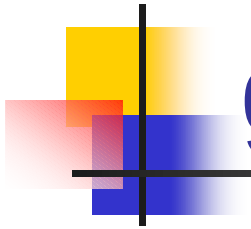
# Statistisk grammatikgranskning

Johnny Bigert
**johnny@nada.kth.se**

# Traditionell grammatikgranskning



Hitta stavningsfel och grammatiska fel:

- Regler
- Lexikon

# Traditionell grammatikgranskning

Fördelar:

- Säkert och beprövat
- Bra resultat på många feltyper

Exempel:

- Den lilla huset är gult.

| Den | lilla | huset | är | gult. |
|---|---|---|---|---|
| det.utr | adj.utr/neu | subst.neu | vb | adj.utr |

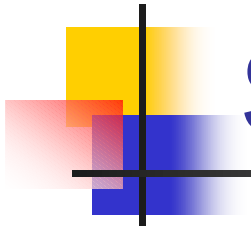
# Traditionell grammatikgranskning

Problem:

- Regler innebär mycket manuellt arbete
- Vissa feltyper är svåra att beskriva med regler och lexikon

Exempel på svåra fel:

- Steve förstod, i motsatts till sina konkurrenter, att om…
- Att Compaq han före IBM med att lansera…
- Några år senare skev han sin första bok…



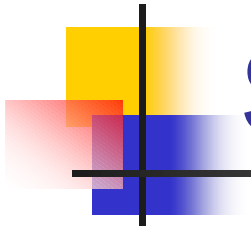
# Statistiska metoder

Statistiska metoder:

- All information hämtas från korpus
- Mindre eller inget manuellt arbete
- Ofta jämförbar prestanda med regelbaserade metoder

Exempel: disambiguering av ordklasser

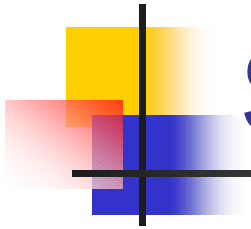
- Jag såg en en en dag.



# Språknorm

Statistisk information finns t.ex. i normalspråkskorpusar:

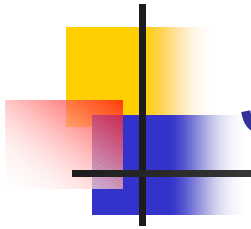
- Stora mängder text
- Innehåller implicit given information om språknormen



# Språknorm

Observationer:

- Grammatiska fel bryter mot språknormen
- Grammatiska konstruktioner som inte observerats i korpus är förmodligen felaktiga



# Jämför språknormen

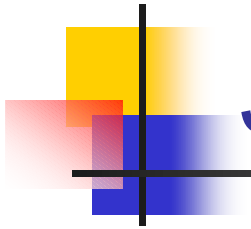
Alltså:

- Jämför texten med den språknorm som ges av korpus!

# Jämför språknormen

Grammatikgranskning:

- Okända konstruktioner är förmodligen ogrammatiska
- (”Förmodligen” eftersom korpus alltid är för liten)



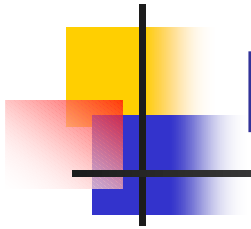
# Jämför språknormen

Problem:

- Hur kan man representera normen som ges av korpus?

Kompromiss:

- Titta endast på lokal information



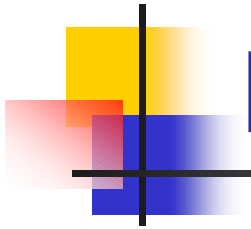
# Lokal information

Lokal information: *n*-gram

- Se indatat som en ström av element
- *n* på varandra följande element bildar ett *n*-gram

Exempel:

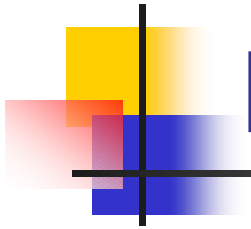
- Ord (tokens): Den svarta katten satt på det lilla bordet.
- Taggar: **(dt.utr.sin.def) (jj.pos.utr/neu.sin.def.nom) (nn.utr.sin.def.nom) (vb.prt.akt) (pp) (dt.neu.sin.def) (jj.pos.utr/neu.sin.def.nom) (nn.neu.sin.def.nom) (mad)**



