CLASIFICAREA FORMELOR PLANE UTILIZÂND DISTANȚE SINTACTICE

dr. ing. Ovidiu Grigore
dr. ing. Octavian Grigore
Universitatea Politehnica *Bucureşti*

Rezumat: În lucrare sunt prezentate metode de clasificare sintactice, bazate pe compararea cu modele ("matching"). În general, formele plane (caractere, imagini de obiecte etc.) au o alcătuire complexă, care, într-o abordare globală de tip statistic, conduc la discriminări greoaie, cu rezultate destul de slabe. Din acest motiv, s-a utilizat o descriere sintactică a formelor, care pune în evidență modul lor de alcătuire atât din punct de vedere al componentelor, cât şi al relațiilor dintre acestea. Pentru clasificarea structurilor clasice rezultate, s-au folosit distanțe corespunzătoare. S-a introdus o nouă distanță sintactică *normată*, care a condus la îmbunătățirea rezultatelor de clasificare obținute prin utilizarea distanței clasice Wagner-Fischer. În final, sunt prezentate rezultatele obținute într-o aplicație de clasificare a rezultatelor scrise de mână.

Cuvinte cheie: recunoaştere sintactică a formelor, distanțe sintactice, clasificarea formelor plane.

1. Calcularea distanțelor între şiruri de primitive

Printre cei mai simpli algoritmi de clasificare a formelor ce pot fi descrişi prin intermediul şirurilor de primitive sunt cei bazați pe compararea (potrivirea) şirurilor atașate formelor. Problema este simplă în cazul în care structura formei necunoscute se potriveşte perfect cu una din structurile formelor lotului de învățare, însă, dacă acest lucru nu se realizează, atunci clasificarea trebuie făcută pe baza unei distanțe minime dintre şirul formei necunoscute şi şirurile de primitive corespunzătoare formelor lotului de învățare.

În general vorbind, a potrivi un şir de caractere A cu altul B, înseamnă să transformăm simbolurile cuvintelor A în literele lui B cu un cost minim al operațiilor posibile de realizat. În general, se presupune că există 3 tipuri de operații ce se pot efectua pentru a transforma un cuvânt în altul:

1. *Înlocuirea caracterelor*, ceea ce presupune schimbarea unui simbol a cu un altul b, operație ce se poate nota $a \to b$;
2. *Inserarea caracterelor*, introducerea unui simbol a în şir, echivalentă cu operația înlocuire: $\Lambda \to a$ în care Λ este simbolul nul;
3. *Eliminarea caracterelor*, adică ştergerea unei litere "b" din şir, operație echivalentă cu înlocuirea simbolului "b" cu cel vid: $b \to \Lambda$.

Fiecare dintre opera]iile de modificare a simbolurilor unui şir presupune că este cotată cu un anumit cost:

- $C(a,b)$ pentru o înlocuire $a \to b$;
- $C_1(a) = C(\Lambda, a)$ pentru o inserare $\Lambda \to a$;
- $C_2(b) = C(b, \Lambda)$ pentru o eliminare $b \to \Lambda$.

Vom admite că, pentru orice trei simboluri a,b,d din alfabetul V sunt adevărate:

a) C (a,b) > 0;

b) C (a,b) = 0, dacă şi numai dacă a=b;

c) C (a,b) = C (b,a);

d) C (a,d) < C (a,b) + C (b,d).

Dacă pentru transformarea unui şir de caractere $A = a_1 a_2 a_3 .... a_n$ în alt şir $B = b_1 b_2 b_3 ... b_m$, este necesară succesiunea de operații $s_1, s_2, ..., s_r$ în care $S = \{ s_i \mid i = 1, r \}$ sunt transformări de tipul celor definite anterior, atunci costul modificării lui A în B prin secvența de transformări $S = [s_1, s_2, ..., s_r]$ este:

$$C_t(S) = \sum_{i=1}^{r} C(S_i)$$

în care $C(S_i)$ este costul transformării $S_i = a_i \to b_j$, cu a_i sau b_j eventual egale cu cuvântul vid Λ.

