Főnévi csoportok tanulása és felismerése

Hócza András, Iván Szabolcs

Szegedi Tudományegyetem, Informatika Tanszék
6720 Szeged, Árpád tér 2.
hocza@inf.u-szeged.hu, ajven@programozo.hu
http://www.inf.u-szeged.hu

Kulcsszavak: főnévi csoportok felismerése, szabály alapú módszerek

Kivonat. A dolgozat azt tanulmányozza, hogy főnévi szerkezetek felismerése milyen részproblémákra bontható, illetve, hogy az egyes részproblémákban, milyen elemzések, teszteredmények segítenek bennünket a továbblépésben a lehető legjobb minőségű megoldás felé. A számos megközelítési lehetőség közül mi a szabály alapú módszereket választottuk, de ez is felvet számos specifikus részproblémát. Két tanuló algoritmust alkalmaztunk szabályok előállítására. Az egyik a közismert *C4.5*, a másik egy saját fejlesztésű algoritmus, az *RGLearn*. A teszteket egy erre a célra kifejlesztett NP elemzővel végeztük.
A kísérleteket és a különféle teszteket jelentős mértékben segítette a körülbelül 1,2 millió szót tartalmazó, kézzel annotált Szeged Korpusz [1], amely különböző (iskolai, szépirodalomi, számítógépes, jogi, üzleti) szövegtípusokra tartalmazza a nyelvészeti szakértők által bejelölt főnévi csoportokat.
Az NP felismerésre kifejlesztett elemzőnk, szakértői szabályokkal 65%-os, környezetfüggetlen szabályokkal 85%-os, környezetfüggő szabályokkal 90%-os pontossággal építette fel tesztállományban található NP szerkezeteket.

1 Bevezetés

A főnévi csoportok (továbbiakban: NP – Noun Phrase) tanulása rendkívül összetett probléma. Amikor egy ilyen problémára gépi tanulási technika segítségével szeretnénk megoldást találni, számos alternatíva nyílik meg előttünk. A választási lehetőségekre hozott döntések jelentős mértékben befolyásolhatják végeredmény minőségét.

Az NP felismerés az automatikus információ kinyerést [2], [3], [4], [5] megvalósító programlánc (ToolChain) részmodulja. Ez azt jelenti, hogy a felismerés minősége függ a megelőző modulok minőségétől is. Más szempontból viszont ez lehetőséget ad olyan megoldásokra, hogy bizonytalan esetekben döntést a folyamat következő moduljaira hárítsuk.

1. Ábra: Az NP felismerés helye az információ kinyerés folyamatában.

Munkánkat jelentős mértékben támogatta az elkészült álló Szeged Korpusz [1] amely különböző (iskolai, szépirodalomi, számítógépes, jogi, üzleti) szövegtípusokra tartalmazza a nyelvész szakértők által bejelölt főnévi csoportokat. A gépi tanuláshoz szükséges tréning és teszt adatokat a korpuszból vettük. A tesztelés során a kiértéke-

lést, a manuális bejelölés minőségi ellenőrzésénél is jól bevált saját fejlesztésű (NpCheck) algoritmussal végeztük, amely összehasonlította az előállított és etalon XML fájlokat, hogy milyen mértékben térnek el a bennük tárolt NP szerkezek.

A következő fejezetekben az általunk alkalmazott szabály alapú módszer kialakításának részleteit ismertetjük.

2 Az NP felismeréshez alkalmazott módszer kiválasztása

Számos megközelítési lehetőség van az NP felismerés kapcsán. Igen népszerűek és hatékonyak a különféle valószínűségi modelleket alkalmazó módszerek, mint például a HMM [5]. Ezeknél a módszereknél a döntést támogató modell egy valószínűségi értékeket tartalmazó számhalmazzal van reprezentálva, melyek emberi szem számára nem értelmezhetők, módosítani, belőlük következtetéseket levonni nem lehet. A szabály alapú módszereknek viszont számos előnyük van:

- Könnyen áttekinthetők, értelmezhetők.
- Könnyen kiegészíthetők szakértői tudás beépítésével, amelyek megnyilvánulhatnak szakértők által adott szabályokban, kezdeti hipotézisben, vagy a meglévő szabályok finomításában
- A támogatás gyors és egyszerűen megvalósítható.
- A szövegben rejlő mintázatok jók kereshetők és reprezentálhatók.

