



UPPSALA
UNIVERSITET

Språkteknologiska metoder för stavningsövningar i ett CALL-system

*Magisteruppsats i Språkteknologi, 30hp
Institutionen för lingvistik och filologi
Uppsala Universitet*

Camilla Liljhammar, camlij@stp.lingfil.uu.se

Rebecka Sundström, rebsund@stp.lingfil.uu.se

Handledare: Joakim Nivre

Biträdande handledare: Jonas Jonsson

Uppsala University
Department of Linguistics and Philology
Språkteknologiprogrammet
(Language Technology Programme)
Master's thesis in Computational Linguistics

October 11, 2010

Sammandrag

I denna uppsats kommer vi att undersöka hur väl språkteknologiska metoder som enkel mönstermatchning med reguljära uttryck, stoppfilter och lexikonuppslagning med ordklasstaggar fungerar vid skapandet av ett program. Programmet genererar övningar i form av stavningsövningar på meningsnivå för andraspråksinlärare. Detta sker utifrån en text. Programmet är en applikation till ett system för inlärning av språk.

Grunden till vårt program ligger i ett befintligt CALL-system (Computer-Assisted Language Learning system) som använder sig av morfologiskt analyserad text för att kunna generera luck-övningar i form av grammatikövningar på meningsnivå.

Stavningsövningarna är ett önskemål företaget fått från en kund. De specifika bokstavsklusterna har vi tagit fram genom att se till målgruppens behov och vi har valt att generera övningar med suffixen -en, -et, -arna, -nas, -ar, -re, -ande, hela alfabetet, ck/k förutom i ordbörjan, ng/gn i ordslut, sje-ljud, o/å, e/ä, å/ä/ö, l/r och b/p. De språkteknologiska metoder vi använder för programmet är:

- **Enkel mönstermatchning** – Enkla reguljära uttryck används för att hitta bokstäver/bokstavskluster som ska bytas ut mot luckor i en text, till exempel 'ng' '_' byter ut alla ng i ordslut mot en lucka.
- **Stoppfilter** – Ord innehållande bokstäver/bokstavskluster som inte ljudas korrekt samt frekventa ord ska filtreras bort och programmet ska inte generera någon övning på dessa. Stoppfilter har tagits fram från en delmängd av SUC, närmare bestämt 10%.
- **Lexikonuppslagning med ordklasstaggar** – Suffixövningarna ska endast genereras på specifika ordklasser. Genom lexikonuppslagning baserad på taggar ur SUC märker vi bara upp de ord som kan generera övningar och istället för en fullständig ordklasstagning som taggar alla ord. Lexikonet består av ord med suffixen baserat på 90% ur SUC förutom de 101 meningar vi utvärderar på.

Övningarna utvärderas stegvis på en 101 meningar (testdata) från de resterande 90% av SUC med måtten precision, täckning och F-score. Vi har satt F-score på 90% för vad som är en godtagbar övning samt att felaktig ljudning och ordklass och även frekventa ord inte ska generera luckor. Därmed är större delen av luckorna korrekt genererade.

Resultatet av utvärderingen visar att alla övningstyper ger godtagbara övningar enligt våra kriterier. Vi har dock sett att vissa övningar är svårare att anpassa för alla användare beroende på skillnader i dialekt.

Abstract

In this thesis, we examine how well language technology methods such as simple pattern matching using regular expressions, stop filters and dictionary lookup with POS tags work for creating a program. The program generates fill-in-the-blanks in form of spelling exercises on the sentence level, based on text material, for second language learners. This program is an application for a second language learning system.

The starting point of our program is an existing CALL (Computer-Assisted Language Learning) system that uses a text that is morphologically analyzed to be able to generate fill-in-the-blanks as grammar exercises on the sentence level.