Se defineşte distanța d (A,B) între şirurile A şi B ca minimul costurilor tuturor secvențelor de transformări care se pot face pentru a-l schimba pe A în B:

$D(A,B) = \min\{C_t(S) | S = s_1,\ldots,s_r \quad este\ o\ \text{sec}\,vențe\ de\ transformare\ a\ lui\ A\ în\ B\}$.

Dacă $A = a_1\,a_2\,a_3\ldots a_n$ şi $B = b_1\,b_2\,b_3\ldots b_m$, atunci se definesc *transformările speciale* ale lui A în B, secvenţele de modificări de forma:

$a_1 \to b_1, \quad a_2 \to \Lambda, \quad a_3 \to b_2,\ldots,a_n \to b_{m-1}, \quad \Lambda \to b_m,$

care sunt reprezentate în mod sugestiv în diagrama din figura 1, în care barele indică substituţiile,

$$\begin{array}{cccccccc} a_1 & a_2 & a_3 & a_4 & \ldots & a_{n-1} & a_n \\ | & . & . & & & . & \\ b_1 & b_2 & b_3 & & \cdots & b_{m-1} & b_m \end{array}$$

Figura 1. Secvenţa de operaţii necesare transformării cuvântului A în B

iar literele izolate sunt inserate sau eliminate. Într-o transformare specială, folosind reprezentarea schematică din figura 1, barele de substituire nu trebuie să se intersecteze şi niciodată două sau mai multe bare nu pornesc şi nu ajung la acelaşi caracter.

Se poate demonstra [18] că distanţa d (A,B) dintre două şiruri A şi B este egală cu minimul costurilor transformărilor speciale între A şi B. Această afirmaţie este de un real folos, deoarece numărul transformărilor oarecare între componentele celor două şiruri A şi B este mult mai mare faţă de numărul total al transformărilor speciale între A şi B, astfel că se restrâng destul de mult posibilităţile de căutare a costului minim în scopul determinării distanţei d (A,B).

Fie $A = a_1\,a_2\,a_3\ldots a_n$ şi $B = b_1\,b_2\,b_3\ldots b_m$ două cuvinte din dicţionarul V*. Vom nota prefixele acestor cuvinte cu:

$$A(i) = a_1\,a_2\,a_3\ldots a_i$$
$$B(j) = b_1\,b_2\,b_3\ldots b_j$$

Principiul de bază al algoritmului de calcul a distanţei între două cuvinte, care va fi prezentat în continuare, se bazează pe evaluarea distanţei între prefixele A (i) şi B (j) pe baza distanţelor între prefixele cu lungime mai mică cu o unitate, astfel:

$$d(A(i),B(j)) = D(i,j) = \min\begin{cases} D(i-1,j-1) + C(a_i,b_j), \\ D(i-1,j) + C_1(a_i), \\ D(i,j-1) + C_2(b_j) \end{cases}$$

1.1. Algoritmul Wagner – Fischer [20]

P1. $D(0,0) = 0$

P2. Pentru i de la 1 la n se calculează: $D(i,0) = D(i-1,0) + C_1(a_i)$

P3. Pentru j de la 1 la m se calculează: $D(0,j) = D(0,j-1) + C_2(b_j)$

P4. Pentru i = 1, n şi j = 1, m se determină:

$$d_1 = D(i-1,j-1) + C(a_i,b_j)$$
$$d_2 = D(i-1,j) + C_1(a_i)$$
$$d_3 = D(i,j-1) + C_2(b_j)$$
$$D(i,j) = \min\{d_1,d_2,d_3\}$$

P5. Distanţa între şirurile A şi B este $d(A,B) = D(n,m)$.

