), şi obţinem o extensie neliniară a algoritmului.

Kerneluri dependente de date dau naştere la maşini instruibile semisupervizate: funcţia kernel nu depinde în continuare numai de cele două puncte în discuţie, ci utilizează data completă, informaţiile cuprinse în setul complet de instruire disponibil. Matematic vorbind, dacă D_1 şi D_2 notează două seturi de date, $D_1 \neq D_2$, $\mathbf{x}, \mathbf{z} \in D_1 \cap D_2$, atunci

$$k(\mathbf{x}, \mathbf{z}; D_1) \eqslantless k(\mathbf{x}, \mathbf{z}; D_2)$$

unde prin $k(\cdot, \cdot)$ notăm funcţiile kernel, ";" însemnând condiţionare şi "$\eqslantless$" însemnând "nu neaparat egal". Kernelurile dependente de date sunt folosite pentru a îmbunătăţi măsura de similaritate care ia în considerare doar data etichetată. Reprezentarea în spaţiul caracteristicilor este acum aleasă folosind informaţiile exploatate din seturile de date etichetate şi neetichetate. Kernelurile de acest fel pot fi utilizate în oricare metodă kernel în cazul în care sunt disponibile şi date neetichetate.

Subiectul acestei teze este construcţia acestor kerneluri dependente de date pentru învăţare supervizată şi semisupervizată şi de a dovedi superioritatea kernelurilor dependente de date faţă de kerneluri convenţionale.

1.2 Structura tezei

Teza este organizată în opt capitole. Capitolul 1 este capitolul introductoriu, prezentând pe scurt tema şi stucturra tezei. Capitolul 2 constă într-o prezentare a tehnicilor SSL şi a conceptelor asociate. După prezentarea prezumţiilor folosite în SSL, o clasificare a metodelor SSL este dată printr-o scurtă prezentare a unei metode din fiecare categorie.

În Capitolul 3 introducem metodele kernel şi prezentăm o descriere detaliată a maşinilor cu suport vectorial (*Support Vector Machines – SVMs*), a metodei *Principal Component Analysis* (PCA) şi a metodei *Kernel Principal Component Analysis* (KPCA). Metodele prezentate aici vor fi utilizate mai târziu în cadrul tezei.

Capitolul 4 prezintă tehnicile de construcţie a kernelurilor dependente de date, de exemplu kernelul ISOMAP, kernelul de vecinătate, kernelul bagged cluster, kernelul cluster multi-type şi un kernel dependent de date legat de regularizări de varietăţi (*manifold regularization*).

Următoarele trei capitole constituie partea principală a tezei: ele conţin principalele contribuţii în domeniul construcţiei kernelurilor dependente de date. Capitolul 5 prezintă kernelul propus de autor bazat pe Wikipedia pentru categorizarea textelor (*Text Categorization – TC*). Acest capitol oferă în acelaşi timp o introducere detaliată în domeniul categorizării textelor.

Capitolul 6 prezintă kernelul cluster ierarhic propus pentru învăţarea semisupervizată. S-au implementat kernelurile dependente de date prezentate în capitolul 4 şi le-am comparat în mod experimental cu kernelul ierarhic pe diferite seturi de date.

În capitolul 7 propunem trei kerneluri cluster utilizând proprietatea produsului Hadamard a matricelor pozitiv semidefinite.

Capitolul 8 rezumă şi încheie teza, iar Anexa A descrie seturile de date pe care le-am utilizat în evaluarea metodelor prezentate în teză.

Contribuţia autorului în domeniu poate fi rezumată în următorul fel:

- un nou kernel pentru categorizarea documentelor, bazat pe informaţii extrase din Wikipedia
  - propunerea includerii structurii pe linkuri a Wikipediei în kernel
  - ponderarea conceptelor în kernelul bazat pe Wikipedia utilizând algoritmul PageRank

- o nouă metodă de construire a kernelurilor cluster ierarhici pentru învăţarea supervizată şi semisupervizată
  - propunerea unui cadru general pentru construcţia kernelurilor cluster ierarhici
  - definirea kernelurilor cluster ierarhici şi celor bazate pe grafuri
- construcţia kernelurilor prin utilizarea proprietăţii produsului Hadamard a matricelor pozitiv semidefinite
  - introducerea kernelului gaussian de reponderare
  - introducerea a doi kerneluri de reponderare prin utilizarea produselor scalare ale vectorilor de apartenenţă cluster

În cele de mai jos prezentăm metodele propuse în teză, incluse în capitolele 5–7.

2 Kerneluri de clasificare a documentelor bazate pe Wikipedia

Clasificarea textelor sau a documentelor constituie problema determinării categoriilor predefinite reale ale documentelor scrise în limbaje naturale pe baza unor exemple de instruire. Constituie un subdomeniu al regăsirii informaţiei (*Information Retrieval – IR*), dar de multe ori este considerată în mod eronat a fi o problemă de prelucrare a limbajelor naturale. Reprezentarea documentelor cu ajutorul modelului de spaţiu vectorial (*Vector Space Model – VSM*) cel mai performant şi utilizat pe scală largă creează vectori extrem de multidimensionali şi împrăştiaţi; din această cauză pentru ca ei să funcţioneze în mod eficient în acest spaţiu de reprezentare, este nevoie de aplicarea unor tehnici de reducere a dimensiunii. Metodele kernel oferă o metodă elegantă de a depăşi dimensionalitatea: au nevoie doar de matricea kernel care conţine similarităţile de date.

Propunem o metodă de construcţie a kernelurilor dependente de date care are mari şanse să producă rezultate mai potrivite în cazul problemelor de clasificare a documentelor. Pentru a construi un kernel mai potrivit pentru similarităţile documentelor, utilizăm informaţiile textuale valoroase furnizate de către Wikipedia.

2.1 Reprezentarea documentelor bazată pe Wikipedia

În VSM cu `tfidf` documentele de ponderare sunt reprezentate prin frecvenţa termenilor multiplicate cu factorul `idf`. Matricea termen$\times$document $\mathbf{D}$ este definit

în modul următor:

$$\mathbf{D} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n_D} \\ \vdots & \vdots & \vdots & \vdots \\ w_{n_T1} & w_{n_T2} & \cdots & w_{n_Tn_D} \end{bmatrix}$$

În acest fel avem o dublă reprezentare: (i) reprezentarea documentelor (coloanele), şi (ii) reprezentarea termenilor (rândurile). Fiecare document este reprezentat de termenii care apar în document, iar fiecare termen este reprezentat de documentele în care acel termen apare.

Acum trecem la o altă reprezentare a documentelor. Prima dată dăm o nouă reprezentare a termenilor, adică reprezentăm fiecare termen ca şi distribuţia acelui termen în câteva documente (diferite). Ceea ce priveşte reprezentarea unui document – a unui set de termeni – în acest spaţiu al noului document formăm suma ponderată a vectorilor termenilor. Considerăm următorul exemplu: avem trei termeni de indexare iar documentul nostru arată în următorul fel: $[1\ 0\ 1]'$. Alegem doar două alte documente din corpus care arată în următorul fel:

$$\begin{bmatrix} 2 & 1 & 0 \\ 1 & 3 & 2 \end{bmatrix}$$

unde aşezăm documentele în rândurile matricei. Acum termenii primesc următoarele reprezentări:

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix} \cdot [1] = \begin{bmatrix} 2 \\ 1 \end{bmatrix}; \quad \begin{bmatrix} 1 \\ 3 \end{bmatrix} \cdot [0] = \begin{bmatrix} 0 \\ 0 \end{bmatrix}; \quad \begin{bmatrix} 0 \\ 2 \end{bmatrix} \cdot [1] = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$

în cazul cărora vectorul documentului va fi $1 \cdot [2\ 1]' + 0 \cdot [0\ 0]' + 1 \cdot [0\ 2]' = [2\ 3]'$. Se poate observa uşor faptul că acesta este chiar kernelul GVSM [Wong et al., 1985], aşadar transformăm un document $\mathbf{d}$ folosind $\mathbf{D}'\mathbf{d}$, cu condiţia că documentele din care extragem distribuţia termenului formează acelaşi corpus din care provin documentele.

Gabrilovich and Markovitch [2007] au modificat transformarea GVSM în următorul fel: în locul utilizării aceluiaşi corpus pentru detectarea corelărilor termenilor ei au folosit Wikipedia pentru extragerea distribuţiei termenilor. Wikipedia este compusă din aproximativ 2.5×10^7 articole[1] şi astfel va oferi o mai bună, o mai bogată reprezentare. În cele de mai jos vom utiliza termenii "articol" şi "concept" alternativ.

Să presupunem că Wikipedia conţine n_C articole, sau am ales atâţia din setul complet, şi suntem interesaţi de distribuţia în aceste articole Wikipedia ai termenilor de indexare. Transformarea documentului în spaţiul conceptual al Wikipedia

[1] August, 2008

devine următorul:

$$\underbrace{\begin{pmatrix} c_{11} & c_{12} & \dots & c_{1n_T} \\ c_{21} & c_{22} & \dots & c_{2n_T} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n_C1} & c_{n_C2} & \dots & c_{n_Cn_T} \end{pmatrix}}_{\boldsymbol{W}} \cdot \underbrace{\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_{n_T} \end{pmatrix}}_{\mathbf{d}}$$

Denumirea acestei matrice este matricea concept×termen Wikipedia, denumirea scurtă fiind matricea Wikipedia.

Gabrilovich and Markovitch [2007] numesc această metodă Explicit Semantic Analysis (ESA) pentru că spre deosebire de Latent Semantic Analysis (LSA) [Deerwester et al., 1990], ei aplică aceşti termeni/documente cu concepte explicite şi nu unele latente. Putem interpreta această transformare – şi în mod similar transformarea GVSM – ca transformarea vectorului documentului într-un vector al similarităţilor între document şi conceptele Wikipedia. Astfel o comparaţie în această nouă reprezentare compară aceşti vectori în mod similar, măsurând proximitatea acestor similarităţi. În acest fel noul kernel de document devine

$$k(\mathbf{d}_i, \mathbf{d}_j) = \mathbf{d}_i' \boldsymbol{W}' \boldsymbol{W} \mathbf{d}_j$$

unde $\boldsymbol{W}'\boldsymbol{W}$ este matricea termen×termen de co-apariţie a termenilor din Wikipedia.