The spelling exercises have been developed from wishes that the company received from a client. The specific letter combinations are based on the target groups need for practice and we have chosen to generate exercises with the suffixes -en, -et, -arna, -nas, -ar, -re, -ande, the entire alphabet, ck/k apart from word beginnings, ng/gn in word endings, sje, o/å, e/ä, å/ä/ö, l/r och b/p. The language technology methods we use for the program are:

- **Simple pattern matching** – Simple regular expressions are used to find the letters/letter combinations to be exchanged with blanks in a text, eg 'ng' '_' replaces all ng in word endings with a blank.
- **Stop-filters** – Word containing the letters/letter combinations that don't have the right pronunciation and frequent words will be filtered out and the program will not generate exercises on these words. Stop filters have been developed by 10% of SUC.
- **Dictionary lookup using POS tags** – The suffix exercises should only be generated on specific POS-tags. By using dictionary lookup based on tags from SUC only the words that can generate blanks are tagged instead of a complete POS-tagger which give all words tags. The dictionary contains words from 90% of SUC (the 101 sentences for the evaluation omitted).

The exercises are evaluated step by step on 101 sentences of the remaining 90% of SUC, by measuring precision, recall and F-score. We have set F-score at 90% for an acceptable exercise and that wrong pronunciation and part-of-speech and also frequent words will not generate blanks. Thus, most of the blanks will be properly generated.

Our conclusion drawn from the evaluation results is that all exercise types generate acceptable exercises according to our criteria. We have seen that some exercises are more difficult to adapt to any user due to differences in dialect.

Förord

Först och främst vill vi tacka vår handledare Joakim Nivre, för att han hjälpt oss med tillvägagångssättet, uppsatsens struktur och innehåll. Vi vill tacka företaget, för att de tog emot oss och vår biträdande handledare Jonas Jonsson som kom med uppsatsförslaget, bistod med material, tips och hjälp. Sedan vill vi tacka Ola Knutsson på KTH i Stockholm för att vi fick tillgång till det befintliga programmet och för att han delade med sig av material, samt de kunder till företaget som kommit med synpunkter på hur vi skulle kunna anpassa vårt program för andraspråksinlärare. Sist men inte minst vill vi tacka vår korrekturläsare Viktoriya Kotelevskaya, för den tid hon lagt ner på att läsa och kommentera vår uppsats.

Innehåll

Sammandrag	ii
Abstract	iii
Förord	iv
Innehåll	v
Figurer	vii
Tabeller	vii
1 Inledning	1
1.1 Syfte	1
1.2 Målgrupp – svenska som andraspråk	2
1.3 Företaget	2
1.4 Användares önskemål	3
1.5 Uppdelning av arbete	3
2 Bakgrund	5
2.1 Datorstödd Språkinläring	5
2.1.1 Pedagogiska perspektiv	5
2.1.2 Utgångspunkter i olika CALL-system	6
2.1.3 Utvärderingsramar för CALL-system	7
2.2 Behov av datorstödd andraspråksinläring	8
2.3 Grammatikövningar på webben	8
2.3.1 Digitala Spåret	8
2.3.2 WERTi System	9
2.3.3 Studentlitteratur	10
2.3.4 Gramte	10
2.4 Språkteknologiska metoder	10
2.4.1 Mönstermatchning med reguljära uttryck	10
2.4.2 Stopppfilter	11
2.4.3 Lexikonuppslagning med ordklasstaggar	12
3 Tekniska utgångspunkter	14
3.1 Befintliga övningar	14
3.1.1 Program för luck-övningar	14
3.1.2 Användarperspektiv	15
3.1.3 XML definition	16

3.2	LanguageTool	17
3.3	Granska	17
4	Skapandet av ett nytt program	19
4.1	Övningarna	19
4.2	Programbeskrivning	20
4.2.1	Övningstyper	20
4.2.2	Huvudprogrammet	23
4.2.3	Implementering av stoppfilter	26
4.2.4	Lexikonuppslagning med ordklasstaggar	29
5	Utvärdering	33
5.1	Kriterier	33
5.2	Utvärdering med enkel mönstermatchning	34
5.3	Utvärdering med stoppfilter	34
5.4	Utvärdering med lexikonuppslagning	34
6	Diskussion av utvärderingsresultat	38
6.1	Enkel mönstermatchning	38
6.2	Tillämpning av stoppfilter	39
6.3	Lexikonuppslagning med ordklasstaggar	40
6.4	Sammanvägning av resultat och exempel på genererade övningar	40
7	Slutsats	50
	Litteraturförteckning	52

Figurer

4.1	Flödesschema för stavningsövningsprogrammet	24
4.2	Användarens vy i kommandotolk när 10 luckor för Suffixövning (a) är vald med lexikonuppslagning och XML-filer är skapade. "Unknown words" är de ord som inte funnits i lexikonet	25