A szabály alapú módszerek is számos alternatívát vetettek fel. Alapvetően a megvalósítandó rendszer döntéstámogatáshoz használhat szakértői szabályokat, vagy valamilyen gépi tanulási technika által előállított szabályokat.

2.1 Szakértői szabályok

A Szeged Korpusz létrehozásának a NP bejelölési fázisában ki lett dolgozva egy útmutató a manuális munkát elvégző nyelvész szakértők számára, annak érdekében, hogy a körülbelül 15-20 fő által elvégzendő munka egységes irányelvek szerint legyen megvalósítva. Azonban ez az útmutató számos helyen tartalmazott intuitív elemeket, mely információ a számítógép nyelvére nehezen fordítható le. Történtek kísérletek nyelvészeti szakértők által megadott szabályrendszer kialakítására, de a fentebb említett problémák miatt a szabályrendszerbe végül csak a legbiztosabb (95%-100% találati pontosságú) szabályok kerültek be. Ez azt jelentette, hogy ha az elemző döntött valamilyen szabály alapján, az nagy valószínűséggel jó volt, de nagyon sok NP-t nem jelölt be. A szakértői szabályok végül a manuális munka felgyorsításában játszottak szerepet, a CLaRK[1] rendszerrel végzett előfeldolgozás segítségével. A szakértői szabályok felhasználásával az NP elemző átlagosan 65%-os pontosságot ért el.

2.2 Szabály alapú gépi tanulási technikák alkalmazása

A Szeged Korpusz feldolgozásának különféle szakaszban alkalmaztunk ILP (*Inductive Logic Programming*) módszereket különféle szabályrendszerek előállításánál. Az IMPUT algoritmus [6] a szabályok specializálásával képes megjavítani egy

[1] Programfejlesztő: Kiril Simov, BulTreeBank Project, Linguistic Modelling Laboratory, CLPP, Bulgarian Academy of Sciences (http://www.bultreebank.org).

szabályrendszer pontosságát, azonban kellett ehhez szükség volt egy szabályrendszert előállító módszerre is. Másrészt a Prológ rendszer lehetőségeit nehéz lett volna átvinni egy önállóan működni képes programba, márpedig a automatikus információ kinyerés részeként nekünk ilyen modulra volt szükségünk. Így a C4.5 [7] döntési fa készítő algoritmus és egy saját fejlesztésű módszer az RGLearn (*Rule Generalization Learner*) mellett döntöttünk végül.

3 A tanulási példák előállítása

A tanulási fázis megkezdése előtt az XML fájlokban tárolt információkat át kell alakítani gépi tanulásra alkalmas formába, azaz a rendelkezésre álló információkat úgy kell átszervezni, hogy az tanulási probléma legyen. Szabály alapú tanulás esetén ez egy táblázattal reprezentálható, melynek a sorai a ***tanulási példák***, az oszlopai az ***attribútumok*** és a táblázat egy kitüntetett oszlopa a ***döntés***. A tanulás lényegében a táblázatnak a döntés szerint ellentmondásmentes tömörítéséből (általánosításából) áll.

2.2 Főnévi szerkezetek előfeldolgozása

A tanulás minőségét több előfeldolgozással kapcsolatos tényező is befolyásolhatja:

- **Az attribútumok száma:** Ha túl sok az attribútum, az jelentős mértékben lelassíthatja a tanuló futását.
- **Az attribútumok információ tartalma:** Ha nincs elég információ, a tanuló nem tud elfogadható pontosságú szabályokat előállítani.
- **A példák száma:** Minél több a példa, annál jobb az eredmény.
- **Redundancia:** Ha egy adott típusból sok hasonló példa van, más típusból viszont kevés az ronthatja az eredményt.
- **Konzisztencia:** Ha a példák sok hibát, ellentmondást tartalmaznak az szintén rontja az eredményt.