Gabrilovich and Markovitch [2007] au folosit similaritatea cosinus pentru a compara cuvinte şi texte, şi au testat noua reprezentare pe colecţia WordSimilarity-353[2] şi pe o colecţie de 50 de documente din serviciile de ştiri a *Australian Broadcasting Corporation* [Lee et al., 2005]. Valorile de similaritate care au rezultat din utilizarea noii reprezentări cu similaritatea cosinus au fost comparate cu judecata umană, calculând corelaţiile dintre fiinţele umane şi calculator. Au obţinut coeficienţi de corelaţie de 0.75 şi 0.72 pentru cuvinte respectiv texte, care sunt valoarile cele mai ridicate produsă de un astfel de sistem automatizat. Pentru performanţele celorlalti algoritmi a se vedea studiul de referinţă.

Inspirat de performanţa reprezentării documentelor în spaţiul de concepte Wikipedia, am decis să utilizăm acest kernel pentru clasificarea textelor. Am încercat să reducem dimensionalitatea şi să filtrăm zgomotul din această reprezentare.

Această metodă de transformare a documentelor poate fi considerată o tehnică semisupervizată unde un corpus larg neetichetat al articolelor Wikipedia este utilizat pentru a conferi documentelor o nouă reprezentare.

[2] `http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353`

2.2 Reducerea dimensionalităţii kernelului Wikipedia

În timpul construirii kernelului Wikipedia nu am utilizat toate articolele, pentru că multe dintre ele sunt prea scurte pentru a fi folosite şi sunt multe articole irelevante, de exemplu paginile colectoare sau de categorizare (e.g. `Category:Machine_learning`), pagini interne Wikipedia (e.g. `Wikipedia:Statistics`) etc. Din această cauză am selectat o fracţiune a articolelor de pe Wikipedia pe care le-am considerat utilizabile, metodologia completă fiind descrisă în teză şi în Capitolul 5.

Pentru filtrarea termenilor irelevanţi am decis să utilizăm LSA, adică aproximarea matricei Wikipedia cu o matrice de rang inferior,

$$\mathbf{W} \approx \widehat{\mathbf{W}} = \mathbf{U}\mathbf{S}_k\mathbf{V}'$$

unde $\mathbf{U}\mathbf{S}\mathbf{V}'$ reprezintă descompunerea valorilor singulare (*Singular Value Decomposition – SVD*) a matricei $\mathbf{W}$. Acesta înseamnă că noul kernel Wikipedia devine

$$k(\mathbf{d}_i, \mathbf{d}_j) = \mathbf{d}_i'\widehat{\mathbf{W}}'\widehat{\mathbf{W}}\mathbf{d}_j = \mathbf{d}_i'\mathbf{V}_k\mathbf{S}_k^2\mathbf{V}_k'\mathbf{d}_j$$

Am experimentat cu kernelul de mai sus, dar am observat că prin folosirea $\mathbf{S}_k$ unele dimensiuni au primit o foarte mare influenţă, care la rândul ei a dus la reducerea performanţei. Astfel am înlocuit $\mathbf{S}_k$ cu $\mathbf{I}_k$ şi am obţinut kernelul următor:

$$k(\mathbf{d}_i, \mathbf{d}_j) = \mathbf{d}_i'\mathbf{V}_k\mathbf{V}_k'\mathbf{d}_j$$

O altă interpretare a transformării de mai sus ar putea fi următorul: cum primul component al descompunerii SVD a matricei $\mathbf{D}$ conţine vectorii proprii sau componentele principale ale spaţiului feature în LSA, în acest caz $\mathbf{V}$ îl va conţine (din cauza că $\mathbf{D}$ este o matrice termen×document, iar $\mathbf{W}$ este o matrice concept×termen), şi am asumat că articolele Wikipedia produc o covariaţie mai bună, de aceea am înlocuit $\mathbf{D}\mathbf{D}'$ cu $\mathbf{W}'\mathbf{W}$.

O problemă cu decompoziţia de mai sus este că $\mathbf{W}$ devine destul de mare – mai ales că n_C este mare – astfel SVD-ul matricei este foarte ineficient, şi chiar dificil de efectuat în timp acceptabil. Totuşi putem observa faptul că prin decompoziţia pe baza valorilor proprii a matricei mult mai mici $\mathbf{W}'\mathbf{W}$ (de mărimea $n_T \times n_T$) putem obţine $\mathbf{V}$,

$$\mathbf{W}'\mathbf{W} = \mathbf{V}\mathbf{S}^2\mathbf{V}'$$

2.3 Structura pe linkuri a Wikipedia

Wikipedia este de asemenea structurată pe linkuri, ceea ce poate fi utilizată în mod eficient prin propagarea frecvenţelor prin aceste conexiuni. Considerăm matricea de legătură concept×concept $\mathbf{E}$, care este definită ca $E_{ij} = 1$ în cazul în care există

un link între conceptele i şi j, iar altfel ia valoarea 0. Pentru că dorim să păstrăm ponderile deja atribuite, setăm diagonala principală cu valorile 1. Astfel matricea actualizată va deveni

$$\widetilde{\boldsymbol{W}} = \mathbf{E}'\boldsymbol{W}$$

Acesta înseamnă că $\widetilde{W}_{ij} = (\mathbf{E}_{\cdot i})' \cdot \boldsymbol{W}_{\cdot j}$, adică adăugăm W_{ij}-ului totalul aparţiilor termenului j în conceptele $C = \{k_1, \ldots, k_n\}$, unde setul C conţine conceptele – sau indicile acelor concepte – din direcţia cărora s-a produs un link către conceptul i.

2.4 Ponderarea conceptelor

În timp ce cuvintele/dimensiunile în corpusul de instruire primesc importanţe variate prin utilizarea ponderii `tfidf`, conceptele sunt considerate de către Wikipedia de a avea importanţe egale. O schemă posibilă de ponderare poate fi obţinută prin clasificarea articolelor Wikipedia pe baza analizei citatelor sau a referinţelor, astfel aceste importanţe pot fi extrase din structura pe linkuri discutată în capitolul anterior. Faimosul algoritm de PageRank [Page et al., 1998] a Google obţine acest lucru doar prin luarea în considerare a structurii de linkuri sau de citate a unui set de pagini hyperlinkate: o pagină are un rang înalt dacă deţine multe back-linkuri sau dacă sunt doar câteva back-linkuri, dar acele sunt importante. Acest lucru poate fi formulat în mod recursiv prin

$$\mathbf{r}_i = \sum_{j \in N^{-1}(i)} \frac{\mathbf{r}_j}{|N^{-1}(j)|}$$

unde $\mathbf{r}_i$ notează rangul paginii al i-lea şi $N^{-1}(i)$ este setul de vecini backward a i, adică vecinii care sunt îndreptate spre i. Dacă $\mathbf{P}$ notează matricea de adiacenţă având $P_{ij} = 1/|N(i)|$, unde $N(i)$ reprezintă vecinii forward ai i, vecini în direcţia cărora se îndreaptă i, rangurile pot fi calculate prin rezolvarea problemei de valori proprii $\mathbf{r} = \mathbf{P}'\mathbf{r}$.

PageRank poate fi considerat un mers aleator pe un graf, unde rangurile sunt probabilitatea ca un navigator aleatoriu să fie pe pagina respectivă în pasul k. În cazul în care k este suficient de mare, probabilităţile converg într-o distribuţie unică fixă. Singura problemă cu formularea de mai sus este că graful poate avea noduri fără vecini forward, şi grupuri din care nu iese nici un link forward. Pentru a rezolva aceste probleme este necesar să se adauge incertitudine în procesul de navigare

$$\mathbf{r} = c\mathbf{P}'\mathbf{r} + (1-c)\mathbf{1}, \quad c \in [0,1]$$

unde prin $\mathbf{1}$ am notat vectorul $[1\ 1\ \ldots\ 1]'$ de mărimea n_p având n_p pagini. În experimentele noastre am folosit valoarea $c = 0.85$ conform [Page et al., 1998].

Prin utilizarea rangurilor produse de algoritmul PageRank înlocuim $\boldsymbol{W}$ cu $\mathrm{diag}(\widetilde{\mathbf{r}})\boldsymbol{W}$, unde $\widetilde{\mathbf{r}} = \log(\mathbf{r} + \mathbf{1})$. Utilizând funcţia log am încercat să rafinăm ponderile într-o oarecare măsură, pentru că proporţia conceptului de cel mai înalt şi cel mai scăzut rang a fost de 8.8×10^3. Am utilizat translaţia $\mathbf{r} + \mathbf{1}$ în loc de $\mathbf{r}$ pentru a evita ponderile negative.

3 Kerneluri cluster ierarhici

În acest capitol introducem kerneluri cluster ierarhici în învăţarea semisupervizată. Propunem utilizarea distanţelor induse de algoritmi de clustering în locul distanţelor euclidiene în spaţiul input. Dacă datele neetichetate sunt adăugate unui set relativ mic de date etichetate, presupunem că noua distanţă, obţinută prin clustering şi utilizarea datelor neetichetate să inducă un spaţiu de reprezentare mai potrivit pentru clasificare. Pentru clustering utilizăm tehnici speciale de clustering ierarhic – acelea care rezultă în matrice de distanţă ultrametrice – care are ca rezultat matrice kernel pozitiv semidefinite.

Metoda propusă se bazează pe kernelul de conectivitate [Fischer et al., 2003] şi extindem asupra acestei construcţii kernel prin admiterea oricărei metode de clustering ierarhic dacă acesta are ca rezultat o matrice de distanţă ultrametrică. Procesul de clustering necesită setul complet de date, şi doar subsetul lui etichetat este utilizat pentru construirea kernelului în problema de clasificare. Totuşi, dacă datele de testare sunt disponibile în timpul instruirii, putem să le includem în setul de instruire ca date neetichetate adiţional, şi apoi utilizând partea corespunzătoare a matricei de distanţă. În cazul seturilor de date unde prezumţia de varietate (*manifold assumption*) este aşteptată a ţine, construim kernelul cluster ierarhic utilizând distanţele determinate pe grafurile de date kNN (k-*nearest neighbors*) şi ϵNN (ϵ-*nearest neighbors*) [von Luxburg, 2006].