Tabeller

1.1	Uppdelning av arbete	4
4.1	Omfattning av lexikonet med taggar	31
5.1	Utvärderingsresultat av enkel mönstermatchning	35
5.2	Utvärderingsresultat av stoppfilter	36
5.3	Utvärderingsresultat av lexikonuppslagning med ordklasstaggar	37
6.1	Samlade utvärderingsresultat	41

1 Inledning

I det här kapitlet presenteras syftet med uppsatsen, att undersöka hur väl tre språkteknologiska metoder fungerar vid skapandet av ett program som genererar stavningsövningar. Vi presenterar sedan vår målgrupp, invandrare som lär sig svenska. Sedan kommer vi i korthet att presentera företaget vi skapat applikationen för och användares önskemål samt hur vi delat upp arbetet.

1.1 Syfte

I denna magisteruppsats kommer vi att undersöka hur väl tre språkteknologiska metoder fungerar vid skapandet av ett program. Programmet genererar stavningsövningar i form av luck-övningar för suffix, hela alfabetet, ck/k förutom i ordbörjan, ng-ljud i ordslut, sje-ljud, o/å, e/ä, å/ä/ö, l/r och b/p.

De språkteknologiska metoderna är:

- **Enkel mönstermatchning** – Enkla reguljära uttryck används för att hitta bokstäver/bokstavskluster som ska bytas ut mot luckor till exempel 'ng' '_ ' byter ut alla ng i ordslut mot en lucka.
- **Stoppfilter** – Ord innehållande bokstäver/bokstavskluster som inte ljudas korrekt samt frekventa ord ska filtreras bort och programmet ska inte generera någon övning på dessa. Stoppfilter har tagits fram från en delmängd av SUC, närmare bestämt 10%.
- **Lexikonuppslagning med ordklasstaggar** – Suffixövningarna ska endast genereras på specifika ordklasser. Genom lexikonuppslagning baserad på taggar ur SUC märker vi bara upp de ord som kan generera övningar och istället för en fullständig ordklasstagning som taggar alla ord. Lexikonet består av ord med suffixen baserat på 90% ur SUC förutom de 101 meningar (testdata) vi utvärderar på.

Övningarna genereras på meningsnivå utifrån en text och sparas i XML-format, sedan importeras de till servern som företaget tillhandahåller. Målgruppen är individer som lär sig svenska som andraspråk.¹

För att göra oss bekanta med hur CALL-system (Computer-Assisted Language Learning system) kan vara uppbyggda, har vi tagit del av ett befintligt

¹Ordet 'andraspråk' används ofta som en "paraplyterm" för alla språk en inlärare lär sig efter sitt modersmål, detta kan vara missvisande då det egentligen syftar på det språk som lärs in i den miljö som språket används i och fungerar som det huvudsakliga kommunikationsspråket. Medan de språk som lärs in i en icke naturlig miljö, som exempel engelska i grundskolan, bör benämnas som "främmandespråk" (Abrahamsson, 2009, s.14).

program hos företaget. Detta program skapar grammatiska övningar, också i form av luck-övningar, baserad på genomförd grammatisk analys, lemmatisering, ordklassbestämning, morfologisk analys och ord översätts med lexikon. Övningarna sparas även här i XML-format se 3.1.3 XML-defition.

För varje stegvis tillämpning av en språkteknologisk metod utvärderas övningarna på vår testdata med måtten precision (antalet korrekta genererade övningar av de genererade övningarna), täckning (antalet korrekta genererade övningar av alla potentiellt korrekt genererade övningar) och F-score (se kapitel 5). För att en övning ska anses godtagbar bör den uppnå en F-score på 90%. För enkel mönstermatchning ska mönstren med korrekt ljudning och ordklass matchas. För stoppfilter ska mönstren med korrekt ljudning och ordklass matchas och även inte matcha frekventa ord. För lexikonuppslagning ska mönstren på korrekt ordklass matchas.