Pentru a arăta posibilitatea de aplicare a algoritmului Wagner – Fischer în clasificarea formelor vom apela la următoarea aplicaţie de recunoaştere a contururilor. Vom presupune

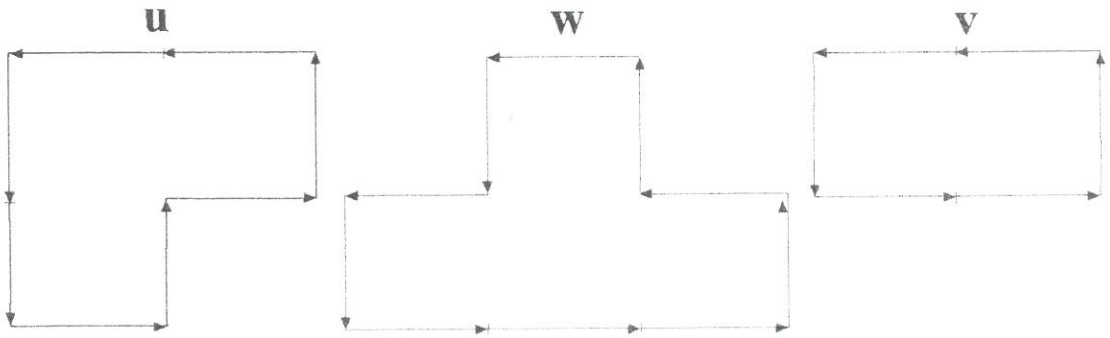


Figura 2. Reprezentarea sintactică a contururilor unor forme plane

ca formele u şi w prezentate în figura 2 sunt prototipurile a două clase ω_1 şi ω_2, iar forma v din aceeaşi figură trebuie clasificată într-una din cele două clase. Pentru a lua decizia de clasificare pe baza distanţei între şiruri de simboluri, se face descompunerea celor trei forme în raport cu primitivele prezentate în figura 3, împreună cu simbolurile ataşate: a, b, c, d.

Figura 3. Primitivele în raport cu care se descompun contururile din figura 2

Astfel, se obţin următoarele şiruri de primitive ataşate celor trei forme studiate:

u = a a b b c d c d

w = a b a b c c c d a d

v = a a b c c d

În raport de diferenţa dintre direcţiile celor patru primitive, vom atribui următoarele costuri pentru operaţiile de schimbare a simbolurilor:

$$C(e \to \Lambda) = 1$$
$$C(\Lambda \to e) = 1$$
$$C(e \to e) = 0$$

pentru orice $e \in \{a,b,c,d\}$ şi:

$C(a \to b) = 1$ $C(a \to c) = 2$ $C(a \to d) = 1$

$C(b \to c) = 1$ $C(b \to d) = 2$ $C(b \to a) = 1$

$C(c \to d) = 1$ $C(c \to a) = 2$ $C(c \to b) = 1$

$C(d \to a) = 1$ $C(d \to b) = 2$ $C(b \to c) = 1$

Rezultatul final al aplicării algoritmului Wagner – Fischer pentru calcularea distanţei dintre cele trei forme luate două câte două este prezentat în tabelul 1.

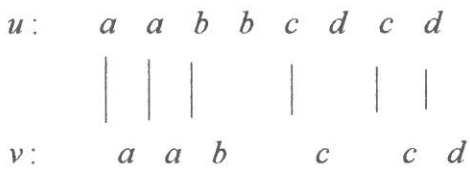
Tabelul 1. Distanţele Wagner – Fischer între contururile din figura 1

d_{W-F}	u	w	v
u	0	5	2
w	5	0	4
v	2	4	0

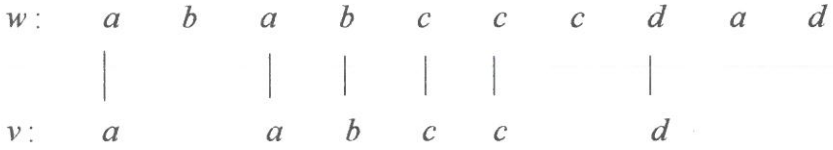
Se observă că distanţa minimă se obţine între u şi v şi, prin urmare, se ia decizia ca v să aparţină clasei din care face parte prototipul u.

