Lösningsförslag för tentamen i Språkteknologi, DH2418
2010-10-21

1. Flertydighet (4 poäng)
Meningen är flertydig eftersom ordet ”bara” antingen kan vara ett adverb (”bara”=”enbart”) eller ett adjektiv (”bara män”=”oklädda män”), och eftersom ordet ”utan” antingen kan vara en subjunktion eller en preposition (”utan röda kläder”=”som saknade röda kläder”). Dessutom är det syntaktiskt flertydigt hur prepositionsfrasen ”på festen” ska kopplas till resten av meningen. Utöver den semantiskt rimliga tolkningen ”På festen såg hon …”, kan man även se ”röda kläder på festen” som en fras (dvs. festen har röda kläder på sig; jämför ”Hon såg röda kläder på skyltdockan”). Ytterligare tolkningar är säkert tänkbara.

2. Maskininlärning (4 poäng)
2 (a) Övervakad maskinlärning genererar en funktion som avbildar indata på utdata, genom att generalisera från en träningsmängd av indata-utdata-par. Oövervakad maskinlärning försöker hitta mönster i en datamängd. Ett exempel är klustring som identifierar grupper av liknande dataobjekt.

(b) Övervakad inlärning: En text med POS-taggar. Oövervakad inlärning: En samling dokument.

3. Informationsteori (4 poäng)
Perplexiteten är 2^entropin. För en ordprediktor motsvarar perplexiteten hur många ord den har att välja på. Ordprediktorns perplexitet ska minimeras.

4. Ordklasstaggning (4 poäng)
Det måste finnas regelmallar som kan generera regler av typen:

Tagg1 → Tagg2 i kontexten K.

Utgångsgissningen sätter taggen för ”sticka” till VB.

Vissa antaganden om den ordklasstaggade korpusen (träningsmängden) måste göras, och det är rimligt att det finns flera fall som säger att mönstret DT NN är vanligt.

En regel som genereras utifrån detta blir då:

VB → NN om föregående tagg är DT

Denna regel rättar fler fel än den orsakar och därför behålls den, vid jämförelse med facit.

När meningen ”Han fick en sticka i fingret” skall taggas får ”sticka” först taggen VB, men regel ovan ändrar till den korrekta taggen NN.

5. Morfologi (4 poäng)

Vid lemmatisering försöker man hitta grundformen för ett ord, och för detta krävs det morfologisk analys. Till exempel lemmatiseras ”hundarnas” såväl som ”hunden” till ”hund” och både ”sprang” och ”sprungit” lemmatiseras till ”springa”. Lemmatisering som skall välja rätt lemma utifrån kontexten kräver ordklasstaggning.

Vid stemming däremot försöker man med enkla omskrivningsregler föra in morfologiskt (och ibland även semantiskt) besläktade ord under en gemensam stam. Till exempel skulle ”cyklade”, ”cyklarnas” och ”cykeln” (via en regel som i detta fall även tar bort e:et) kunna stammas ner till den gemensamma stammen ”cykl”

Lemmatisering är mycket mer krävande än stemming och metoden för inte föra ihop semantiskt besläktade ord från olika ordklasser. Detta ger lägre täckning, men bättre precision.

Vid stemming finns det risk för övergenerering, t.ex. at banan (frukten) och banan (något som man kör på) får samma stam: ”ban” Detta sänker precisionen.

Del 2: Problemdel (30 poäng)

6. Stavningskontroll (6 poäng)
Gå igenom en stor korpus (till exempel hela BNC) och räkna hur många gånger varje bokstavsfyrgram förekommer. Ordbörjan/ordslut behandlas som en egen bokstav, så det blir 27 bokstäver i alfabetet. I den tredje ordningens markovmodellen har vi ett tillstånd för varje bokstavstrigram (som förekommer). Sannolikheten för en övergång från tillstånd abcd till tillstånd bcde är
C(bcde)/C(bcd)=C(bcde)/(sum C(bcdx)) där x löper över alla 27 bokstäver och C() är korpusfrekvensen.

Startsannolikhetsfördelningen ges av frekvenserna för bokstavstrigram som inleder ord dividerat med antalet ord i korpusen. För varje ord får vi då en totalsannolikhet som är en produkt av startsannolikheten och övergångssannolikheterna.