3.1 Clustering ierarhic

Clustering este împărţirea unui set în grupe separate numite clustere. Cu toate că clusterele sunt de multe ori dezmembrate, ele pot fi în acelaşi timp şi suprapuse, ceea ce înseamnă că un punct poate aparţine mai multor clustere. O metodă de clustering poate fi ierarhic sau partiţionat. Clusteringul ierarhic construieşte un arbore în paşi succesivi, unde nodurile arborelui reprezintă partiţionările suprapuse ale datelor. Prin partiţionare înţelegem o decompoziţie a setului de date în mai multe grupe. Clusteringul partiţionat rezultă într-o singură partiţie [Jain et al., 1999; Berkhin, 2002]. În cele de mai jos vor fi utilizate în primul rând metode de clustering ierarhic.

Avem de-a face cu un clustering suprapus în cazul în care o partiţie este formată din componente fuzionate. Clusteringul ierarhic este compus dintr-o secvenţă de partiţii cu fiecare partiţie suprapusă cu următoarea partiţie a secvenţei. În cazul metodelor aglomerative la început fiecare punct este un cluster, iar clusterele cele mai apropiate sunt îmbinate până la obţinerea unui singur cluster mare. În cazul metodelor divizive la început tot setul de date este un singur cluster, apoi clusterele sunt despărţite până când sunt obţinute clustere formate de câte un punct [Jain and Dubes, 1988; Duda et al., 2001]. În cazul kernelului cluster ierarhic utilizăm o formă specială a metodei de clustering aglomerativ. Un algoritm general de clustering, care produce un dendrogram, este următorul:

Algoritmul 1 Clustering aglomerativ

1: Se definesc clusterele iniţiale ca şi punctele în sine.
2: Se găseşte perechea de clustere cele mai similare.
3: Se îmbină acestea pentru a crea un nou cluster.
4: Se repetă de la pasul 2 până când se obţine un singur cluster.

Pentru a specifica algoritmul complet, este nevoie de măsurarea similarităţilor între clustere; acestea sunt denumite *distanţe de legătură*. Pe baza alegerii distanţei de legătură devine posibilă designul unei mari varietăţi de metode de clustering aglomerativ, prezentate în detaliu mai jos.

3.2 Distanţele de legătură

Distanţele de legătură (*linkage distances*), $D(C_1, C_2)$, măsoară distanţele între clustere în clusteringul aglomerativ. Aici prezentăm trei distanţe de legătură populare. Acestea sunt bazate pe $d(\mathbf{x}_1, \mathbf{x}_2)$, *distanţa pe puncte* (*pointwise distance*) în spaţiul input, care este de obicei distanţa euclidiană, $d(\mathbf{x}_1, \mathbf{x}_2) = \|\mathbf{x}_1 - \mathbf{x}_2\|_2$.

(1) Clusteringul *single linkage* foloseşte următoarea funcţie de distanţe cluster:

$$D(C_1, C_2) = \min\{d(\mathbf{x}_1, \mathbf{x}_2) \mid \mathbf{x}_1 \in C_1, \mathbf{x}_2 \in C_2\} \tag{1}$$

Adică alegem să fuzionăm două clustere unde distanţa pe puncte este minimală. Se poate observa că clusteringul single linkage este echivalentul găsirii arborelui de acoperire al grafului de date, adică dacă considerăm graful iniţial ca şi graful complet al punctelor de date, atunci prin alegerea repetată a muchiei cu ponderea minimală obţinem arborele de acoperire minim al grafului [Duda et al., 2001].

(2) Clusteringul *complete linkage* defineşte distanţa între clustere în felul următor:

$$D(C_1, C_2) = \max\{d(\mathbf{x}_1, \mathbf{x}_2) \mid \mathbf{x}_1 \in C_1, \mathbf{x}_2 \in C_2\} \tag{2}$$

şi poate fi considerat ca unul care lucrează cu grafurile complete în cadrul clusterelor, adică în fiecare cluster fiecare nod este conectat cu toate celelalte noduri.

Dacă definim diametrul unui cluster fiind muchia cea mai lungă între două puncte, iar diametrul unei partiţii este definit ca fiind cel mai mare diametru al clusterelor sale, în complete linkage, din formula (2), alegem să îmbinăm acele clustere care măresc cel mai puţin diametrul partiţiei [Duda et al., 2001].

Metodele de mai sus tind să fie sensibile punctelor outlier. De altfel, clusteringul single linkage poate lega în mod eronat clustere şi poate forma clustere de o omogenitate redusă, iar clusteringul complete linkage poate rezulta în clustere care nu sunt bine separate [Jain and Dubes, 1988].

(3) Clusteringul *average linkage* – reprezentând un compromis – ia în considerare media distanţelor pe puncte dintre toate perechile de elemente din cele două clustere:

$$D(C_1, C_2) = \frac{1}{|C_1||C_2|} \sum_{\mathbf{x}_{1i} \in C_1} \sum_{\mathbf{x}_{2j} \in C_2} d(\mathbf{x}_{1i}, \mathbf{x}_{2j}) \tag{3}$$

În această teză experimentăm doar cu cele trei distanţe de legătură prezentate mai sus. Alte tehnici populare, bazate pe variaţia clusteringului average linkage (denumit *UPGMA – unwighted pair group method using arithmetic mean*) include *weighted average linkage clustering* (WPGMA), *average group linkage clustering* (UPGMC), *weighted average group linkage clustering* (WPGMC) şi metoda lui Ward [Jain et al., 1999; Berkhin, 2002].

Toate cele trei metode au următoarele calităţi utilizate pentru construcţia kernelurilor pozitiv semidefinite pentru clusteringul aglomerativ: să presupunem că alegem să îmbinăm trei clustere, C_1, C_2 and C_3 în următoarea ordine: prima dată îmbinăm C_1 cu C_2 ceea ce rezultă în C_{12}, apoi vom îmbina acesta din urmă cu C_3. Dacă

$$D(C_1, C_2) \leq D(C_1, C_3) \quad \text{şi} \quad D(C_1, C_2) \leq D(C_2, C_3)$$

atunci

$$D(C_1, C_2) \leq D(C_{12}, C_3)$$

Această calitate este denumită *proprietatea ultrametrică* sau ultrametricitate. Pe baza unei metode de clustering aglomerativ care utilizează distanţe de legătură ultrametrice, putem defini o matrice de distanţe ultrametrice, şi pe rând o funcţie kernel care poate fi utilizată pentru o reprezentare mai potrivită.

3.3 Construirea kernelului

Aşa cum am afirmat mai sus, clusteringul ierarhic rezultă într-un dendrogram, nodurile căruia sunt etichetate cu distanţa între clusterele care sunt îmbinate la nodul respectiv. Construim o matrice de distanţe prin considerarea etichetei ataşate strămoşului comun cel mai apropiat al punctelor de pe arbore. Pentru a transforma

distanţele în produse scalare utilizăm o metodă similară scalării multidimensionale (*Multi-Dimensional Scaling – MDS*) [Borg and Groenen, 2005]:

$$\mathbf{K} = -\frac{1}{2}\mathbf{JMJ} \qquad \text{cu} \quad \mathbf{J} = \mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}'$$

unde $\mathbf{M}$ conţine distanţele bazate pe dendrogramă la pătrat, şi $\mathbf{J}$ este matricea de centrare construită din matricea de identitate şi produsul tensorial al vectorului cu toate elementele egale cu 1. Matricea care rezultă din aceasta conţine produsele scalare între un set de vectori $\{\mathbf{z}_i\}_{i=1}^N$ cu distanţele euclidiene la pătrat $\|\mathbf{z}_i - \mathbf{z}_j\|_2^2 = M_{ij}$. Utilizăm următoarea teoremă care declară că prin această transformare matricele de distanţă ultrametrică rezultă întotdeauna în matrice Gram.

Theorem 1 (Fischer et al. [2003])**.** *Dat fiind un* $\mathbf{M}$ *ultrametric care conţine distanţele bazate pe dendrogramă, matricea* $\mathbf{K} = -\frac{1}{2}\mathbf{JMJ}$ *este o matrice Gram pozitiv semidefinită.*

În cele de mai jos construim kernelul cluster cu utilizarea distanţelor de legătură din clusteringul ierarhic. Astfel putem aplica punctele într-un spaţiu feature unde distanţele pe puncte sunt egale cu distanţele cluster în spaţiul input. Paşii acestuia sunt:

Algoritmul 2 Kernelul cluster ierarhic

1: Efectuăm un clustering aglomerativ pe datele etichetate şi neetichetate – de exemplu prin utilizarea funcţiilor de legături single, complete sau average.
2: Definim matricea $\mathbf{M}$ cu elemente M_{ij} = distanţa de legături în arborele ultrametric la subsumarea cel mai puţin generală a i şi j ; $M_{ii} = 0$, $\forall i$.
3: Definim matricea kernel ca şi $\mathbf{K} = -\frac{1}{2}\mathbf{JMJ}$.

Kernelul care rezultă în acest fel este în mod evident unul dependent de date. Folosim datele neetichetate în pasul de clustering pentru a determina distanţe pe puncte "mai bune" care duc la kernel. Aşteptarea noastră este obţinerea unei mai bune similarităţi prin includerea părţii neetichetate. Este important faptul că pentru a calcula funcţia kernel pentru setul de testare, le includem în setul neetichetat. Acesta înseamnă că dacă punctul de testare nu este disponibil la momentul instruirii, întregul proces de clustering ar trebui repetat prin încetinirea clasificării. În această teză prezentăm o metodă care poate fi utilizată pentru a evita recalcularea kernelului.

3.4 Kerneluri cluster ierarhici cu distanţe pe graf

În construirea kernelului cluster ierarhic am folosit doar prezumţia cluster. Aici extindem kernelul de mai sus pentru a exploata prezumţia de varietate utilizând

un kernel cluster ierarhic bazat pe grafuri. Aproximăm distanţele prin utilizarea drumurilor cele mai scurte care sunt bazate pe grafurile kNN sau ϵNN, similare cu ISOMAP [Tenenbaum et al., 2000]. În acest proces substituim distanţele pe puncte $d(\cdot,\cdot)$ cu distanţele pe graf.

Rezultatul este că algoritmul de clustering ierarhic este precedat de următorii trei paşi:

Algoritmul 3 Kernel cluster ierarhic bazat pe graf

-2: Determinăm cei mai apropiaţi k vecini sau ϵ-vecinătatea fiecărui punct şi a se considera distanţa celorlalte puncte a fi egală cu zero.