De asemenea, este interesant de arătat care sunt transformările speciale, care conduc la un cost minim. Astfel:

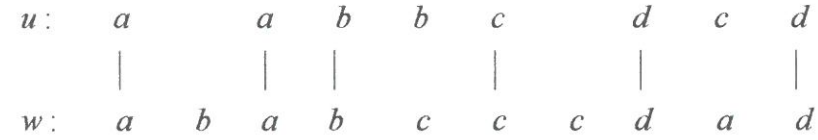
- pentru transformarea $u \rightarrow v$ avem:



- pentru modificarea $w \rightarrow v$:



- între prototipurile celor 2 clase, $u \rightarrow w$:



În reprezentările de mai sus ale transformărilor speciale, caracterele izolate sunt inserate sau eliminate, liniile subţiri reprezintă potriviri primitive, iar liniile îngroşate reprezintă înlocuiri de caractere.

1.2. Algoritmul Wagner – Fischer normat

Distanţa dintre 2 cuvinte este puternic influenţată de lungimea respectivelor şiruri de caractere. Pentru a proba această afirmaţie, trebuie să observăm că nu este echivalentă o anumită distanţă între cuvinte de lungime mică cu aceeaşi distanţă între cuvinte de lungime mare. Pentru a explica mai bine, să presupunem o distanţă egală cu 2, care, pentru cuvinte de 3 caractere, conduce la o eroare de 67% între cuvinte, iar pentru cuvinte de 10 litere procentul de nepotrivire a simbolurilor din cele două cuvinte scade la 20%.

Din acest motiv, se propune o metodă de normare a distanţei Wagner – Fischer între cuvinte, astfel încât aceasta să devină independentă de lungimea cuvintelor.

Fie u şi v două şiruri de caractere de lungimi n_u şi respectiv n_v.

Notăm $d_{W-F}(u,v)$ distanţa dintre cuvintele u şi v, calculată cu algoritmul Wagner – Fischer. Distanţa normată se calculează cu formula:

$$d_{normată}(u,v) = \frac{d_{W-F}(u,v)}{\max\{n_u, n_v\}}$$

Această distanţă are valoarea maximă egală cu 1 când toate simbolurile celor două cuvinte sunt diferite, indiferent de lungimea lor.

Valorile normate ale distanţelor între formele u, w şi v din figura 1, folosite anterior, sunt prezentate în tabelul 3.

Tabelul 3. Distanţele normate între contururile din figura 1

$d_{normată}$	u	w	v
u	0	0.5	0.25
w	0.5	0	0.4
v	0.25	0.4	0

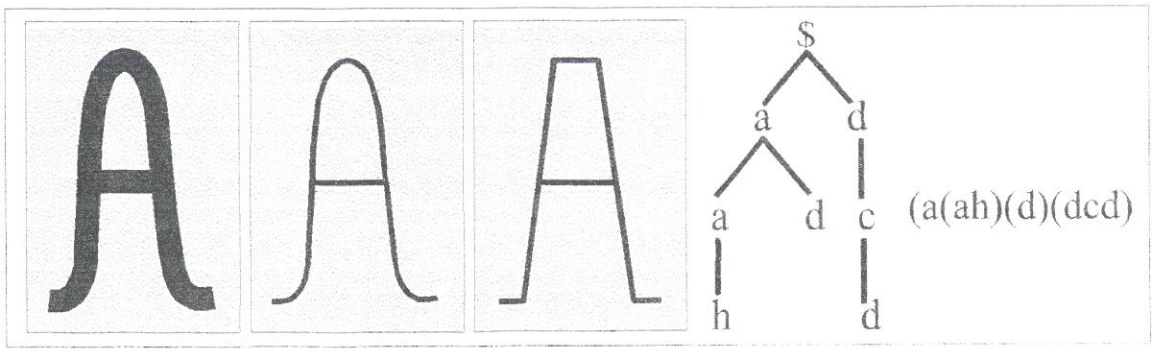
2. Aplicaţie de recunoaştere a caracterelor alfa – numerice, scrise de mână

Pentru o comparare mai bună a celor două metode, s-a realizat o aplicaţie de recunoaştere a caracterelor alfa–numerice, scrise de mână, descrisă în [21].

