Teljes mondatszintaxis tanulása és felismerése

Hócza András

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

Kivonat: A dolgozat teljes szintaxis tanulására mutat be egy szabály alapú módszert, amely több mélységű faminták általánosításával oldja meg a problémát. A módszer algoritmusa a korábban hasonló problémákra alkalmazott *RGLearn* egy erre acélra továbbfejlesztett változata. Bemutatásra kerül továbbá egy részfa-kiválasztó módszer ami a részfák alakja alapján működik. Segítségével egy teljes szintaxisfa szintenként lebontható kisebb részfákká. A mondatelemző a tanult szabályrendszer alapján képes felépíteni egy ismeretlen mondat szintaxisát. A megvalósításhoz szükséges információkat a Szeged Korpusz adattárából vettük

Kulcsszavak: teljes szintaxis, gépi tanulás, szabály alapú módszerek

1 Bevezetés

Egy mondat teljes szintaxisának felismerése egy olyan folyamat, amely során meg kell határozni, hogy milyen egymás után következő szavak csoportosíthatók egybe, mint például főnévi, melléknévi, igei, ... szerkezetek. További feladat lehet a mondat egymástól távolabb eső részeinek összekapcsolása, azaz a vonzatkeretek meghatározása. Ezek az információk nélkülözhetetlenek a gépi eszközökkel történő mondatértés megvalósításánál. Ez azt jelenti, hogy be kell azonosítani a mondat minden egyes szavát, szócsoportját és mivel ezek egymásra épülnek, fel kell tárni a mondatszintaxis faszerkezetét. További jellemzője még a teljes szintaxisnak, hogy a végeredményül kapott szintaxis-fa összefüggő, teljesen lefedi a mondatot és a gyökere hagyományosan egy S szimbólum.

Az elkészült mondatszintaxis számos természetes nyelvi feladathoz szolgálhat nélkülözhetetlen információkkal. Ilyen feladat lehet például az adatbányászat, információkinyerés, gépi fordítás. Ezekből a lehetőségekből az üzleti hírekből történő információkinyerés megvalósítása vált számunkra a fő célkitűzéssé. Az általunk kifejlesztett automatikus információkinyerést (Hócza et al., 2003) megvalósító programlánc (ToolChain) moduljai folyamatos továbbfejlesztés alatt állnak. Ezek a modulok különféle természetes nyelvi feladatokat oldanak meg, mint például mondat- és szószegmentálás, morfológiai elemzés, szófaji egyértelműsítés, szintaxis felismerés, ontológiai elemzés és szemantikus keretek illesztése.

A különféle gépi tanulással megvalósított természetes nyelvi feladatokhoz szükséges információk a Szeged Korpusz (Alexin et al., 2003) adattárából származtak. Az 1.2 millió szavas korpusz különböző (iskolai, szépirodalomi, számítógépes, jogi, üzleti) szövegtípusokra tartalmazza a nyelvész szakértők által megvalósított szófaji egyértelműsítést és teljes szintaxis elemzést. A teljes szintaxis előállítása két lépésben történt, a munka első fázisában a főnévi szerkezetek (NP) bejelölése történt meg, a teljes szintaxis bejelölésének a munkálatai nemrég fejeződtek be.

A dolgozat a következő módon épül fel: a 2. rész általánosan mutatja be a mondatszintaxis kutatását, a 3. részben a tanulási példák előállításáról lesz szó, a 4. rész az alkalmazott gépi tanulási módszert írja le, az 5. rész a szabályok segítségével történő szintaxis felismerésről szól és végül a 6. rész összefoglalja az elért eredményeket.