# Lokal information

Informationssamling:

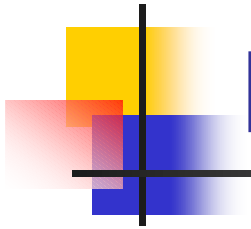
- Man vill behålla så mycket information som möjligt, dvs helst *n*-gram av ord
- Dock har man otillräcklig storlek på korpus för detta



# Lokal information

Informationssamling:

- Man använder *n*-gram av taggar
- Bibehåller ingen semantik (innebörd)
- Bibehåller syntax (struktur)



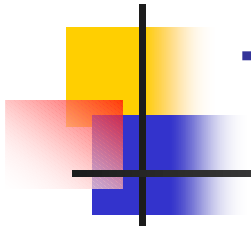
# Lokal information

Vår taggуppsättning:

- 149 taggar

Exempel:

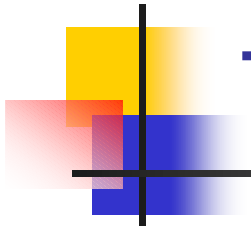
- Kattens **(nn.utr.sin.def.gen)**
- Äpplen **(nn.neu.plu.ind.nom)**
- 25 substantiv, 26 verb, 15 adjektiv, 4 adverb osv.



# Trigram av taggar

*n*-gram av taggar med befintligt korpus:

- För litet *n* ger för lite men säker information (*n* = 1, 2)
- För stort *n* ger för gles och därför osäker information (*n* >= 4)
- Vi använder *n* = 3, dvs **trigram** av taggar



# Trigram av taggar

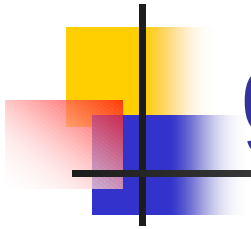
Befintliga korpusar:

- Stockholm-Umeå korpus (SUC): 1M ord och 95000 unika trigram (ger 2.9% av $149^3$)
- Parole: 22M ord och 261000 unika trigram (ger 7.9% av $149^3$)

Observation:

- Antalet unika trigram ökar med storleken på korpus

# En första statistisk grammatikgranskare



Algoritm:

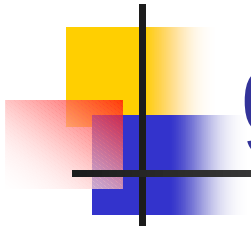
För varje position *i* i indataströmmen

Om frekvensen för ($t_{i-1}\ t_i\ t_{i+1}$) är låg

Rapportera fel till användaren

Rapportera inget fel

# En första statistisk grammatikgranskare

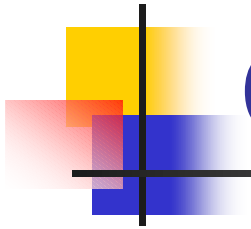


Tidigare observerat:

- Antalet unika trigram ökar med storleken på korpus

Konsekvens:

- Trigramfrekvensen på en grammatiskt korrekt konstruktion kan vara noll
- Korpus är aldrig tillräckligt stor
- Otillräckligt storlek ger glesa data



# Glesa data

Tidigare observerat:

- Vi behåller inte innebörd, bara syntax

Idé:

- Byt ut ovanliga taggar mot vanligare med liknande betydelse.