1.2 Målgrupp – svenska som andraspråk

Vid inläring av ett andraspråk kan det för många vara svårt att förstå hur det nya språket är upplagt. Vilken ordföljd som är korrekt och vilken grammatisk böjning ett ord har i olika sammanhang kan vara problematiskt. När vi tänker på andraspråksinläring är det för oss på grundskolenivå, att eleverna får välja om de vill lära sig tyska, franska eller spanska, men vi har även engelska (ett obligatoriskt ämne i svenska grundskolan), denna typ av inläring bör nämnas med ordet främmandespråk (Abrahamsson, 2009, kap.1), då de lärs in i en onaturlig miljö och inte används som huvudsakligt kommunikationsspråk. För de individer som kommer till Sverige och har ett annat modersmål än svenska ges möjlighet till svenskundervisning, en utbildning som ger kunskaper i svenska språket samt kunskaper om det svenska samhället. Efter en avslutad kurs ska inläraren ha viss hör-, läs- och skrivförståelse, samt kunna göra sig förstådd i tal. Svenska blir då ett andraspråk (Språkrådet, 2007).

I Stockholm finns möjligheten att använda datorstödd språkinläring² som kan göra inläringen mer flexibel, och på så sätt eventuellt öka inlärarens kunskapsnivå efter avslutad kurs. Därför är det intressant att få bidra med ett förslag på en ny applikation till ett datorstött språkinläringssystem.

1.3 Företaget

Företaget vi skapat programmet för är beläget norr om Stockholm och har utvecklat en elektronisk webb- och mobiltjänst för snabb inläring av översatta ord, grammatik- och dialogövningar för olika språk, som till exempel svenska, finska, ryska, engelska och tyska. Företaget är intresserade av att utöka sitt system med automatiskt genererade övningar utifrån en text. Företaget har av kommersiella skäl begärt att få vara anonyma, och kommer därför inte att nämnas vid namn i uppsatsen.

²<http://www.stockholm.se/ForskolaSkola/Svenska-for-invandrare-sfi/>

1.4 Användares önskemål

För att få inblick i vilka slags övningar som efterfrågas samt vad som kan vara användbart i systemet, har vi träffat och hört en kunds önskemål om övningar. Tillsammans med kunden kom vi fram till att vissa mål måste sättas upp för övningarnas utformning. Antalet meningar måste begränsas till en mening per sida (speciellt för mobilapplikationen). Detta för att det inte ska bli för mycket information på en gång för användaren. Det ska också bara komma upp en uppgift per sida. Svaren ska vara tydligt markerade med färg eller på annat utmärkande sätt.

Från en av kunderna fick vi genom företaget förslaget om bokstavsövningar i syfte att öva på det latinska alfabetet med svenska tecken, något vi intresserade oss för.

1.5 Uppdelning av arbete

Vi har till stor del arbetat gemensamt med uppsatsen men har huvudansvar för olika delar. Uppdelningen av arbetet redovisas i tabell 1.1.

Tabell 1.1: Uppdelning av arbete

Kap.	Avsnitt	Camilla	Rebecka
1		1 Inledning	1 Inledning
2	2.1		2.1 Datorstödd språkinläring
	2.2		2.2 Behov av datorstödd språkinläring
	2.3	2.3.2 WERTi System, 2.3.4 Gramte	2.3.1 Digitala Spåret, 2.3.3 Studentlitteratur
	2.4	2.4 Språkteknologiska metoder	
3	3.1	3.1.2 Användarperspektiv, 3.1.3 XML-defition	3.1.1 Program för luckövningar
	3.2		3.2 LanguageTool
	3.3		3.3 Granska
4	4.1	1. Suffixövning, 7. o/å-övning, 9. l/r-övning, 10. b/p-övning Motivering	2. Alfabetsövning, 3. ck/k-övning, 4. sje-ljud-övning, 5. ng-ljud-övning, 6. e/ä-övning, 8.å/ä/ö-övning Motivering
	4.2	4.2.2 Huvudprogrammet, 4.2.3 Implementering av stoppfilter	4.2.1 Övningstyper, 4.2.4 Ordklasstagning
5			5 Utvärdering
	5.1–5.3	Suffix, o/å, l/r, b/p	Alfabet, ck/k, sje-ljud, ng-ljud, e/ä, å/ä/ö
	5.4	5.4 Utvärdering med lexikonuppslagning	
6	6.1–6.2	Suffix, o/å, l/r, b/p	Alfabet, ck/k, sje-ljud, ng-ljud, e/ä, å/ä/ö
	6.3	Lexikonuppslagning med ordklasstagar	
	6.4	Suffix, o/å, l/r, b/p	Alfabet, ck/k, sje-ljud, ng-ljud, e/ä, å/ä/ö
7		7 Slutsats	7 Slutsats