-1: Calculăm drumurile cele mai scurte pentru fiecare pereche de puncte – utilizând de exemplu algoritmul lui Dijkstra.

0: Utilizăm aceste distanţe în clusteringul pentru distanţele pe puncte $d(\cdot,\cdot)$.

Am început numărarea în mod deliberat de la -2 pentru a accentua faptul că aceşti paşi *preced* algoritmul din subcapitolul anterior. Subliniem faptul că aceşti paşi sunt opţionali: ei trebuie folosiţi în cazul în care prezumţia de varietate se ţine în setul de date.

Utilizăm aici cele mai scurte drumuri calculate folosind graful de cei mai apropiaţi k vecini sau de ϵ-vecinătate a datelor, astfel – dacă datele se află pe o varietate de dimensiuni scăzute – aproximând distanţele pe această varietate [vezi Bernstein et al., 2000, pentru condiţii].

Graful construit folosind cei mai apropiaţi k vecini sau ϵ-vecinătatea punctelor poate conţine mai multe componente deconectate, de exemplu dacă k sau ϵ este prea mic. În [Tenenbaum et al., 2000] se afirmă că clusterele mici deconectate de componenta "gigantică" conţin de obicei outlieri, şi din această cauză pot fi neglijate. Aici lăsăm detectarea de outlieri ca şi un pas de preprocesare, însemnând faptul că nu neglijăm intrări şi dorim să obţinem un singur component conectat. Urmăm [Yong and Jie, 2005] pentru a obţine un component: prin $\mathbf{M}$ notăm matricea de distanţe pe puncte după rărirea matricei luând cei mai apropiaţi k vecini sau ϵ-vecinătatea a punctelor. În mod similar notăm prin $\mathbf{G}$ matricea de drumuri cele mai scurte bazată pe $\mathbf{M}$ rărit. Dacă vom găsi valori $G_{ij} = \infty$, înseamnă că graful nu este conectat în totalitate. Astfel alegem perechea de $\mathbf{x}_i$ şi $\mathbf{x}_j$ neconectat având distanţa minimă euclidiană, şi le conectăm prin utilizarea unei distanţe relativ mari,

$$\widehat{G}_{ij} = g_{max} + \frac{d_{min}}{d_{max}}$$

unde g_{max} este drumul maxim în $\mathbf{G}$, d_{min} şi d_{max} sunt distanţele minime şi maxime – fie euclidiene, fie bazate pe graf – între punctele de date. După schimbarea $\widehat{G}_{ij}$, actualizăm fiecare element în $\mathbf{G}$ pentru care $G_{k\ell} = \infty$. Pentru orice astfel de

pereche k şi ℓ actualizarea este:

$$G_{k\ell} = \min\{G_{ik} + \widehat{G}_{ij} + G_{j\ell}\,,\; G_{kj} + \widehat{G}_{ij} + G_{i\ell}\}$$

Dacă totuşi rămân componente neconectate, procedura de mai sus trebuie repetată până la obţinerea unui singur component conectat.

4 Kerneluri de reponderare

În continuare propunem trei tehnici uşor diferiţi de reponderare a unui kernel de bază utilizând prezumţia cluster a învăţării semisupervizate. Înaintea acestuia enumerăm combinaţiile kernel care au ca rezultat kerneluri pozitiv semidefinite şi sunt utilizate în cele de mai jos [Lütkepohl, 1996; Abadir and Magnus, 2005]:

(i) Dacă $\mathbf{K}_1$ şi $\mathbf{K}_2$ sunt matrice pozitiv semidefinite, atunci $\mathbf{K}_1 + \mathbf{K}_2$ este este de asemenea pozitiv semidefinită.

(ii) Dacă $\mathbf{K}$ este o matrice pozitiv semidefinită, iar $a > 0$, atunci $a\,\mathbf{K}$ este de asemenea pozitiv semidefinită.

(iii) Dacă $\mathbf{K}_1$ şi $\mathbf{K}_2$ sunt matrice pozitiv semidefinite, atunci $\mathbf{K}_1 \odot \mathbf{K}_2$ este este de asemenea pozitiv semidefinită, unde $\odot$ notează produsul Hadamard.

În acest capitol creăm kerneluri dependente de date prin reponderare: utilizăm prezumţia cluster şi mărirea similarităţii pentru punctele din acelaşi cluster şi scăderea acesteia când clusterele punctelor sunt diferite [Weston et al., 2006].

Dezvoltăm tehnici care reponderesc matricea kernel prin exploatarea structurii cluster a datelor de instruire. Astfel în cazul în care două puncte sunt în acelaşi cluster, similaritatea lor primeşte o pondere ridicată, ≈ 1 sau > 1, iar dacă sunt în diferite clustere, similaritatea lor primeşte o pondere mai scăzută, adică de < 1 sau $\ll 1$, ceea ce noi numim *kernel de reponderare*, sau $k_{rw}(\mathbf{x}_1, \mathbf{x}_2)$. Ponderile de similaritate sunt combinate cu valorile kernelului original sau de bază $k_b(\mathbf{x}_1, \mathbf{x}_2)$, şi vor forma o altă matrice kernel pentru a garanta natura semidefinită a kernelului final. Rezumând, noul kernel cluster este

$$k(\mathbf{x}_1, \mathbf{x}_2) = k_{rw}(\mathbf{x}_1, \mathbf{x}_2)\, k_b(\mathbf{x}_1, \mathbf{x}_2)$$

unde $k_{rw}(\cdot, \cdot)$ este reponderarea şi $k_b(\cdot, \cdot)$ este kernelul de bază. Folosind notaţia matriceală acesta poate fi scris astfel

$$\mathbf{K} = \mathbf{K}_{rw} \odot \mathbf{K}_b$$

Ne confruntăm cu două probleme în construcţia kernelului cluster de mai sus: (i) kernelul de reponderare trebuie să fie pozitiv semidefinit, (ii) matricea kernel de

bază trebuie să fie pozitiv semidefinit şi *pozitiv*. Prima cerinţă este evidentă: este nevoie de ea pentru a garanta natura pozitiv semidefinită a kernelului construit.

A doua condiţie este crucială mai ales pentru că în cazul valorilor negative în matricea kernel de bază este nevoie de efectuarea unei reponderări diferite. Pentru a evita complicaţiile datorate negativităţii, solicităm un kernel de bază pozitiv, $k_b(\mathbf{x}_1, \mathbf{x}_2) \geq 0$. Utilizând produse scalare aceasta poate fi îndeplinită prin deplasarea fiecărei direcţii a datelor în semispaţiul pozitiv, sau pot fi folosite doar kerneluri pozitive ca şi cel gaussian sau kernelurile polinomiale de gradul doi [Schölkopf and Smola, 2002].

În cele de mai jos prezentăm kernelul bagged cluster, apoi propunem trei metode pentru construirea kernelurilor de reponderare pozitiv semidefinite.

4.1 Kernelul bagged cluster

Kernelul bagged cluster, propus în [Weston et al., 2006], repondereşte valorile kernelului de bază ţinând cont de probabilitatea că punctele aparţin aceluiaşi cluster. Pentru a calcula această probabilitate, kernelul bagged cluster utilizează clusteringul k-means [Jain and Dubes, 1988], împreună cu proprietatea că alegerea centrelor clusterului iniţial afectează ieşirea algoritmului. Presupunând că avem N itemi de date şi K clustere, kernelul este construit aşa cum apare în Algoritmul 4 [Weston et al., 2006].

Algoritmul 4 Kernelul bagged cluster

1: Lansăm k-means de t ori, ceea ce rezultă în atribuiri cluster $c_j(\mathbf{x}_i)$, $j = 1, \ldots, t$, $i = 1, \ldots, N$, $c_\cdot(\cdot) \in \{1, \ldots, K\}$.

2: Construim kernelul bagged în următorul fel:

$$k_{bag}(\mathbf{x}, \mathbf{z}) = \frac{\sum_{j=1}^{t} [c_j(\mathbf{x}) = c_j(\mathbf{z})]}{t}$$

Următorul pas este înmulţirea kernelului de bază cu kernelul bagged obţinut în pasul precedent, adică folosind notaţia matriceală:

$$\mathbf{K}_{bag} = \frac{1}{T} \sum_{t=1}^{T} \mathbf{U}^{(t)\prime} \mathbf{U}^{(t)} \quad \text{şi} \quad \mathbf{K} = \mathbf{K}_{bag} \odot \mathbf{K}_b$$

cu matricea de apartenenţă cluster $\mathbf{U}$ de dimensiune $K \times N$, unde prin $\mathbf{U}^{(t)}$ am notat matricea obţinută în pasul t.

4.2 Kernelul de reponderare gaussian

Să supunem că sunt date ieşirile unui algoritm de clustering, matricea de apartenenţă cluster $\mathbf{U}$ de tipul $K \times N$. Distingem cele două familii de metode clusterig: metodele clustering partiţionale şi fuzzy. Metodele partiţionale produc o matrice de apartenenţă cluster în care fiecare coloană conţine o singură valoare de 1, toate celelalte fiind nule; un algoritm fuzzy de clustering produce o matrice, unde fiecare coloană este o distribuţie probabilistică asupra clusterilor, adică u_{ij} conţine probabilitatea că punctul al j-ilea aparţine clusterului i. În ambele cazuri presupunem că cele două puncte aparţin aceluiaşi cluster(e) dacă vectorii de apartenenţă cluster sunt similare sau apropiate unul de celălalt. Definim similaritatea prin kernelul gaussian în felul următor:

$$k_{rw}(\mathbf{x}_1, \mathbf{x}_2) = \exp\left(-\frac{\|\mathbf{u}_{\cdot\mathbf{x}_1} - \mathbf{u}_{\cdot\mathbf{x}_2}\|^2}{2\sigma^2}\right) \tag{4}$$

unde $\mathbf{u}_{\cdot\mathbf{x}}$ notează vectorul de apartenenţă cluster al punctului $\mathbf{x}$, adică coloana lui $\mathbf{x}$ în $\mathbf{U}$. Ştim că matricea construită este pozitiv semidefinită [Schölkopf and Smola, 2002] cu fiecare element fiind între 0 şi 1. În acest caz parametrul σ defineşte cantitatea separării între punctele similare şi disimilare: dacă σ este mare, gaura dintre valorile exprimând similaritate şi disimilaritate devine mai mic, iar pentru un σ mai mic, aceste valori se deplasează mai departe una de cealaltă.