# Glesa data

Exempel:

- ”Det är varje chefs uppgift att…”
- ”Det är varje” är taggad **(pn.neu.sin.def.sub/obj, vb.prs.akt.kop, dt.utr/neu.sin.ind)** och har frekvens noll
- Möjlig orsak: av 1M ord i korpus har endast 709 fått taggen **(dt.utr/neu.sin.ind)**

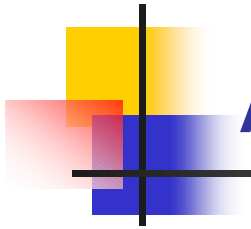
# Glesa data

Vi byter ut ovanliga konstruktioner:

- ”Det är varje chefs uppgift att…”

byts mot

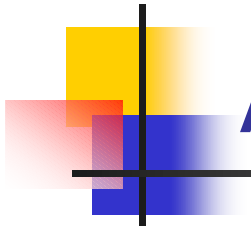
- ”Det är en chefs uppgift att…”
- ”Det är en” är taggad **(pn.neu.sin.def.sub/obj, vb.prs.akt.kop, dt.utr.sin.ind)** och har frekvens 231
- **(dt.utr.sin.ind)** har frekvens 19112, **(dt.utr/neu.sin.ind)** hade frekvens 709



# Avstånd mellan taggar

Vid byte av tagg:

- Alla taggar passar inte att ersätta alla andra
- Vissa taggar passar, men olika bra
- Vi önskar ett straff för byten



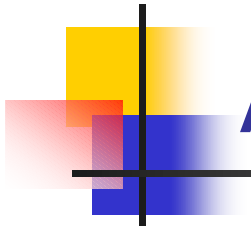
# Avstånd mellan taggar

Att ta hänsyn till:

- Manuellt arbete med att skapa byteslistan för varje tagguppsättning och språk
- Svårt att uppskatta avstånd

Förslag:

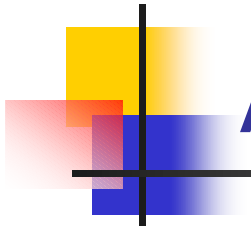
- Bygg statistisk information för avståndet mellan två taggar



# Avstånd mellan taggar

Översikt:

- Avståndsmatrisen är baserad på trigram av taggar
- Givet en kontext, byt ut taggen mot dess representant
- Skillnad i trigramfrekvens pga bytet



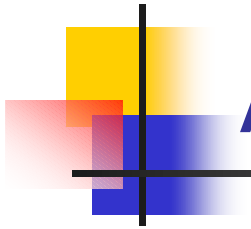
# Avstånd mellan taggar

Hjälpfunktion:

- u($c_v$ $t$ $c_h$) = freq($c_v$ $t$ $c_h$) / freq($t$ )
- Normerar trigramfrekvensen

Exempel:

- **(dt.utr.sin.ind)** har frekvens 19112
- **(dt.utr/neu.sin.ind)** hade frekvens 709
- Första determineraren borde ha trigramfrekvenser i snitt 19112/709=27.0 gånger högre än den andra
- Frekvensjämförelse utan normering orättvis



# Avstånd mellan taggar

Kontextberoende taggavstånd:

- Givet: vänsterkontext $c_v$ och högerkontext $c_h$
- Avstånd givet kontext:

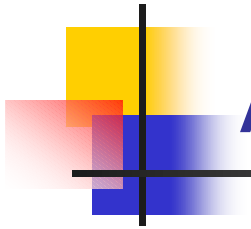
$$\text{dist}_{c_v c_h}(t_1\ t_2) = ?\ |u(c_v\ t_1\ c_h) - u(c_v\ t_2\ c_h)|$$

# Avstånd mellan taggar

Taggavstånd:

- Betrakta samtliga vänsterkontexter $c_v$ och högerkontexter $c_h$ (dvs $149^2$ st)
- Avstånd mellan två taggar:

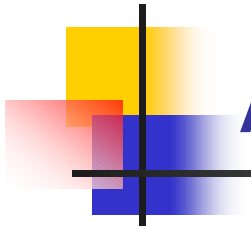
$$\text{dist}(t_1\ t_2) = ?\ \text{dist}_{c_v c_h}(t_1\ t_2)$$



# Avstånd mellan taggar

Utnyttja data maximalt:

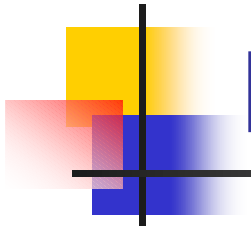
- Vi beräknade avstånd för ($c_v$ $t$ $c_h$)
- Vi beräknar även avstånd för ($t$ $c_1$ $c_2$) och ($c_1$ $c_2$ $t$) på samma sätt