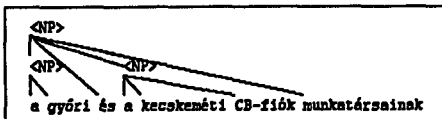
Ezért az előfeldolgozásnak jelentős szerepe van abban, hogy milyen minőségű lesz a gépi tanulással előállított szabályrendszer, tehát ez a lépés nem egyszerűen csak konvertálás, hanem valóban a tanulási probléma megszerkesztése.

2.2 Főnévi szerkezetek előfeldolgozása

A főnévi szerkezetek többnyire fa struktúrát alkotnak, ez azt jelenti, hogy egy NP magasabb szinten összekapcsolódhat egy vagy több másik NP-vel vagy szóval, együtt egy újabb NP-t alkotva. A gépi tanulás megvalósításához a fa struktúrát le kell bontani elemi fa építő utasításokká. A tanulási példák szókörnyezet, döntés párokból állnak, a tanulási probléma az lesz, hogy egy adott szó pozíciója kezdő eleme-e egy NP-nek

- **Szókörnyezet:** szavak és MSD kódok a vizsgált szó pozíciójából kiindulva. A következő attribútumok vannak: ..., CL2, CL1, C, CR1, CR2,...,WL2, WL1, W, WR1, WR2,... (pl: CL3 attribútum jelentése: a vizsgált szótól balra 3-mal lévő szó MSD kódja.)
- **Döntés:** Lehetséges értékei: NONE, NP1, NP2, NP3,... (pl: NP3 szimbólum jelentése az, hogy a vizsgált szó kezdő szava egy 3 szó hosszú NP-nek).

A főnévi szerkezetek lebontása több menetben történik, addig tart amíg van feldolgozható NP. Mindig a legbelső NP kerül először lebontásra.




```
<NP>
     <NP>
          a [Tf]
          győri [Afp-sn]
     </NP>
     és [Ccsw]
     <NP>
          a [Tf]
          kecskeméti [Afp-sn]
          CB-fiók [Nc-sn]
     </NP>
     munkatársainak [Nc-pg---s3]
</NP>
```

Példák kiírása:
[az 'a' szó környezete], NP2
[a 'győri' szó környezete], NONE
[az 'és' szó környezete], NONE
[az 'a' szó környezete], NP3
[a 'kecskeméti' szó környezete], NONE
[a 'CB-fiók' szó környezete], NONE
[a 'munkatársainak' szó körny.], NONE

A legbelső NP-k helyettesítése:

```
<NP>
     NP
     és [Ccsw]
     NP
     munkatársainak [Nc-pg---s3]
</NP>
```

2. Ábra: Egy főnévi szerkezet lebontásának és a példák kiíratásának menete.

3 A tanuló algoritmus

A C4.5 algoritmus az ID3-algoritmus egy továbbfejlesztett változata. J. R. Quinlan[2] nevéhez fűződik. A C4.5 egy döntési fát állít elő, melyben a csomópontok egy-egy attribútumra vonatkozó kérdések, a levelek pedig a döntések. A C4.5 úgy próbálja előállítani a döntési fát, hogy minél kevesebb kérdéssel el lehessen jutni a döntéshez, ezért azokat az attribútumokat választja ki csomópontoknak, melyeknek legnagyobb az információs nyeresége. A döntési fa pedig átkonvertálható szabályokká.