2 Bakgrund

I detta kapitel har vi sett över begreppet "datorstödd språkinlärning" (Computer-Assisted Language Learning), behoven av datorstödd andraspråksinlärning, de språkteknologiska metoder vi använder oss av, samt befintliga grammatikövningar som finns på webben som inspirerar oss.

2.1 Datorstödd Språkinlärning

CALL (Computer-Assisted Language Learning) är den engelska termen för datorstödd språkinlärning och är en relation mellan utveckling av teknik och pedagogik. Det är ingen metod utan ett verktyg för att underlätta språkinlärning. Programvarorna finns ofta tillgängliga fritt via webbsidor eller som paketerade medier i cd-rom eller dvd, ofta för olika språk (Svensson, 2008). Att använda ett CALL-system kan till exempel ge större motivation (Svensson, 2010), förbättring av prestationen, ge ett mer erfarenhetsbaserat lärande för användaren, med mera. Stora framsteg har skett i utveckling av denna typ av system inom tal- och språkteknologi, och har ofta en mer direkt användning i språkutbildningen (Svensson, 2007). Eftersom vi skapar ett program som ska ingå i ett befintligt CALL-system anpassat för andraspråksinlärning har vi tittat på utgångspunkter i sådana system och hur de kan utvärderas.

2.1.1 Pedagogiska perspektiv

Framtagandet av ett CALL-system består bland annat av konceptdesign, pedagogisk principdesign (övergripande beskrivning av användargränssnittets utseende), verktyg för utformning, programmering, integration, testning samt marknadsföring.

Den lärare som är med i utvecklingsarbetet av ett CALL-system behöver inte ha kunskaper i programmering och programmeraren behöver inte ha kunskaper i det främmande språket, men en bra förståelse för NLP (natural language processing) (Gimeno-Sanz och Davies, 2010). Utvecklaren måste se över vem användaren av systemet är.

Några punkter att se över, innan skapande av ett program sker, är:

- Målgrupp
- Nivå på språket
- Generella/specifika ändamål

- Pedagogisk metod
- Användning av teknik

Huvudanvändaren i ett CALL-system är läraren som tillhandahåller innehållet. En uppsättning mallar ligger som grund för systemet och skrivs av en programmerare. Läraren behöver därför inte förstå programmering utan behöver bara bidra med systemets innehåll. Den grafiska designen, bland annat bilder, ikoner, teckensnitt, färg, skärmlayout bör utformas av en kvalificerad grafiker. Annars kan systemet misslyckas med att vara välutformat (Gimeno-Sanz och Davies, 2010). Ett område som tydligast förknippas med CALL är olika typer av övningsprogram för grammatisk och lexikal träning (Svensson, 2008). En typisk övning som lingvister och innehållsleverantörer använder för att designa ett CALL system är luckövningar (Gimeno-Sanz och Davies, 2010). Dessa kan gå ut på att en del av en text plockas bort och ersätts med en lucka där användaren ombeds att fylla i det korrekta svaret.

Inläraren

För inläraren måste tydliga instruktioner för programmet finnas (Gimeno-Sanz och Davies, 2010). Betoningen ska ligga på innehållet och inte hur innehållet ska användas. Det ska alltså inte vara oklart för inläraren hur ett program fungerar. Det ska vara klart vad programmet består av, var användaren är i programmet, var och hur användaren kan flytta sig samt vart användaren redan har varit. Övningarna får inte vara för omfattande och det ska finnas en ledtråd för vad som ska utföras. För varje övning ska det finnas tydliga mål för vad användaren uppnår. För texter bör det finnas svårighetsgrader. En övning ska helst kunna avslutas när som helst och det ska vara möjligt att fortsätta på en ofärdig uppgift.