Vi behöver nu sätta en tröskel för vad ett ords totalsannolikhet ska vara för att det ska räknas som korrekt. Det blir en tröskel för varje ordlängd, och trösklarna ska sättas så att nästan inga riktiga ord hamnar under tröskeln. När man kontrollerar ett ord måste man alltså slå upp sannolikheterna, beräkna produkten, jämföra med rätt tröskel och underkänna ordet om produkten är mindre än tröskeln.

7. Ordprediktion (6 poäng)
Sannolikheten är noll för vissa trigram eftersom dessa inte ingår i träningsmängden. Detta är ett problem speciellt om ordprediktorn är tränad på en liten datamängd, eller en datamängd som inte är representativ för situationen i vilken applikationen kommer att användas. I dessa fall ökar sannolikheten att det ord man vill skriva härnäst inte föreslås av ordprediktorn. Detta kan undvikas genom smoothing, t.ex. att sannolikheten för trigrammet xyz estimeras med en viktad summa som även inkluderar sannolikheten för bigrammet yz och unigrammet z (se kap 4.5 i boken). Ytterligare en åtgärd man kan utföra är att ordprediktorn tränas om allt eftersom användaren skriver, så att applikationen anpassas efter användarens ordval och stil.

8. Informationssökning (6 poäng)
Normalisera dokumentvektorerna med avseende på tf, idf, och längd. Använd cosinus-mått för att mäta skillnad mellan Q och D, i stället för det föreslagna måttet (som kommer att gynna längre dokument, och gynna märkliga dokument som upprepar populära söktermer många gånger). Använd PageRank eller liknade för att ordna dokumenten i träfflistan.

9. Utvärdering och språkgranskning (6 poäng)

Detta problem handlar ju delvis om resurser. Men vi får anta att Eva har en rimlig budget. Eva bör anställa en språkteknolog och en lingvist för uppdraget. Hon bör också begära från leverantörerna att få en körbar version av programmet. Materialet bör delas in olika delar eftersom det kan vara stor skillnad mellan en religionsbok och en fysiskbok.

När väl detta är på plats bör språkteknologen testköra grammatikkontrollerna på en del av läroboksmaterialet för att få svar på frågan om det redan finns tillräckligt med fel i materialet. Om språkteknologen är klok så har hen byggt ett litet verktyg som matar fram de textavsnitt där programmen gör fel. Detta verktyg kan lingvisten använda för att skapa ett ”felkorpus” baserat på utdata från samtliga program. På detta sätt samlas samtliga kontrollerade larm med feletiketter från de olika grammatikkontrollerna i en ”pool”. Lite beroende på resurser och hur många fel som hittas automatiskt kan Eva låta lingvisten manuellt analysera en mindre del av materialet.

Språkteknologen bygger under tiden ett verktyg som automatiskt kan räkna ut felfrekvenser per feltyp, precision, täckning, och F-värde (F-measure) för det läroboksmaterial man är intresserad av. Hen utvecklar också ett verktyg som kan säga om det finns statistiskt signifikanta skillnader mellan två grammatikkontroller utifrån deras performans på felkorpusen.

Efter en första analys med framräknande av felfrekvenser, precision, och relativ täckning kan beslut tas om det behöver skapas ”nya” fel i materialet eller om det räcker med dem som redan finns. Om det finns för få fel, t.ex. för en viss feltyp behöver ett verktyg som skapar fel utvecklas. Med ett sådant verktyg kan felen annoteras automatiskt. Lingvisten får undersöka den skapade felkorpusen med stickprovskontroller att felen som skapas är vettiga. När felkorpusen är kontrollerad kan utvärderingsverktyget köras igen, och språkteknologen och lingvisten kan ge Eva en rapport om vilken grammatikkontroll som är bäst.

10. Dialogsystem (6 poäng)

Eftersom problemet är så väldefinierat verkar det enklast att systemet ställer en rad frågor om önskad film, biograf, osv., och användaren svarar. Denna dialoglogik skulle kunna åstadkommas med en ändlig automat. Efter varje fråga sker en taligenkänning med en grammatik som är specifik för den frågan (en grammatik som specar alla filmtitlar i fråga 1, alla biografer i fråga 2, osv.). Systemet kan utvärderas på flera sätt, till exempel andel kunder som lyckas göra en reservation (task completion), eller genomsnittligt antal feligenkänningar per dialog. En nackdel med den föreslagna lösningen är att vana användare inte kan ta genvägar, typ "Avatar på Rigoletto i kväll kl 19", eftersom endast en fråga i taget kan besvaras.