Azonban szükség volt egy olyan algoritmusra, amely megengedi, hogy az NP felismerés specifikus részproblémáit is implementálni lehessen. Ezt a C4.5 nem támogatta. Ezért fejlesztettük ki az RGLearn algoritmust, mely egy kezdeti szabályrendszerből kiindulva úgy próbálja azt általánosítani, hogy az általánosítással bejövő hiba egy előre megadott küszöbérték alatt maradjon. A kezdeti szabályrendszer lehet szakértők által megadott szabályok, vagy a tréning példákból generált, nem alapértelmezett döntést tartalmazó esetek (NP tanulás esetén ezek a NP1, NP2, ... döntés esetei).

```
RULE_SET = non default cases from EXAMPLE_SET
while change RULE_SET do
{
     foreach RULE of RULE_SET do unification RULE
     foreach RULE of RULE_SET do generalization RULE
     foreach RULE of RULE_SET do delete rules covered by RULE
}
```

Unification RULE:
Összevonja RULE szabályt azzal a szabállyal ami hozzá legjobban hasonlít, ha az így kapott szabály pontossága nagyobb egy előre megadott küszöbértéknél.

Generalization RULE:
Általánosítja a RULE szabályt úgy, hogy egy attribútumot értékét általánosabb reguláris kifejezésre cseréli, vagy elhagyja, ha az így kapott szabály pontossága nagyobb egy előre megadott küszöbértéknél.

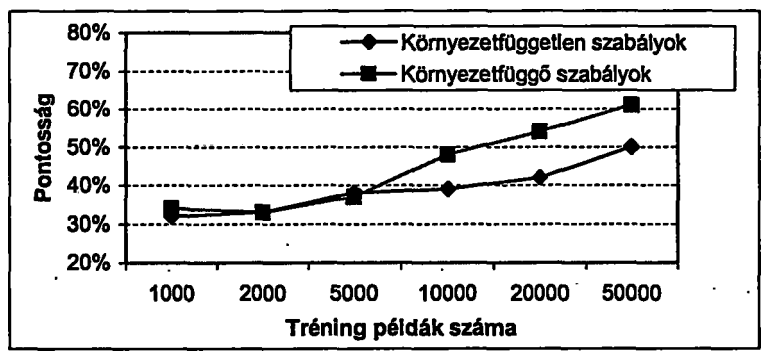
[2] J. R. Quinlan: C 4.5: Programs for Machine Learning, Morgan Kaufmann Publishing, (1993).

4 NP felismerés

Az NP felismerésnél az előfeldolgozás fordítottja, az NP szerkezetek felépítése történik a legbelső NP-ből kiindulva. Akkor jó az NP elemző, ha a tréning példákon nagy pontossággal reprodukálni tudja a megtanult szerkezeteket és ismeretlen teszt szövegen is jó hatásfokkal, az etalonnal egyezőnek ismer fel. A felismerés módja azonban számos kérdést felvet.

4.1 Környezetfüggő vagy környezetfüggetlen szabályok?

Az alábbi grafikonon található mérési eredmények arról informálnak bennünket, hogy az egyforma anyagon, gépi tanulással előállított környezetfüggő szabályok a tréning példák növelésével dominánsan jobbak, mint a környezetfüggetlen szabályok. Ez a különbség a többletinformációból adódik. A teljes korpuszon végzett gépi tanulással kapott szabályrendszer környezetfüggetlen esetben átlagosan 85%, környezetfüggő esetben átlagosan 90%-os eredményt ért el a tesztállományokon.



3. Ábra: A környezetfüggő és környezetfüggetlen szabályokkal végzett NP felismerés eredményei különböző számú tréning példán

4.2 Mohó algoritmus vagy a döntés elhalasztása?

Az NP elemző által használt szabályrendszer lehetővé teszi a szabályok közötti gyors (bináris) keresést. Az először a legjobb szabályokat (melyek kevés hibával sok esetet lefednek) próbálja meg alkalmazni. A végrehajtás két lépésben történik: először van a lehetséges szabályalkalmazások bejelölése a teljes mondatra. A második lépésben történik a bejelölések végrehajtása. Vitás esetben, ha a bejelölt NP-k átlógnak egymásba, a legjobb szabály által bejelölt NP mellett dönt az elemző. A végrehajtás addig megy, amíg van bejelölhető NP. A NP elemző algoritmusa a következő:

```
while change do
{
    foreach tag of sentence do
    foreach rule of ruleset do
    if rule covers tag then sign tag with rule

    foreach tag of sentence do
    if tag sign with better rule then subtitute NP
}
```