Astfel, s-au folosit caractere din 3 clase: A, C, E. Descrierea sintactică s-a realizat mai întâi sub formă de arbore. Pentru aceasta, s-a realizat, mai întâi, o prelucrare de subţiere după care aceasta a fost aproximată poligonal, folosind un anumit prag de eroare. Fiecare segmant al aproximării poligonale a fost codat ca primitivă în funcţie de domeniul său unghiular, după cum urmează:

- Primitiva *a* pentru $\alpha \in [-7\pi/8, -5\pi/8)$
- Primitiva *b* pentru $\alpha \in [-5\pi/8, -3\pi/8)$
- Primitiva *c* pentru $\alpha \in [-3\pi/8, -\pi/8)$
- Primitiva *d* pentru $\alpha \in [-\pi/8, \pi/8)$
- Primitiva *e* pentru $\alpha \in [\pi/8, 3\pi/8)$
- Primitiva *f* pentru $\alpha \in [3\pi/8, 5\pi/8)$
- Primitiva *g* pentru $\alpha \in [5\pi/8, 7\pi/8)$
- Primitiva *h* pentru $\alpha \in [7\pi/8, 9\pi/8)$

Reprezentarea de tip arbore este transformată în reprezentare sub forma de şir de caractere. Aşa cum s-a arătat în [21], metoda de transformare arbore-şir, care conduce la rezultatele cele mai bune este cea a delimitatorilor.



forma iniţială | forma subţiată | aproximare poligonală | structura iniţială de tip arbore | structura finală de tip şir

Figura 4. Lanţul de procesare pentru transformarea caracterelor de intrare în structura echivalentă de tip şir de primitive

Întreg lanţul de procesări necesare transformării caracterului de intrare în structura sa echivalentă de tip şir de primitive poate fi urmărit în exemplul dat în figura 4.

Rezultatele clasificării sunt prezentate în tabelul 4. Pentru comparare, s-au utilizat 4 parametri care pun în evidenţă capacitatea metodei de a mări sensibilitatea între clase, precum şi calitatea discriminării formelor aflate în zona de graniţă dintre clase.

Tabel 4. Rezultatul clasificării caracterelor

Parametrii comparaţi	Distanţa Wagner - Fischer	Distanţa normată
Distanţa medie intraclasă	2.85	0.21
Distanţa medie interclase	6.6	0.5
$\frac{\text{distanţa medie interclase}}{\text{distanţa medie intraclasă}}$	2.31	2.38
$\frac{\text{distanţa maximă intraclasă}}{\text{distanţa minimă interclase}}$	0.66	0.61

3. Concluzii

Se observă că, pentru metoda normată, se obţine o valoare mai mare a raportului (distanţa medie interclase/distanţa medie interclase), ceea ce presupune o mărire a sensibilităţii între clase, respectiv o concentrare a formelor în jurul mediilor şi o mărire a distanţei între clase. De asemenea, metoda normată este mai bună şi din punct de vedere al raportului (distanţa maximă intraclsae/distanţa minimă interclase), ceea ce presupune ca algoritmul normat realizează şi o clasificare mai bună a formelor aflate în vecinătatea graniţelor claselor.