2 A mondatszintaxis tanulása és felismerése

A magyar nyelv számos olyan elemet tartalmaz, ami megnehezíti a szintaxis felismerést más indoeurópai nyelvekhez képest. Az egyik nagy probléma a viszonylag szabad szórend, azaz hogy egy adott mondat szavai, mondatrészei sokféle sorrendben átrendezhetők anélkül, hogy megváltozna annak értelme. Ennek a ténye jelentősen megnöveli a lehetséges minták, nyelvi sémák számát, ami rontja az ezen alapuló gépi tanulás hatékonyságát. Ezt a hatást még fokozza számos egyéb nyelvi sajátosság, mint például az, hogy a főnévi csoportokból hiányozhat a névelő és akár maga a főnév is melyek jól felismerhető határelemei lehetnének ezeknek a csoportoknak. Mindezeket a magyar nyelv ragozással, képzők alkalmazásával oldja meg, ami viszont bonyolultabbá teszi a morfológiai elemzést és a szófaji egyértelműsítést, melynek hibái továbbgyűrűzhetnek az ezeken alapuló mondatszintaxis felismeréséhez.

A mondatszintaxis felismerésének létezik egy olyan megközelítése is, amely csak részleges elemzést (*Shallow Parsing*) végez el. Ez a módszer, előállítva a mondatszintaxis leglényegesebb elemét, a főnévi csoportokat, igen hasznos információt nyújt az erre épülő feladatoknak, például az információkinyerésnek.

Az eredményeket az összehasonlíthatóság érdekében közös mérőszámokkal kell jellemezni. Erre a gyakorlatban a következő 3 érték szolgál:

- Pontosság: a helyesen felismert szócsoportok száma / az összes felismert szócsoportok száma.
- Fedés: a helyesen felismert szócsoportok száma / a teszt mintában ténylegesen szereplő szócsoportok száma.
- Középarány ($F_{\beta=1}$): 2* Pontosság* Fedés/(Pontosság+ Fedés)

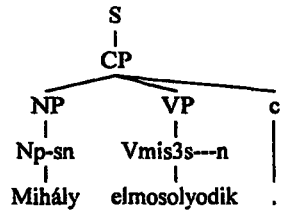
Angol nyelvre számos eredmény létezik a mondatszintaxis felismerésének témakörében. Az első publikációban (Abney, 1991) nyelvtani kódok alapján osztályozta a szavakat, hogy azok kezdő, vég vagy belső elemei-e egy adott típusú frázisnak. A Penn Treebank (Marcus et al., 1993) annotált szövegeiben elkülönítettek egy részt, amely a megjelenése óta összehasonlítási alapot képez a témához kapcsolódó publikációk eredményeihez. (Ramshaw and Marcus, 1995) transzformáción alapuló tanulást valósított meg ($F_{\beta=1}$=92.0). (Argamon, 1998) egyszerre végezte főnévi és igei szerkezeteket felismerését ($F_{\beta=1}$=91.6). (Tjong Kim Sang and Veenstra, 1999) bevezette a több fokozatban (kaszkád) alkalmazott felismerést ($F_{\beta=1}$=92.37). A legújabb módszerek úgy érnek el javulást az eredményekben, hogy több módszert is összekombinálva,

szavazással hozzák meg a döntéseket, (Tjong Kim Sang, 2000) öt különféle módszert kombinált össze ($F_{\beta=1}$=93.26).

Magyar nyelvre idáig nem készült igazán jó minőségű szintaxiselemző program. Az igazi probléma nem is a felismerő program megvalósításában, hanem az azt működtető szabályrendszer előállításában van. Az előzőekben vázolt nehézségek miatt szinte lehetetlen olyan nyelvész szakértők által kézzel készített szabályrendszert megalkotni, ami megfelelő hatékonyságú és minden lehetséges esetre kiterjed. A másik probléma, hogy idáig nem volt elegendő mennyiségű annotált magyar szöveget tartalmazó korpusz, ami a gépi módszerek alkalmazását lehetővé tette volna. A MorphoLogic által kifejlesztett HumorESK mondatelemző (Kis, 2003) 1995 óta folyamatosan fejlődik. Ez idő alatt különféle nyelvészeti területeken alkalmazták, mint például tulajdonnév-, főnévi csoport- és igevonzat-felismerés. Fő jellemzője, hogy a szimbólumokhoz jegyszerkezeteket (*feature structure*) kapcsol és elemzési erdőt épít az egyes jegyek öröklötetésével. Az elemzőben használt nyelvtan nyelvész szakértők közreműködésével állt elő. A Nyelvtudományi Intézet készülő szintaktikai elemzője (Váradi, 2003) főnévi szerkezeteket ismer fel, amely (Abney, 1998) ötletén alapulva reguláris kifejezésekkel leírt, több fokozatú (kaszkád) szabályrendszert alkalmaz. A szabályrendszert nyelvész szakértők készítették a CLaRK rendszer (Simov, 2001) segítségével és az elemzések futtatása is ezzel történt. Az elemző tesztjét egy kisebb annotált szövegen végezték ($F_{\beta=1}$=58.78). A Szegedi Egyetemen a nyelvtan előállítása gépi tanulási módszerekkel történik, egy ilyen nyelvtanon alapuló elemző (Hócza, 2004) szintén főnévi szerkezetek felismerésére készült. Az ebben alkalmazott reguláris kifejezésekkel leírt, környezetfüggetlen, valószínűségi nyelvtan tréningjének forrását, a Szeged Korpusz annotált szövegei adták. A kiértékelés a korpusz teszt célra elkülönített szövegein készült (általános szövegek: $F_{\beta=1}$=78.59, üzleti hírek: $F_{\beta=1}$=83.11).