# Avstånd mellan taggar

Utnyttja data maximalt:

- Vi får $3 \cdot 149^2 = 66000$ möjliga observationer
- Av dessa är i genomsnitt $3 \cdot 149^2 \cdot 0.029 = 1912$ nollskilda



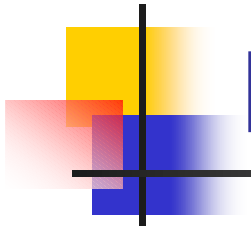
# Bytesmatris

Exempel ur bytesmatrisen (SUC):

| | | |
|---|---|---|
| **0** | **nn.neu.sin.def.nom** | **nn.neu.sin.def.nom** |
| **1.14** | **nn.neu.sin.def.nom** | **nn.utr.sin.def.nom** |
| **2.01** | **nn.neu.sin.def.nom** | **nn.utr.plu.def.nom** |
| **2.13** | **nn.neu.sin.def.nom** | **nn.neu.plu.def.nom** |

Mannen såg huset.

Mannen såg bilen.



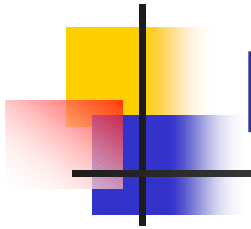
# Bytesmatris

Exempel ur bytesmatrisen:

```
0      vb.prt.akt.mod     vb.prt.akt.mod
1.96   vb.prt.akt.mod     vb.prs.akt.mod
3.22   vb.prt.akt.mod     vb.prt.akt____
3.28   vb.prt.akt.mod     vb.prs.akt____
```

Mannen var glad.

Mannen är glad.



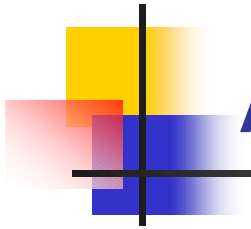
# Bytesmatris

Exempel ur bytesmatrisen:

| | | |
|---|---|---|
| 0 | dt.utr/neu.plu.def | dt.utr/neu.plu.def |
| 3.38 | dt.utr/neu.plu.def | dt.utr/neu.plu.ind/def |
| 3.47 | dt.utr/neu.plu.def | ps.utr/neu.plu.def |
| 3.57 | dt.utr/neu.plu.def | jj.pos.utr/neu.plu.ind.nom |

Mannen hälsar på de anställda.

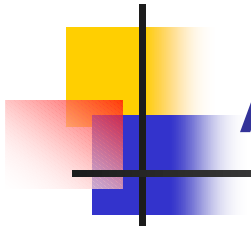
Mannen hälsar på våra anställda.



# Automatisering

Automatisering av bytet mellan taggar:

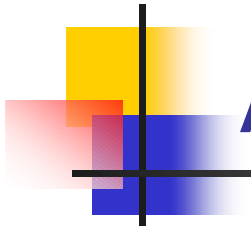
- Givet: en tagg *t* och en representant *r*
- Vi slår upp bytet i bytesmatrisen och finner straffet *d*
- Överför till ”sannolikhet” med funktion g(x) : R → [0,1], typiskt g(x) = 1/(1+x)



# Automatisering

Automatisering av bytet mellan taggar:

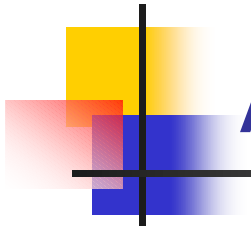
- Vi har ett ovanligt trigram ($t_1$ $t_2$ $t_3$)
- $t_1$ bytes bäst mot $r_{11}$ $r_{12}$ $r_{13}$ ... $r_{1m}$
- $t_2$ bytes bäst mot $r_{21}$ $r_{22}$ $r_{23}$ ... $r_{2m}$
- $t_3$ bytes bäst mot $r_{31}$ $r_{32}$ $r_{33}$ ... $r_{3m}$



# Automatisering

Byten och straff för individuella taggar:

- Antag att $t_1$ byts mot $r_{1i}$
- Bytet medför ett straff $d_{1i} = \text{dist}(t_1\, r_{1i})$
- Samma för $t_2$ och $t_3$ med representanter $r_{2j}$ och $r_{3k}$ (ger straff $d_{2j}$ resp $d_{3k}$)



# Automatisering

Byten av trigram:

- Antag att ($t_1$ $t_2$ $t_3$) byts mot ($r_{1i}$ $r_{2j}$ $r_{3k}$)
- Ny trigramfrekvens $f_{ijk}$
- Denna frekvens är för hög utan hänsyn till bytesstraff
- Inför straff: $f'_{ijk} = f_{ijk} \cdot g(d_{1i}) \cdot g(d_{2j}) \cdot g(d_{3k})$

# Automatisering

Undersök samtliga representantkombinationer:

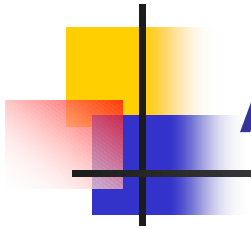
- Slå upp frekvensen för ($r_{11}$ $r_{21}$ $r_{31}$)
- Slå upp frekvensen för ($r_{12}$ $r_{21}$ $r_{31}$)
- Slå upp frekvensen för ($r_{13}$ $r_{21}$ $r_{31}$)

  ... ...
- Slå upp frekvensen för ($r_{1m}$ $r_{21}$ $r_{31}$)
- Slå upp frekvensen för ($r_{11}$ $r_{22}$ $r_{31}$)
- Slå upp frekvensen för ($r_{12}$ $r_{22}$ $r_{31}$)

  ... ...
- Slå upp frekvensen för ($r_{1m}$ $r_{22}$ $r_{31}$)

  ... ...
- Slå upp frekvensen för ($r_{1m}$ $r_{2m}$ $r_{3m}$)

Totalt $m^3$ st

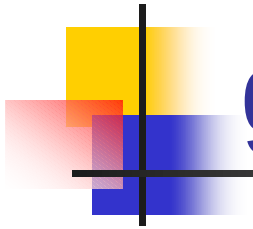


# Automatisering

Byten av trigram:

- Beräkna $f'_{ijk}$ för alla *i*, *j* och *k*
- Väg ihop de viktade frekvenserna
- Förslag: *sum* eller *max*

# Modifierad statistisk grammatikgranskare



Algoritm:

För varje position *i* i indataströmmen

Om viktade frekv för ($t_{i-1}\ t_i\ t_{i+1}$) är låg

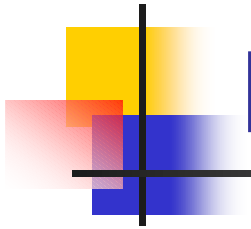
Rapportera fel till användaren

Rapportera inget fel

# Modifierad statistisk grammatikgranskare

Viktad frekvens (med *sum*):

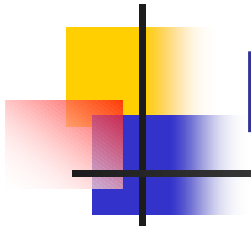
*sum* ← 0
För varje *i* ← 1,2,...,m
    För varje *j* ← 1,2,...,m
        För varje *k* ← 1,2,...,m
            $f \leftarrow \text{frekv}(r_{1i}\ r_{2j}\ r_{3k})$
            $p \leftarrow g(d_{1i}) \cdot g(d_{2j}) \cdot g(d_{3k})$
            $sum \leftarrow sum + f \cdot p$
Returnera *sum*



# Resultat

Kort om resultaten:

- Hittar ovanliga grammatiska konstruktioner
- Ofta konstruktioner som avviker från normen
- Prestanda kan förbättras väsentligt med hjälp av fraser och satsgränser (framtida föredrag)



# Referenser

Artikel:

- Bigert 2002, ”POS Tag Distance Metrics and Unsupervised Error Detection”
- Finns att hitta på:
  **`http://www.nada.kth.se/~johnny`**