4.3 Kerneluri de reponderare bazate pe produse scalare

O altă posibilitate de utilizare a vectorilor de apartenenţă cluster este definirea următorului kernel de reponderare:

$$\mathbf{K}_{rw} = \mathbf{U}'\mathbf{U} + \alpha\mathbf{1}\mathbf{1}' \tag{5}$$

unde $\mathbf{U}$ notează matricea de apartenenţă cluster şi $\alpha \in [0, 1)$. (Pentru o formă mai generală s-ar putea folosi kernelul polinomial $k(\mathbf{x}_1, \mathbf{x}_2) = (\alpha + \mathbf{x}_1'\mathbf{x}_2)^p$ cu α pozitiv şi p pozitiv natural.) Primul termen – $\mathbf{U}'\mathbf{U}$ – punctează similaritatea pe baza apartenenţei cluster, iar al doilea termen este utilizat pentru a evita similarităţile zero: dacă doi vectori de apartenenţă sunt ortogonale, conducând la valoarea de zero în matricea produselor scalare. Dacă presupunem că clusteringul primit nu este foarte "de încredere", putem folosi o valoare mică α, astfel apartenenţa la un cluster este diminuat cu ajutorul termenului $\alpha\mathbf{1}\mathbf{1}'$.

Este clar faptul că kernelul de reponderare din formula (5) este pozitiv semidefinit: primul şi al doilea termen este un produs tensorial, astfel pozitiv semidefinit, şi în baza proprietăţilor definite în Capitolul 4 putem deduce faptul că kernelul este într-adevăr unul pozitiv semidefinit.

	#eigv	mP	mR	mBEP	mF1	MP	MR	MBEP	MF1
$\chi^2$5209	–	88.46	84.59	86.52	86.48	71.61	61.25	66.43	66.02
$\chi^2$4601	–	87.88	83.09	85.49	85.42	64.21	54.62	59.41	59.03
covarianţă Wikipedia	–	48.95	35.42	42.18	41.10	8.79	5.35	7.07	6.65
LSA	4500	88.05	83.65	85.85	85.80	62.48	54.38	58.43	58.15
LSA+linkuri	4500	87.89	83.71	85.80	85.75	65.11	56.05	60.58	60.24
LSA+PageRank	4000	87.44	83.68	85.56	85.52	59.75	53.86	56.80	56.65

Tabela 1: Performanţa metodei pe corpusul Reuters exprimat în procente. Notaţii: mP=micro-precision, mR=micro-recall, mBEP=micro-breakeven, mF1=micro-F_1, MP=macro-precision, MR=macro-recall, MBEP=macro-breakeven, MF1=macro-F_1

O altă versiune a kernelului din (5) este

$$\mathbf{K}_{\text{rw}} = \beta\,\mathbf{U}'\mathbf{U} + \mathbf{1}\mathbf{1}' \tag{6}$$

unde $\beta \in (0, \infty)$. Aici valorile kernel pentru care matricea de produs scalar a vectorilor de apartenenţă cluster corespund cu zero, prin $\mathbf{1}\mathbf{1}'$ rămân aceleaşi, totuşi dacă punctele se află în acelaşi cluster, $\beta\mathbf{U}'\mathbf{U}$ conferă o pondere mai mare decât zero, astfel această valoare kernel va fi mărită.

Pentru clusteringul fuzzy, unde o coloană din $\mathbf{U}$ conţine probabilităţi de apartenenţă cluster, prima dată normalizăm coloanele pentru a obţine valoarea 1 în cazul vectorilor identici de apartenenţă cluster.

Pentru a construi kernelul de reponderare punctele trebuie clusterate, adică trebuie implicat un algoritm de clustering. În caz optim dacă setăm numărul clusterelor la numărul claselor în problema de clasificare, vom obţine o partiţie care corespunde separării adecvate de clasă a datelor, dar din nefericire acest lucru se întâmplă foarte rar; astfel se poate experimenta cu utilizarea diferitelor numere de cluster.

5 Experimente

În această parte descriem pe scurt experimentele şi rezultatele obţinute.

În experimentele noastre de categorizare am utilizat depozitul Wikipedia (*dump*) în limba engleză din luna noiembrie 2006[3] care conţinea aproximativ 1.6 milioane de articole, care este egal cu aproximativ 8 gigabyte de date textuale – în afara imaginilor şi altor documente adiţionale, dar incluzând tagurile XML. În mod asemănător cu [Gabrilovich and Markovitch, 2007], pentru a filtra articolele irelevante (paginile de categorizare, pagini interne, cioturi Wikipedia [vezi `Wikipedia:Stub`], etc.), am eliminat următoarele articole:

- articole care conţin mai puţin de 500 de cuvinte

[3] `http://download.wikimedia.org/enwiki`

- articole care conţin mai puţin de 5 linkuri forward către alte articole Wikipedia

După această reducere am construit un index inversat pentru cuvintele care apar în Wikipedia, în afara cuvintelor stop şi a celor mai frecvente 300 de cuvinte. Am exclus de asemenea cuvinte care apar în mai puţin de 10 articole. Excluderea acestor cuvinte a fost efectuată pentru a ajuta filtrarea termenilor irelevanţi de indexare, fiindcă am presupus că un cuvânt care nu este important în Wikipedia este de asemenea irelevant în indexarea corpusului de clasificare a documentelor. Astfel au rămas 327 653 articlole.

Pentru testarea kernelului Wikipedia asupra categorizării de documente am utilizat corpusul Reuters-21578, împărţirea ModApté cu $90 + 1$ categorii, în cadrul căruia categoria "unknown" (*necunoscut*) nu a fost utilizată.

În acelaşi timp am utilizat o metodă de selectare a termenilor de filtrare pentru a selecta cei mai relevanţi termeni din corpusul Reuters. În acest scop am utilizat selectarea de termeni χ^2 şi am selectat 5209 termeni (am obţinut un bun rezultat cu acest număr de trăsături în [Minier et al., 2006]). Pentru aceste cuvinte am construit vectori de cuvinte reprezentând distribuţia acestor cuvinte în articolele Wikipedia. Toate cuvintele extrase din articolele Wikipedia şi din corpusul Reuters au fost prelucrate utilizând algoritmul lui Porter [Baeza-Yates and Ribeiro-Neto, 1999].

După construirea vectorilor de documente am folosit SVM-uri pentru a învăţa datele de instruire; în acest scop am utilizat implementarea LIBSVM [Chang and Lin, 2001].

Rezultatele sunt prezentate în Tabela 1. Pentru evaluarea sistemului am utilizat măsura *precision*, *recall*, punctul de echilibru (*breakeven*) *precision–recall* şi F_1. $\chi^2 5209$ şi $\chi^2 4601$ prezintă rezultatele obţinute utilizând metoda de selectare a termenilor χ^2. Rândurile covarianţei Wikipedia, LSA, LSA+linkuri şi LSA+PageRank arată rezultatele obţinute în urma utilizării kernelului Wikipedia, adică kernelul Wikipedia fără reducţie, kernelul Wikipedia cu LSA, kernelul Wikipedia cu LSA şi cu utilizarea matricei link, şi la sfârşit kernelul Wikipedia cu LSA şi ponderarea conceptelor cu ajutorul algoritmului PageRank. Coloana #eigv arată numărul vectorilor proprii alese.

Cele mai bune rezultate au fost obţinute prin utilizarea selecţiei χ^2 a termenilor, iar pe locul doi se află utilizarea kernelului Wikipedia cu LSA.

Tabela 2 prezintă rezultatele obţinute în urma utilizării kernelurilor dependente de date pe seturile de date USPS, Digit1, COIL2 şi Text [Chapelle et al., 2006]. Am utilizat precizia (*accuracy*) ca şi măsura de evaluare, iar rezultatele sunt date în procente. În cazul fiecărui set de date am indicat cele mai bune rezultate obţinute. Acest lucru nu a fost posibil în tabela anterioară fiindcă acolo am avut două

	USPS		Digit1		COIL2		Text	
	10	100	10	100	10	100	10	100
linear	72.82	86.43	81.07	90.86	60.74	80.43	58.26	67.86
Gaussian	80.07	89.71	56.11	93.86	57.38	82.50	59.06	56.43
ISOMAP	85.10	86.71	94.43	97.43	62.62	80.64	59.80	72.43
vecinătate	76.31	94.14	87.11	94.21	64.43	84.43	51.68	62.79
bagged	87.38	92.79	93.29	96.93	71.28	85.57	63.29	66.14
multi-type, pas	80.07	92.86	91.01	91.29	55.77	84.86	53.56	74.79
multi-type, pas linear	80.07	92.86	91.01	91.36	55.77	84.86	53.02	75.29
multi-type, polinomial	80.07	80.29	48.86	65.07	54.23	82.29	50.60	56.71
HCK, single	80.07	81.79	48.86	70.21	67.85	96.00	66.78	73.14
HCK, complete	82.01	89.50	60.67	89.71	55.64	86.36	50.27	49.57
HCK, average	81.48	92.86	71.75	93.79	68.05	91.71	64.63	50.14
gHCK, single	80.07	81.79	48.86	70.21	60.60	93.86	66.78	73.14
gHCK, complete	88.26	95.64	75.50	93.71	68.52	88.79	56.17	67.71
gHCK, average	89.26	95.64	94.70	95.21	60.54	90.64	47.32	66.86
RCK1, k-means	84.45	92.98	83.14	94.28	58.55	83.76	–	–
RCK1, ierarhic	86.17	95.29	89.06	94.94	62.08	85.93	62.35	68.07
RCK1, spectral	81.43	90.87	88.32	95.20	58.22	83.83	63.26	66.93
RCK2, k-means	83.86	92.45	84.58	94.08	58.95	83.76	–	–
RCK2, ierarhic	86.11	95.50	89.06	95.29	62.08	85.64	61.28	71.14
RCK2, spectral	81.63	91.39	88.32	94.64	58.03	83.37	61.50	70.07
RCK3, k-means	83.66	92.59	84.13	92.96	58.60	83.28	–	–
RCK3, ierarhic	84.97	95.29	89.06	94.57	62.95	86.07	59.13	71.21
RCK3, spectral	81.16	91.56	88.32	94.73	55.83	83.20	59.26	71.00

Tabela 2: Rezultatele de precizie (*accuracy*) după utilizarea diferitelor kerneluri. Rezultatele sunt prezentate în procente. Pentru fiecare set de date cele mai bune rezultate sunt înramate.

linii de referinţă.