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. A mondatban szereplő szócsoportok egymásba ágyazottak, ami azt jelenti, hogy ezek a szerkezetek fa-struktúrát alkotnak. Ezért a gépi tanulás megvalósításához a fa-struktúrát le kell bontani, olyan mintákat elkülönítve benne, melyeket alkalmazni lehet egy tetszőleges (ismeretlen) mondat fa-struktúrájának az előállítására. Egy szócsoport megadása az elemek nyitó és záró címkék közötti felsorolásából áll. A szavakhoz tárolt információ MSD (Morpho-Syntactic Description) kódból (Erjavec and Monachini, 1997) és a szótőből áll. Az annotált szövegek a következő csoportokat tartalmazhatják: főnévi csoport (NP), melléknévi csoport (ADJP), határozószói csoport (ADVP), igei csoport (VP), főnévi igenév (INF), tagadószó (NEG), igekötő (PREVERB), kötőszó (C), névmás (PP), tagmondat (CP), mondat(S).



A mondat leírása:
<S> <CP> <NP> Np-sn|Mihály </NP> <VP> Vmis3s---n|elmosolyodik </VP> cl. </CP> </S>

Kinyerhető minták:
<NP> Np-sn|Mihály </NP>
<VP> Vmis3s---n|elmosolyodik </VP>
<CP> NP VP cl. </CP>
<S> CP </S>

1. Ábra: Egy rövid példa mondat teljes szintaxisra és az ebből kinyerhető mintákra.

Az 1. ábrán egy lehetséges példa látható minták kinyerésére, mely több lépésben áll elő, úgy hogy mindig csak azok a csoportok kerülnek kiírásra, melyben a nyitó és záró címkék között csak terminálisok fordulnak elő. Ennek az a hátránya, hogy mivel a minták környezetfüggetlenek elvesznek a minta környezetében szereplő információk, melyek meghatározzák, hogy milyen körülmények között alkalmazható a minta ismeretlen szövegen. Például a 2. ábra mintáiból kinyerhető az **NP → Np-sn|Mihály** szabályról azt lehet elmondani, hogy a Mihály szót megelőzheti névelő és melléknév (a szőke Mihály), tehát ilyen esetben hibát okoz az alkalmazása. Ennek a problémának a mérséklésére a mintagyűjtő algoritmus több egymásba ágyazott szócsoportot is kigyűjt, azaz nem csupán terminális-sorozatok, hanem részfák lesznek eltárolva. Azonban a másik véglet sem túl jó, a túl nagy részfák tárolása, mert ezek többnyire annyira egyediek, hogy nem fordulnak elő még egyszer a tréning vagy a teszt szövegben. A középutas megoldást két faalak típus alkalmazása adja:

1. "Gödör": Legalább 2 terminálist tartalmaz, egy monoton mélyülő és egy monoton emelkedő szakaszra osztható, csak egy irányváltás van benne. Ez egy rekurzív szerkezet, ami többnyire úgy keletkezik, hogy a belső fához, hozzácsapódik 1-2 terminális egy-egy újabb fa-szintet alkotva, például:

```
<NP>
    <NP>
        Tilegy
        <ADJP>
            Afp-sn|rettenetes
        </ADJP>
        Nc-sn|örvény
    </NP>
    Nc-sp---s3|szélén
</NP>
```