Bibliografie

1. **ANDREWS, H.C.**: Introduction to Mathematical Techniques in Pattern Recognition, Wiley, New York, 1972.
2. **ATANASIU, I., A. MATEESCU**: Limbaje formale, Bucureşti, 1990.
3. **BUNKE, H.**: Attributed Programmed Graph Grammars and their Application to Schematic Diagram Interpretation to Texture Analysis. În: IEEE Trans. Pattern Analysis Mach. Intell., Vol. PAMI – 4, No. 6, November, 1982, pp. 1103-1114.
4. **ESHERA, M. A., K.S. FU**: A Graph Distance Measure for Image Analysis. În: IEEE Trans. Syst. Man, Cybernetics, Vol. SMC – 13, No. 3, March, 1994, pp.398-408.
5. **EVANS, T.G.**: Grammatical Inference Techniques in Pattern Analysis. În: Software Engineering, Vol. 2, J.T.Tou (Ed.), Academic Press, New York, 1971.
6. **Filipski, A. J.**: a LEAST Mean – Squared Error Approach to Syntactic Classification. În: IEEE Trans. Pattern Mach. Intell., Vol. PAMI – 2No. 3, March 1980, pp. 252-255.
7. **GRIGORE, O.**: Syntactical Clustering Using the Guided Random Search Method, special issue of Buletinul Ştiinţific al Univeritătii "Politehnica" Timişoara and Tran. Automatic Control and Computer Science and Engineering, Nov. 1996, pp. 174-182.
8. **GRIGORE, O.**: Syntactical Clustering Using Genetic Algorithms. În: Real World Applications on Intelligence Technologies, Editura Academiei, 1996.
9. **GRIGORE, O.**: Syntactical Self – Organization Map. În: Computational Intelligence, Theory and Applications, B. Reusch (ED.), Springer Verlag, Dortmund, 1997, pp. 101-109.
10. **GRIGORE, O.**: The Cumulative Inference Algorithm for Regular Grammars. În: Proc. of the 10^{th} Scandinavian Conference on Image Analysis, June 1997, Finland, pp. 191-199.
11. **GONZALES, R. C., P. WINTZ**: Digital Image Processing, Addison – Wesley, MA, 1996.
12. **LAM, L., S. W. LEE, S. Y. SUEN**: Thinning Methodologies – A Comprehensive Survez. În: IEEE Trans. Pattern Anal. Mach. Intell, Vol. PAMI – 14, No. 9, September 1992, pp. 869-885.
13. **LEMONE, K. A.**: Similarity Measures Between Extended to Sets of Strings. În: IEEE Trans. Pattern Anal. Mach. Intell, Vol. PAMI – 4, No. 3, May 1982, pp. 345-347.
14. **LU, S. Y., K. S. FU**: Error – Correcting Tree Automata for Syntactic Pattern Recognition and Image Processing, RPI; Troy, NY 1997, pp. 115-127.
15. **LU, S. Y., K. S. FU**: A Sentence – to –Sentence Clustering Procedure for Pattern Analysis. În: IEEE Trans. Syst. Man Cybern., Vol. SMC –8, No. 5, May 1994, pp. 219-227.
16. **Y. LU**: A Tree – to – Tree Distance and Its Applications to Cluster Analysis. În: IEEE Trans. Pattern Anal. Mach. Intell., Vol. PAMI – 1, No. 2, April 1979.
17. **NEAGOE, V. E., O. STĂNĂŞILĂ**: Teoria recunoaşterii formelor, Editura Academiei Române, Bucharest 1992.
18. **PAVLIDIS, T.**: Structural Pattern Recognition, Springer Verla, Berlin, 1977.
19. **WAGNER, R. A., M. J. FISCHER**: The String to String Correction Problem. În: Journal Ass. Comput. Mach., Vol. 21, No. 1, January 1974, pp. 42-51.
20. **GRIGORE, OV., O. GRIGORE**: Metode de clasificare sintactică a formelor plane descrise prin structuri de tip arbore. În: Revista Română de Informatică şi Automatică, vol. 10, nr. 2, 2000, pp. 23-32.