Primele două rânduri conţin rezultatele liniilor de referinţă obţinute prin utilizarea kernelurilor lineare şi gaussiene. În principal am dorit să îmbunătăţim aceste rezultate. Hiperparametrul kernelului gaussian a fost fixat prin aplicarea a două metode – vezi [Chapelle et al., 2006] şi [Zhu and Ghahramani, 2002] – pentru estimarea valorii de bază σ_0, iar apoi am utilizat o validare încrucişată de 10 ori pe setul mai mare etichetat conţinând 100 de puncte etichetate utilizând valorile $[\sigma_0/8\ \sigma_0/4\ \sigma_0/2\ \sigma_0\ 2\sigma_0\ 4\sigma_0\ 8\sigma_0]$. Pentru parametrii optimali găsiţi de cele două metode s-a stabilit o medie de $[4.0039\ 1.2555\ 265.3732\ 0.9041]$ pentru seturile de date USPS, Digit1, COIL2, respectiv Text.

Următoarele 6 rânduri arată rezultatele obţinute în urma aplicării kernelului ISOMAP [Tenenbaum et al., 2000], kernelului de vecinătate [Weston et al., 2006], kernelului bagged cluster [Weston et al., 2006] şi kernelul cluster multi-type cu

diferite funcţii de transfer [Chapelle et al., 2002].

ISOMAP este o tehnică neliniară de reducere a dimensiunii care combină metodele PCA, MDS şi aproximarea de varietate (*manifold approximation*). Pentru kernelul ISOMAP am utilizat un graf kNN cu k setat la 7.

Kernelul de vecinătate este bazat pe prezumţia cluster şi reprezintă fiecare punct ca şi media vecinilor. Aici am încercat mai multe configuraţii ai parametrilor prin schimbarea numărului vecinilor de la 2 la 7; în tabelă apar cele mai bune rezultate (am omis cei mai buni parametri din cauza lipsei de spaţiu). În cazul kernelului bagged cluster am setat parametrul $t = 20$, şi în mod similar în cazul kernelului precedent am experimentat prin utilizarea diferitelor numere de clustere: K a fost ales din mulţimea $\{2, 3, \ldots, 7\}$.

Kernelul cluster multi-type combină cu succes mai multe tehnici ca şi clusteringul spectral, kernel PCA şi mersurile aleatorii; funcţiile de transfer determină modul în care valorile proprii sunt transformate. În cazul acestui kernel am utilizat parametrii fixaţi:

- funcţia transfer de pas (*step transfer function*): au fost utilizate cele mai mari valori proprii $\ell + 10$, unde ℓ este numărul exemplelor etichetate;
- funcţia transfer de pas linear: au fost utilizate cele mai mari valori proprii $\ell + 10$;
- funcţia transfer polinomial: $t = 5$ (folosind notarea din [Chapelle et al., 2002]).

În cazul kernelurilor de vecinătate şi kernelurilor bagged cluster am utilizat kernelul linear pentru kernelul de bază.

Următorii 15 rânduri – sub linea a doua orizontală – prezintă rezultatele obţinute în urma utilizării kernelurilor noştri. HCK şi gHCK notează clusterul ierarhic respectiv clusterul ierarhic bazat pe grafuri. În cazul gHCK, ca şi în cazul kernelului ISOMAP, am utilizat un graf kNN cu $k = 7$. Cuvintele *single*, *complete* şi *average* indică distanţele de legătură utilizate. RCK1, RCK2 şi RCK3 notează kerneluri cluster de reponderare definite în formulele (4), (5), respectiv (6). Aici am experimentat cu trei tehnici de clustering: clusteringul k-means, ierarhic şi spectral. Din cauza că ieşirea metodei k-means depinde puternic de centrul iniţial al clusterului, am repetat clusteringul k-means de 10 ori şi am determinat valorile de precizie ca media acestor 10 încercări. Pentru setul de date Text algoritmul k-means s-a dovedit a fi inadecvat din cauza dimensionalităţii sale mari (în acest caz inadecvat înseamnă timp îndelungat de acţiune). În cazul clusteringului ierarhic am utilizat următoarele metode: distanţe de legături single, complete, average, weighted average, centroid (legături medii pe grupă), median (legături medii ponderate pe grupă) şi ward (metoda lui Ward). În tabelă am prezentat doar cele

mai bune metode. În încheiere am utilizat clusterigul spectral pentru obţinerea kernelului: am utilizat metoda spectrală de clustering normalizat – varianta lui Shi şi Malik [von Luxburg, 2006] cu clustering k-means. Astfel, ca şi în cazul k-meansului convenţional, am determinat valoarea de precizie cu determinarea mediei rezultatelor pe baza a 10 încercări. În cazul fiecărui algoritm de clustering numărul K a clusterelor a fost ales din mulţimea $\{2, 3, \ldots, 7\}$ (ceea ce nu apare în tabelă). Ca şi în cazul kernelurilor de vecinătate şi de bagged cluster, am utilizat aici kernelul linear pentru kernelul de bază. Parametriile de mai jos ai kernelurilor de reponderare au fost fixate în decursul experimentelor:

- kernel gaussian de reponderare definit în (4): $\sigma = 1/\sqrt{2}$
- kernel de reponderare definit în (5): $\alpha = 0.3$
- kernel de reponderare definit în (6): $\beta = 0.5$

Prezentarea detaliată a rezultatelor poate fi găsită în capitolele corespunzătoare ale tezei.

6 Concluzii

Kernelurile dependente de date combină metodele kernel şi de învăţare semisupervizată prin construirea kernelurilor – astfel în mod implicit dând o nouă reprezentare a noilor exemple – cu utilizarea seturilor de date etichetate şi neetichetate. Numim aceste kerneluri dependente de date pentru că funcţia kernel depinde pe setul complet al punctelor etichetate şi neetichetate disponibile în faza de instruire. Matematic vorbind, dacă $D_1 \neq D_2$, $\mathbf{x}, \mathbf{z} \in D_1 \cap D_2$,

$$k(\mathbf{x}, \mathbf{z}; D_1) \eqsim k(\mathbf{x}, \mathbf{z}; D_2)$$

unde "$\eqsim$" se citeşte "nu neaparat egal". În această teză am propus kerneluri dependente de date şi metode de construcţie a kernelurilor în condiţii de învăţare supervizată şi semisupervizată. Am introdus următoarele kerneluri dependente de date:

- kerneluri de categorizare a documentelor bazate pe Wikipedia (Capitolul 5)
- kerneluri cluster ierarhici (Capitolul 6)
- kerneluri cluster de reponderare (Capitolul 7)

Categorizarea textelor este o sarcină de utilizare a unor exemple de instruire pentru construirea unei entităţi care învaţă supervizat, care este capabil să atribuie

una sau mai multe categorii documentelor în limbaje naturale. Kernelul bazat pe Wikipedia este construit prin aproximarea matricei Wikipedia cu o matrice de rang inferior pentru a filtra termenii irelevanţi, şi care în acelaşi timp asigură o reprezentare mai bogată a documentelor. Metoda propusă poate fi interpretată ca şi o simplă înlocuire a matricei de covarianţă a termenilor, clădită prin utilizarea documentelor de instruire, prin matricea de covarianţă indus de Wikipedia. Din cauză că unii termeni de indexare aleşi de metoda de selectare a trăsăturilor χ^2 nu au fost regăsiţi în Wikipedia, avem aici două linii de bază: una cu 5209 şi cealaltă cu 4601 termeni. Metoda noastră a întrecut uşor numai metoda de selectare a trăsăturilor χ^2 cu 4601 termeni. Motivul rezultatelor inferioare în acest caz poate fi specificitatea metodei de selectare a termenului de bază, diferenţele de distribuţie ale cuvintelor în Reuters-21578 şi Wikipedia, etc. Posibilele căi de cercetare în această direcţie includ alegerea altor metode de selecţie a termenilor, găsirea reprezentărilor documentelor mai potrivite şi efectuarea experimentelor şi pe alte corpusuri TC.

Kernelurile cluster ierarhice sunt construiţi prin aplicarea proprietăţii ultrametrice a unor distanţe de legături utilizate în metodele de clustering aglomerativ. În capitolul 6 am propus două kerneluri de acest fel: unul care este bazat pe prezumţia cluster şi un altul care utilizează şi prezumţia de varietate. Desigur – ca şi în cazul tehnicilor semisupervizate – acestea sunt aşteptate să îmbunătăţească performanţa chiar dacă doar una dintre prezumţii se dovedeşte a fi corectă. Am propus în acelaşi timp un cadru general pentru construcţia kernelurilor cluster. Kernelurile au fost testaţi pe diferite seturi de date folosite pentru benchmark în [Chapelle et al., 2006]. HCK şi gHCK au întrecut în mod semnificativ kernelurile lineari şi gaussieni utilizaţi ca măsuri de referinţă; cea mai semnificativă îmbunătăţire măsurată între cea mai bună linie de bază şi rezultatele HCK/gHCK a fost de 13.5% în cazul setului de date COIL2. În acelaşi timp am comparat kernelurile propuse cu alte kerneluri dependente de date şi le-am găsit comparabile cu aceştia: HCK şi gHCK au produs de mai multe ori rezultate mai bune decât celelate kerneluri. Pe baza experimentelor putem concluziona că aplicarea HCK şi gHCK este utilă pentru aproape toate seturile de date. Este loc pentru perfecţionare aici, de asemenea: testarea metodei propuse pentru calcularea valorilor kernel pentru punctele nevăzute.