2. "Füzér": Több irányváltás is lehet benne, de a lebontatlan részfák max. 1 mélységűek és 1 hosszúak. Ezek vagy felsorolás jellegű szerkezetek, vagy a már lebontott részfák összefűzései egy szerkezetbe, például:

```
<NP>
    <NP>
        Np-snlMihály
    </NP>
    <C>
        Ccswlés
    </C>
    <NP>
        Np-snlErzsi
    </NP>
</NP>
```

Ezek a faalak típusok elég gyakran előfordulnak az annotált szövegekben, több kisebb részfát magukba foglalnak, nem túl hosszúak és nem is túl rövidek, valamint belátható, hogy segítségükkel tetszőleges szintaxis-fa lebontható részfákra.

4 Minták tanulása

A tanulási fázis az előző fejezetben vázolt előfeldolgozással előállított minták általánosítását végzi el. Az általánosítás azt jelenti, hogy a minta szavaihoz tartozó információk közül elhagyjuk a szótőt vagy az MSD kód egyes betűit. (MSD kód minden betűje valamilyen morfológiai információt jelöl.) Az általánosítással az adott minta több esetre alkalmazható is lesz és ez a kulcsa annak, hogy az előállított minták ismeretlen szövegre is alkalmazhatók legyenek. A túlzott általánosításnak viszont az a hátránya, hogy a minta több olyan esetet is lefedhet, melyet nem kellene, azaz megnőhet a hibás felismerések száma. Ennek elkerülésére a tanuló algoritmus optimalizálja az általánosítást, hogy a lehető legkisebb hiba mellett legyen megtanult minta halmaz a legáltalánosabb. Ez úgy valósul meg, hogy minden új minta előállításakor a tanuló algoritmus hiba-statisztikát készít a tréning mondatokon.

A fentebb vázlatosan leírt módszer egy többszörösen átdolgozott saját fejlesztésű algoritmus, az RGLearn (Hócza et al., 2003). Ennek az algoritmusnak egy változata volt alkalmazva a főnévi szerkezetek felismerésére (Hócza, 2004), mely most újabb továbbfejlesztésen esett át a teljes szintaxis felismerése kapcsán. Az algoritmus vázlatos működése következő:

```
amíg van feldolgozatlan minta addig {
    (1) vesz egy új mintát
    (2) előállítja a minta a legáltalánosabb alakját (a
          szavak MSD kódjának csak az első betűjét, azaz
          a szófajt hagyja meg)
    (3) előállítja a pozitív (helyes) fedéseket
    legyen v=0 amíg van javulás addig v=v+1 {
          (4) előállítja az összes lehetséges szabályt, ami
                legfeljebb v számú attribútumot hoz be
                a pozitív fedések adataiból
          (5) hibastatisztika készítése a szabályokra
          (6) a legjobb szabályokkal lefedi a pozitív
                fedéseket
    }
    (7) a legjobb fedésben résztvevő szabályokat eltárolja
}
```

Megjegyzések az egyes pontokkal kapcsolatban:

(1) A minták az előfeldolgozásból származnak, a 3. fejezetben leírtaknak megfelelően a tréning mondatok szintaxis fáinak lebontásával. A minták lebontási szintenként vannak csoportosítva, azaz, hogy hányadik lebontási menetben sikerült kinyerni az adott mintát. Ez az információ azért fontos, mert a tesztmondaton is ilyen sorrendben kell majd alkalmazni a mintákból kapott szabályokat.

(2) A minta legáltalánosabb alakja a nyitó és záró címkékből (pl.: <NP>, </NP>), valamint a szavak MSD kódjainak első betűjéből (azaz a szófajból) áll, szótövek nélkül.

(3) A pozitív fedések halmaza úgy áll elő, hogy a tréning mondatokból kigyűjtjük azokat a részsorozatokat, melyekre az általánosított minta hibátlanul ráilleszthető.