A fentebb ismertetett (*mohó*) algoritmusnak az az előnye, hogy egyből előállít egy nagy valószínűséggel jó NP struktúrát egy adott mondatra. Azonban van számos hátránya is, például nem biztos, hogy a legjobbat. Az például jobb megoldás lenne, ha előállítaná az összes lehetséges szerkezetet, úgy, hogy valamilyen heurisztikát használva minden lehetőséghez egy valószínűségi értéket rendelne, a következő szempontok szerint:

- Az alkalmazott szabályok összesített valószínűsége.
- Az NP szerkezetek komplexitása (szintek, levelek száma) minél nagyobb.
- Csak olyan megoldások fogadhatók el melyek lefedik az összes főnevet.
- A legjobb megoldások közül egy későbbi fázis (ontológia, szemantikus keretek illesztése) plusz információi alapján választunk.

Ez utóbbi módszer még fejlesztés alatt van, de a Morphologic Kft által kifejlesztett HumorEsk program felhasználásával lesz megvalósítva.

5 Összefoglalás és fejlesztési lehetőségek

A dolgozatban bemutattuk egy szabály alapú NP tanuló és elemző rendszer előállításának lépéseit. Vázoltuk azt, hogy egyes lépésekben milyen lehetséges megoldások közül választhattunk és milyen megfontolások és előzetes teszteredmények alapján választottuk ki az általunk ítélt legjobbat.

A közeljövőben szeretnék még több módszert kipróbálni, összehasonlítani, hogy még hatékonyabb megoldást kapjunk az információ kinyerésnek erre az igen fontos részproblémájára. Az egyik lehetőség, hogy megvalósítva az összes NP szerkezet előállítását komplexebb problémaként próbálunk választani, esetleg bevonva későbbi fázisokat is, mint ontológia, szemantikus keretek illesztése. A másik lehetséges irány, hogy a szabály alapú módszerek mellett más, például HMM módszert vagy esetleg több módszer kombinációját alkalmazzuk a problémára.

Irodalom

1. Alexin, Z., Csirik, J., Gyimóthy, T., Bibok, K., Hatvani, Cs., Prószéky, G., Tihanyi, L. (2003) *Manually Annotated Hungarian Corpus*, in Proceedings of the Research Note Sessions of the 10th Conference of the European Chapter of the Association for Computational Linguistics EACL03, Budapest, Hungary, pp. 53–56.
2. Hócza, A., Alexin, Z., Csendes, D., Csirik, J., Gyimóthy, T.: *Application of ILP methods in different natural language processing phases for information extraction from Hungarian texts* in Proc. of the Kalmár Workshop on Logic and Computer Science, Szeged, Hungary, 1-2 October, pp. 107-116 (2003)
3. Freitag D. (2000) *Machine Learning for Information Extraction in Informal Domains*, Machine Learning, 39, 169–202.
4. Califf, M. E. and Mooney, R. J. (2001) *Bottom-Up Relational Learning of Pattern Matching Rules for Information Extraction*, in Journal of Machine Learning Research
5. Freitag and McCallum (2000) *Information extraction with HMM structures learned by stochastic optimization*, in Proceedings of the Seventeenth National Conference on Artificial Intelligence, 2000.
6. Alexin, Z., Gyimóthy, T., and Boström, H. (1997) *IMPUT: An Interactive Learning Tool based on Program Specialization*, Intelligent Data Analysis (IDA) Journal, Vol 1 No 4, Elsevier Holland.
7. Quinlan, J. R. (1993) *C 4.5: Programs for Machine Learning*, Morgan Kaufmann Publisher.