În Capitolul 7 am prezentat trei scheme de reponderare pentru construcţia kernelurilor cluster de reponderare (RCK). Idea de bază este împrumutată de la kernelul bagged cluster: este construită o matrice (un kernel) pozitiv semidefinită prin care valorile de bază ale kernelului sunt reponderate: de exemplu un kernel linear sau gaussian. Am studiat două tipuri de kerneluri de reponderare: kernelul de reponderare gaussian şi kernelurile de reponderare care utilizează produsele scalare ale vectorilor de apartenenţă cluster. Experimentele arată rezultate similare şi în

cazul acestor kerneluri, ceea ce întrec în mod evident rezultatele de referinţă, şi prezintă aceste kerneluri cluster comparabili cu kernelurile existente dependente de date.

Rezultatele cuprinse în această teză arată că prin utilizarea kernelurilor dependente de date metodele convenţionale kernel pot fi depăşite în mod semnificativ. Experimentele arată faptul că kernelurile propuşi pot fi utilizate pe o varietate largă a seturilor de date. Din această cauză rezultatele cercetării întreprinse în prezenta teză pot servi ca mijloace utile şi eficiente în multe domenii în care sunt folosiţi algoritmi de clasificare.

Bibliografia tezei de doctorat

K. M. Abadir and J. R. Magnus. *Matrix Algebra*. Cambridge University Press, 2005.

A. Aizerman, E. M. Braverman, and L. I. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley, 1999.

S. Banerjee and T. Pedersen. An adapted lesk algorithm for word sense disambiguation using wordnet. In *CICLing*, pages 136–145, 2002.

S. Basu. *Semi-supervised Clustering: Probabilistic Models, Algorithms and Experiments*. PhD thesis, The University of Texas at Austin, 2005.

M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

A. Berger. Error-correcting output coding for text classification. In *Proceedings of IJCAI-99 Workshop on Machine Learning for Information Filtering*, 1999.

P. Berkhin. Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA, 2002.

M. Bernstein, V. de Silva, J. C. Langford, and J. B. Tenenbaum. Graph approximations to geodesics on embedded manifolds, 2000.

T. D. Bie. *Semi-Supervised Learning Based On Kernel Methods And Graph Cut Algorithms*. PhD thesis, Katholieke Universiteit Leuven, Kasteelpark Arenberg 10, 3001 Leuven (Heverlee), 2005.

T. D. Bie, J. A. K. Suykens, and B. D. Moor. Learning from general label constraints. In A. L. N. Fred, T. Caelli, R. P. W. Duin, A. C. Campilho, and D. de Ridder, editors, *Structural, Syntactic, and Statistical Pattern Recognition, Joint IAPR International Workshops, SSPR 2004 and SPR 2004, Lisbon, Portugal, August 18-20, 2004 Proceedings*, volume 3138 of *Lecture Notes in Computer Science*, pages 671–679. Springer, 2004. ISBN 3-540-22570-6.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer Verlag, New York, 2006.

A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In *Proc. 18th International Conf. on Machine Learning*, pages 19–26. Morgan Kaufmann, San Francisco, CA, 2001.

Z. Bodó. Hierarchical cluster kernels for supervised and semi-supervised learning. In *Proceedings of the 4nd International Conference on Intelligent Computer Communication and Processing (ICCP 2008)*, pages 9–16. IEEE, August 28–30 2008.

Z. Bodó and Z. Minier. On supervised and semi-supervised k-nearest neighbor algorithms. In *Proceedings of the 7th Joint Conference on Mathematics and Computer Science*, volume LIII, pages 79–92. Studia Universitatis Babeş-Bolyai, Series Informatica, July 2008.

Z. Bodó and Z. Minier. Semi-supervised feature selections with svms. In *Proceedings of the conference Knowledge Engineering: Principles and Techniques (KEPT 2009)*, pages 159–162. Presa Universitară Clujeană, July 2–4 2009. Special Issue of Studia Universitatis Babeş-Bolyai, Series Informatica.

Z. Bodó, Z. Minier, and L. Csató. Text categorization experiments using Wikipedia. In *Proceedings of the conference Knowledge Engineering: Principles and Techniques (KEPT 2007)*, pages 66–72. Presa Universitară Clujeană, June 6–7 2007. Special Issue of Studia Universitatis Babeş-Bolyai, Series Informatica.

I. Borg and P. J. F. Groenen. *Modern multidimensional scaling, 2nd edition*. Springer-Verlag, New York, 2005.

B. E. Boser, I. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. *Computational Learning Theory*, 5:144–152, 1992.

C. J. C. Burges. A tutorial on support vector machines for pattern recognition. In *Knowledge Discovery and Data Mining*, volume 2, pages 121–167. 1998.

R. Busa-Fekete and A. Kocsor. Locally linear embedding and its variants for feature extraction. In *IEEE*

International Workshop on Soft Computing Applications, SOFA 2005, 2005.

C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

O. Chapelle, J. Weston, and B. Schölkopf. Cluster kernels for semi-supervised learning. In S. Becker, S. Thrun, and K. Obermayer, editors, *NIPS*, pages 585–592. MIT Press, 2002. ISBN 0-262-02550-7.

O. Chapelle, B. Schölkopf, and A. Zien. *Semi-Supervised Learning*. MIT Press, Sept. 2006. Web page: http://www.kyb.tuebingen.mpg.de/ssl-book/.

Chung. Spectral graph theory (reprinted with corrections). In *CBMS: Conference Board of the Mathematical Sciences, Regional Conference Series*, 1997.

D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996. ISSN 1076-9757.

C. Corley, A. Csomai, and R. Mihalcea. Text semantic similarity, with applications. In *Proceedings of International Conference Recent Advances in Natural Language Processing (RANLP 2005)*, September 2005.

T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 2001.

T. Cover and J. Thomas. *Elements of Information Theory, Second Edition*. Wiley-Interscience, 2006.

T. M. Cover and P. E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, IT-13, 1967.

K. Crammer and Y. Singer. A new family of online algorithms for category ranking. In *The 25th Annual International ACM SIGIR Conference*. ACM, 2002.

N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. S. Kandola. On kernel-target alignment. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, pages 367–373. MIT Press, 2001.

N. Cristianini, J. Shawe-Taylor, and H. Lodhi. Latent semantic kernels. *J. Intell. Inf. Syst*, 18(2-3): 127–152, 2002.

N. Cristianini, J. Kandola, A. Vinokourov, and J. Shawe-Taylor. Kernel methods for text processing. In J. A. K. Suykens, G. Horváth, S. Basu, C. Micchelli, and J. Vandewalle, editors, *Advances in Learning Theory: Methods, Models and Applications*, pages 197–221. IOS Press, 2003.

L. Csató and Z. Bodó. *Neurális hálók és a gépi tanulás módszerei (Reţele neurale şi metode de instruire automată)*. Presa Universitară Clujeană, Cluj-Napoca, 2008.

L. Csató and Z. Bodó. Decomposition methods for label propagation. In *Proceedings of the conference Knowledge Engineering: Principles and Techniques (KEPT 2009)*, pages 127–130. Presa Universitară Clujeană, July 2–4 2009. Special Issue of Studia Universitatis Babeş-Bolyai, Series Informatica.

F. Debole and F. Sebastiani. An analysis of the relative hardness of reuters-21578 subsets. *Journal of the American Society for Information Science and Technology*, 56:971–974, 2004.

S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41, June 1990.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38, 1977.

I. S. Dhillon, Y. Guan, and B. Kulis. Kernel k-means: spectral clustering and normalized cuts. In *ACM SIGKDD – Knowledge discovery and data mining*, pages 551–556, 2004.

T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

L. Diosan, M. Oltean, A. Rogozan, and J.-P. Pécuchet. Improving SVM performance using a linear combination of kernels. In B. Beliczynski, A. Dzielinski, M. Iwanowski, and B. Ribeiro, editors, *Adaptive and Natural Computing Algorithms, 8th International Conference, ICANNGA 2007, Warsaw, Poland, April 11-14, 2007, Proceedings, Part II*, volume 4432 of *Lecture Notes in Computer Science*, pages

218–227. Springer, 2007. ISBN 978-3-540-71590-0.

R. Duda, P. Hart, and D. Stork. *Pattern Classification*. John Wiley and Sons, 2001. 0-471-05669-3.

W. K. Estes. *Classification and Cognition*. Oxford University Press, 1994. ISBN 9780195073355.

B. Fischer, V. Roth, and J. M. Buhmann. Clustering with the connectivity kernel. In S. Thrun, L. K. Saul, and B. Schölkopf, editors, *NIPS*. MIT Press, 2003. ISBN 0-262-20152-6.

N. Fuhr, S. Hartmann, G. Knorz, G. Lustig, M. Schwantner, and K. Tzeras. AIR/X – a rule-based multistage indexing system for large subject fields. In A. Lichnerowicz, editor, *Proceedings of RIAO-91, 3rd International Conference "Recherche d'Information Assistee par Ordinateur"*, pages 606–623, Barcelona, ES, 1991. Elsevier Science Publishers, Amsterdam, NL.

J. Fürnkranz. Round robin classification. *Journal of Machine Learning Research*, 2:721–747, 2002.

E. Gabrilovich and S. Markovitch. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of The 20th International Joint Conference on Artificial Intelligence (IJCAI)*, January 2007.

I. P. Gent, P. Prosser, B. M. Smith, and W. Wei. Supertree construction with constraint programming. In *ICCP: International Conference on Constraint Programming (CP), LNCS*, 2003.

A. Gliozzo and C. Strapparava. Domain kernels for text categorization. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 56–63, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

G. H. Golub and C. F. Van Loan. *Matrix Computations, 3nd Edition*. The Johns Hopkins University Press, Baltimore, MD, 1996.

C. M. Grinstead and J. L. Snell. *Introduction to Probability*. AMS, 2003.

I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

J. Ham, D. D. Lee, S. Mika, and B. Schölkopf. A kernel view of the dimensionality reduction of manifolds. In C. E. Brodley, editor, *Machine Learning, Proceedings of the Twenty-first International Conference (ICML 2004), Banff, Alberta, Canada, July 4-8, 2004*, volume 69 of *ACM International Conference Proceeding Series*. ACM, 2004.

G. Hirst and D. St-onge. Lexical chains as representations of context for the detection and correction of malapropisms, Aug. 31 1997.

P. Jackson and I. Moulinier. *Natural Language Processing for Online Applications: Text Retrieval, Extraction & Categorization*. John Benjamins, 2002. ISBN 90-272-4989-X.

A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice-Hall, 1988.

A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *CSURV: Computing Surveys*, 31, 1999.