(4) Egy adott szabály úgy áll elő, mint az általános minta specializációja, hogy veszünk egy sorozatot a pozitív fedések halmazából és bizonyos számú attribútumot (betűt az MSD kódból, vagy a szótövet) behozunk belőle, kiegészítve vele a legáltalánosabb mintát. Ez a lépést az összes lehetséges módon megteszszük. A behozott attribútumok száma 0-ról indul és addig növeljük amíg javulást tapasztalunk a (6)-os pont elvégzése után, azaz a hibás fedések száma csökken.

(5) Minden szabályt kiértékelünk megvizsgálva azt, hogy hány esetben lehet helyesen vagy esetleg hibásan ráilleszteni a tréning mondatokra, ezekből az adatokból egy $F_{\beta=1}$ értéket képezünk.

(6) Kiválasztjuk a szabályoknak azt a részhalmazát, ami a legkevesebb hibával fedi le a (3) pontban kigyűjtött jó esetek halmazát teljes mértékben.

(7) Amikor már nincs javulás, vesszük a legjobb (6)-os pontban kigyűjtött szabályrészhalmazt és ennek szabályait eltároljuk.

5 Szintaxis felismerés

A szintaxis felismerésnél az előfeldolgozás fordítottja, a szerkezetek felépítése történik. Akkor jó az 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 tanuló algoritmus által előállított szabályrendszer tesztelése az alábbiakban ismertetett algoritmussal történt, melynek előnye az, hogy egyértelműen előállít egy nagy valószínűséggel jó struktúrát egy adott mondatra. Azonban van hátránya is, például nem biztos, hogy a legjobbat, mivel nem az összes lehetséges fa közül választ, hanem minden választási ponton döntést hoz arról, hogyan menjen tovább. Egy adott mondat teljes szintaxisának előállítása a következő algoritmussal történt:

```
amíg van alkalmazható szabály addig {
     az osszes terminálisra {
     az osszes szabályra {
     ha a szabály illeszthető a terminális poziciótól akkor {
            (1) a terminális megjelölése a szabállyal
     }}}
     az osszes megjelölt terminálisra {
     ha nem része más megjelölt terminálisnak akkor {
            (2) a behelyettesítés elvégzése
     }}}
}
```

Megjegyzések az egyes pontokkal kapcsolatban:

(1) A terminális megjelölése azt jelenti, hogy hozzárendeljük a terminálishoz a szabály behelyettesítéshez szükséges adatokat, mint pl. a szabály azonosítója, pontértéke ($F_{\beta=1}$) az illeszkedő sorozat utolsó terminálisa. Ha a terminális már megjelölt, ha az új szabály pontértéke jobb, akkor lecseréljük erre.

(2) A behelyettesítés azt jelenti, hogy a sorozatot egyetlen terminális szimbólummal (az illeszkedő részfa gyökere) helyettesítjük. Egy terminális akkor része egy másik megjelölésnek, ha a szóban forgó terminális sorozat eleje és vége között van.

A tesztelés tréningje úgy történt, hogy a Szeged Korpusz adattárából véletlenszerűen kiválasztott 1000 mondatra lett lefuttatva a 4. fejezetben ismertetett tanuló algoritmus. Az így kialakult szabályrendszer tesztje pedig újabb 100 véletlenszerűen kiválasztott mondaton történt. A teszt eredményét az alábbi táblázat tartalmazza:

Tényleges szócsopotok száma	1835
Felismert szócsopotok száma	1846
Helyesen felismert szócsopotok száma	1557
Pontosság	84,34%
Fedés	84,85%
Középarány ($F_{\beta=1}$)	84,59%

2. Ábra: Teszteredmények 100 mondatra.

A kialakult szabályok pontértéke alapján a legrosszabbak a tagmondatok felismerésére tanult szabályok. A minták a tagmondatok szintjén túlságosan egyediek, kevésnek tűnik a rendelkezésre álló információ, ami arra utal, hogy az elemzést előállító nyelvész szakértők valószínűleg szemantikai információkat is figyelembe vettek a munkájuk során.