J. J. Jiang and D. W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. *CoRR*, cmp-lg/9709008, 1997. informal publication.

T. Joachims. Text categorization with support vector machines: Learning with many relevant features. Technical Report LS VIII-Report, Universität Dortmund, Dortmund, Germany, 1997.

T. Joachims. Text categorization with suport vector machines: Learning with many relevant features. In C. Nedellec and C. Rouveirol, editors, *Machine Learning: ECML-98, 10th European Conference on Machine Learning, Chemnitz, Germany, April 21-23, 1998, Proceedings*, volume 1398 of *Lecture Notes in Computer Science*, pages 137–142. Springer, 1998. ISBN 3-540-64417-2.

I. T. Jolliffe. *Principal Component Analysis*. Series in Statistics. Springer Verlag, 2002.

J. Kandola, J. Shawe-Taylor, and N. Cristianini. Optimizing kernel alignment over combinations of kernels. Technical Report 2002-121, Department of Computer Science, Royal Holloway, University of London, UK, 2002a.

J. S. Kandola, J. Shawe-Taylor, and N. Cristianini. Learning semantic similarity. In S. Becker, S. Thrun, and K. Obermayer, editors, *NIPS*, pages 657–664. MIT Press, 2002b. ISBN 0-262-02550-7.

C. Leacock and M. Chodorow. Combining local context and WordNet similarity for word sense identi-

fication. In C. Fellbaum, editor, *WordNet: An Electronic Lexical Database*, pages 265–283. The MIT Press, Cambridge, Massachusetts, 1998.

M. D. Lee, B. Pincombe, and M. Welsh. An empirical evaluation of models of text document similarity. In *CogSci2005*, pages 1254–1259, 2005.

C. Leslie and R. Kuang. Fast kernels for inexact string matching. In *COLT: Proceedings of the Workshop on Computational Learning Theory, Morgan Kaufmann Publishers*, 2003.

C. S. Leslie, E. Eskin, J. Weston, and W. S. Noble. Mismatch string kernels for SVM protein classification. In S. Becker, S. Thrun, and K. Obermayer, editors, *NIPS*, pages 1417–1424. MIT Press, 2002. ISBN 0-262-02550-7.

D. D. Lewis. Evaluating and optimizing autonomous text classification systems. In *SIGIR '95*, pages 246–254, New York, NY, USA, 1995. ACM Press.

D. Lin. An information-theoretic definition of similarity. In *ICML*, pages 296–304, 1998.

H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444, 2002.

H. Lütkepohl. *Handbook of matrices*. John Wiley & Sons Ltd., Chichester, 1996. ISBN 0-471-97015-8.

G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. Five papers on wordnet. Technical Report, Cognitive Science Laboratory, Princeton University, 1993.

Z. Minier, Z. Bodó, and L. Csató. Segmentation-based feature selection for text categorization. In *Proceedings of the 2nd International Conference on Intelligent Computer Communication and Processing (ICCP 2006)*, pages 53–59. IEEE, September 1–2 2006.

Z. Minier, Z. Bodó, and L. Csató. Wikipedia-based kernels for text categorization. In *Proceedings of the 9th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC 2007)*, pages 157–164. IEEE, September 26–29 2007.

T. M. Mitchell. The discipline of machine learning. Technical Report CMU-ML-06-108, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 2006.

R. Morelos-Zaragoza. *The Art of Error Correcting Coding*. John Wiley and Sons, Inc., pub-WILEY:adr, 2002. ISBN 0-471-49581-6.

A. Moschitti. A study on optimal parameter tuning for rocchio text classifier. In F. Sebastiani, editor, *Advances in Information Retrieval, 25th European Conference on IR Research, ECIR 2003, Pisa, Italy, April 14-16, 2003, Proceedings*, volume 2633 of *Lecture Notes in Computer Science*, pages 420–435. Springer, 2003. ISBN 3-540-01274-5.

A. Y. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

K. Nigam. *Using unlabeled data to improve text classification*. PhD thesis, Carnegie Mellon University, 2001.

K. Nigam, A. McCallum, S. Thrun, and T. M. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134, 2000.

L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

S. Patwardhan, S. Banerjee, and T. Pedersen. Using measures of semantic relatedness for word sense disambiguation. In A. F. Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing, 4th International Conference, CICLing 2003, Mexico City, Mexico, February 16-22, 2003, Proceedings*, volume 2588 of *Lecture Notes in Computer Science*, pages 241–257. Springer, 2003. ISBN 3-540-00532-3.

J. C. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin DAGs for multiclass classification. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *NIPS*, pages 547–553. The MIT Press, 1999. ISBN 0-262-19450-3.

P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *IJCAI*, pages

448–453, 1995.

C. J. V. Rijsbergen. *Information Retrieval*. Butterworths, 1979.

S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

S. Russel and P. Norvig. *Artificial Intelligence: a Modern Approach*. Prentice-Hall, 1995.

G. Salton, A. Wong, and A. C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18:229–237, 1975.

L. K. Saul and S. T. Roweis. An introduction to locally linear embedding, 2001.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. The MIT Press, Cambridge, MA, 2002.

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. Technical Report 44, Max-Planck Institut für biologische Kybernetik, Arbeitsgruppe Bülthoff, Spemannstrasse 38, 2076 Tobingen, Germany, Dec. 1996.

B. Schölkopf, A. J. Smola, and K.-R. Müller. Kernel principal component analysis. *Advances in kernel methods: support vector learning*, pages 327–352, 1999.

H. Schütze. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123, 1998.

F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

J. Shlens. A tutorial on principal component analysis, 2005.

V. Sindhwani, P. Niyogi, and M. Belkin. Beyond the point cloud: from transductive to semi-supervised learning. In L. D. Raedt and S. Wrobel, editors, *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005*, volume 119 of *ACM International Conference Proceeding Series*, pages 824–831. ACM, 2005. ISBN 1-59593-180-5.

D. Sloughter. The calculus of functions of several variables, 2001.

J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, Dec. 2000.

H. tien Lin and C. jen Lin. A study on sigmoid kernels for SVM and the training of non-PSD kernels by SMO-type methods, 2003.

D. Tikk. *Szövegbányászat*. Typotex, Budapest, 2007.

V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

G. Varelas. Semantic similarity methods in wordnet and their application to information retrieval on the web. Technical Report TR-TUC-ISL-01-2005, Technical Univ. of Crete (TUC), Dept. of Electronic and Computer Engineering, Chania, Crete, Greece, 2005.

S. Vishwanathan and N. M. Murty. Kernel enabled K-means algorithm. Technical report, The Indian Institute of Science, Bangalore, 2002.

S. V. N. Vishwanathan and A. J. Smola. Fast kernels for string and tree matching. In S. Becker, S. Thrun, and K. Obermayer, editors, *NIPS*, pages 569–576. MIT Press, 2002. ISBN 0-262-02550-7.

S. V. N. Vishwanathan, K. M. Borgwardt, O. Guttman, and A. J. Smola. Kernel extrapolation. *Neurocomputing*, 69(7-9):721–729, 2006.

U. von Luxburg. A tutorial on spectral clustering. Technical Report 149, Max Planck Institute for Biological Cybernetics, August 2006.

J. Voss. Measuring wikipedia, 2005.

J. H. Ward, Jr. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58:236–244, 1963.

C. wei Hsu and C. jen Lin. A comparison of methods for multi-class support vector machines, 2001.

J. Weston and C. Watkins. Support vector machines for multiclass pattern recognition. In *Proceedings of the Seventh European Symposium On Artificial Neural Networks*, 4 1999.

J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping. Use of the zero-norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, 2003.

J. Weston, C. Leslie, D. Zhou, A. Elisseeff, and W. S. Noble. Semi-supervised protein classification

using cluster kernels, 2004.

J. Weston, C. Leslie, E. Ie, and W. S. Noble. Semi-supervised protein classification using cluster kernels. In O. Chapelle, B. Schölkopf, and A. Zien, editors, *Semi-Supervised Learning*, chapter 19, pages 343–360. MIT Press, 2006.

S. K. M. Wong, W. Ziarko, and P. C. N. Wong. Generalized vector spaces model in information retrieval, proceedings of the 8th annual international ACM SIGIR conference on research and development in information retrieval. In J. M. Tague, editor, *Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 18–25, Montreal, Quebec, Canada, June 1985. ACM Press.

S. K. M. Wong, W. Ziarko, V. V. Raghavan, and P. C. N. Wong. On modeling of information retrieval concepts in vector spaces. *ACM Trans. on Database Sys.*, 12(2):299, June 1987.

B. Y. Wu and K.-M. Chao. *Spanning Trees and Optimization Problems*. Chapman and Hall/CRC, Boca Raton, Florida, 2004.

Z. Wu and M. S. Palmer. Verb semantics and lexical selection. In *ACL*, pages 133–138, 1994.

Y. Yang and C. G. Chute. A linear least squares fit mapping method for information retrieval from natural language texts. In *COLING*, pages 447–453, 1992.

Y. Yang and C. G. Chute. An example-based mapping method for text categorization and retrieval. *ACM Transactions on Information Systems*, 12(3):252–295, July 1994.

Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *International Conference on Machine Learning*, pages 412–420, 1997.

Z.-P. Yang, X. Zhang, and C.-G. Cao. Inequalities involving khatri-rao products of hermitian matrices. *The Korean Journal of Computational & Applied Mathematics*, 9(1):125–133, 2002. ISSN 1229-9502.

Q. Yong and Y. Jie. Geodesic distance for support vector machines. *Acta Automatica Sinica*, 31(2): 202–208, 2005.

J. A. Zdziarski. *Ending spam: Bayesian content filtering and the art of statistical language classification*. No Starch Press, pub-NO-STARCH:adr, 2005. ISBN 1-59327-052-6.

D. Zhou, B. Schölkopf, and T. Hofmann. Semi-supervised learning on directed graphs. In *NIPS*, 2004.

X. Zhu. *Semi-supervised learning with graphs*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, 2005. Chair-John Lafferty and Chair-Ronald Rosenfeld.

X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical Report CMU-CALD-02-107, Carnegie Mellon University, 2002.

X. Zhu, T. J. Rogers, R. Qian, and C. Kalish. Humans perform semi-supervised classification too. In *AAAI*, page 864. AAAI Press, 2007. ISBN 978-1-57735-323-2.