6 Összefoglalás és fejlesztési lehetőségek

A dolgozatban bemutatásra került egy teljes szintaxis felismerésére alkalmazott szabály alapú tanuló módszer. Az előállított szabályrendszer segítségével egy elemző algoritmus azonnali döntések meghozatalával a legvalószínűbbnek ítélt szintaxis fát építi fel egy tetszőleges ismeretlen mondaton. A teszteredmények biztatóak, egy nagyobb anyagon végzett tréning során kialakuló szabályrendszer alkalmas lehet az információkinyerést támogató teljes szintaxis felismerést végző modul megvalósításához.

A közeljövőben szeretnénk kifejleszteni egy olyan elemzőt, amely képes az összes lehetséges elemzés előállítására és ezek közül a legjobb kiválasztására. Tervezzük ontológiai információk figyelembevételét is az elemzések során.

Irodalom

Abney S. (1991) Parsing by chunks, in *Principle-Based Parsing*. Kluwer Academic Publishers.

Abney S. (1996) Partial Parsing via Finite-State Cascades, in *Proceedings of ESSLLI'96 Robust Parsing Workshop*, pp. 1-8.

Argamon, S., Dagan, I., and Krymolowski, Y. (1998) A memory-based approach to learning shallow natural language patterns, in *Proceedings of 36th Annual Meeting of the Association for Computational Linguistics (ACL)*, Montreal, pp. 67-73.

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.

Erjavec, T. and Monachini, M., ed. (1997) Specification and Notation for Lexicon Encoding, *Copernicus project 106 "MULTEXT-EAST"*, Work Package WP1 – Task 1.1 Deliverable D1.1F.

Hócza, A., Alexin, Z., Csendes, D., Csirik, J., Gyimóthy, T. (2003) Application of ILP methods in different natural language processing phases for information extraction from Hungarian texts, in *Proceedings of the Kalmár Workshop on Logic and Computer Science*, Szeged, Hungary, 1-2 October, pp. 107-116.

Hócza (2004) Noun Phrase Recognition with Tree Patterns, in *Proceedings of the Acta Cybernetica*, Szeged, Hungary

Kis, B., Naszódy, M., Prószéki, G. (2003) Komplex (magyar) szintaktikai elemző rendszer mint beágyazott rendszer, *MSZNY 2003 konferencia kiadványa*, Szeged, 145-151 oldal.

Kuba, A., Bakota, T., Hócza, A., Oravecz, Cs. (2003) A magyar nyelv néhány szófaji elemzőjének összevetése, *MSZNY 2003 konferencia kiadványa*, Szeged, 16-23 oldal.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1993) Building a large annotated corpus of English: the Penn Treebank, Association for Computational Linguistics.

Muggleton, S. and Feng, C. (1992) Efficient Induction of Logic Programs, in Inductive Logic Programming (ed.: S. Muggleton), *Academic Press*, New York, pp. 281–297.

Plotkin, G.D (1970) A note on inductive generalization, *Machine Intelligence* (eds: B. Meltzer and D. Michie), Vol 5.

Ramshaw, L. A., and Marcus, M. P. (1995) Text Chunking Using Transformational-Based Learning, in *Proceedings of the Third ACL Workshop on Very Large Corpora*, Association for Computational Linguistics.

Simov K. (2001) CLaRK – an XML-based System for Corpora Development, in *Proceedings of the Corpus Linguistics 2001 Conference*, Lancaster, pp. 553-560.

Tjong Kim Sang, E. F., and Veenstra, J. (1999) Representing text chunks, in *Proceedings of EACL '99*, Association for Computational Linguistics.

Tjong Kim Sang, E. F. (2000) Noun Phrase Recognition by System Combination, in *Proceedings of the first conference on North American chapter of the Association for Computational Linguistics*, Seattle, pp. 50-55.

Váradi, T. (2002) The Hungarian National Corpus, in *Proceedings of the Second International Conference on Language Resources and Evaluation LREC2002*, Las Palmas de Gran Canaria, pp. 385–389.

Váradi T. (2003) Shallow Parsing of Hungarian Business News, in *Proceedings of the Corpus Linguistics 2003 Conference*, Lancaster, pp. 845-851.