Läraren

En steg-för-steg manual för installation av mjukvaran (om så krävs) och innehållet i programmet bör tilldelas läraren, likaså en felsökningsguide (Gimeno-Sanz och Davies, 2010). Om programvaran inte skulle fungera ska det finnas kontaktuppgifter för felanmälan. Läraren ska tilldelas information om vilka undersökningar och läroplaner innehållet hänför sig till och hur många timmars undervisning programmet består av. Programmet ska helst kunna anpassas för en specifik grupp inlärare.

2.1.2 Utgångspunkter i olika CALL-system

Datorn kan användas som lärare och som stimulans (Warschauer, 1996). Datorn som lärare:

- **Grammatikövningar** är utformade för grammatikundervisning i form av tränings- och övningsuppgifter på till exempel bestämd och obestämd artikel.
- **Hörövningar** är anpassade för andraspråksinläring, genom att inläraren kan lyssna på uttal.

- **Uttalsövningar** går ut på att eleven ska kunna spela in sin egen röst och sedan lyssna på den.
- **Läsövningar** omfattar läsprogram för andraspråksinlärning.
- **Text-återuppbyggnad** bygger upp text genom att användaren manipulerar bokstäver, ord och meningar.
- **Ordförrådsövningar** bygger på synonymövningar och målspråksordförråd.
- **Skrivövningar** använder en programvara som stöder ordbehandlare.

Inläraren får stimulans genom möjligheten att analysera, kritisera, diskutera och skriva:

- **Ordbehandling** är det vanligast använda verktyget för språkbehandling. Programvaror är till exempel Microsoft Word och Microsoft Works, vilka är delar av Microsoft Office-paketet.
- **Grammatikkontroll** är anpassad för modersmåltalare och identifierar de fel som en modersmålsanvändare gör när användaren skriver en text, exempelvis omkastning av bokstäver. De är därför inte rekommenderade för andraspråksinlärare.
- **Konkordans** är att användaren söker igenom en korpus för att hitta användningen av ett ord.
- **Kollaborativt skrivande** är när flera skriver på samma text för att till exempel färdigställa en giltig artikel. Wikipedia lämpar sig till exempel väl för kollaborativt arbete eftersom alla kan redigera en wikisida och tillsammans arbeta fram en text (Svensson, 2008).
- **Internet** bidrar med webbaserad undervisning inom tre populära användningsområden, webbläsare, e-post och MOOs (Multiple-user-domains Object Oriented) (Warschauer, 1996). Med en webbläsare som Mozilla Firefox kommer användaren åt internet. Flera program finns för e-post. MOOs möjliggör realtidskommunikation och rollspel med användare i hela världen.

2.1.3 Utvärderingsramar för CALL-system

När det gäller utvärdering av CALL-system kan det vara komplicerat att hitta rätt tillvägagångssätt. Det är mycket som måste tas hänsyn till. Det finns dock utformade ramar för hur en utvärdering skulle kunna göras. En utvärderingsram är utvecklad av Chapelle (Levy och Stockwell, 2006, s.61–65) och är ett teoribaserat och uppgiftsorienterat tillvägagångssätt. En annan utvärderingsram är Hubbards metodologiska tillvägagångssätt (Levy och Stockwell, 2006, s.59–61), där checklistor som riktar sig på just andraspråksinlärning kan användas. Chapelles och Hubbards utvärderingsramar bygger på olika ändamål. Chapelle menar att LLP ('language learning potential', det mått som riktar sig till kvaliteten av inlärningen) ska ha den högsta prioriteten och är det otvetydiga

ändamålet. Hubbard menar däremot att det finns tre områden som ska prioriteras, det första är förfarandet, vilket är ett mer objektiva tillvägagångssätt och det andra och tredje är lärarens och elevens utvärderingsmöjlighet, vilka är mer subjektiva. Det kan ses som att Chapelle har en mer teoridrivna utvärderingsram där inläraren står i fokus och Hubbard en bredare utvärderingsram som innefattar utvecklar-, lärar- och elev-utvärdering. Vår utvärdering kommer att vara fokuserad på utvecklingen som faller inom Hubbards ram, vilka övningar som genereras och vilka övningar som ska genereras.

2.2 Behov av datorstödd andraspråksinlärning

Om datorstödda språkverktyg bara konstrueras efter det egna språket (i detta fall svenska), blir feltyperna mycket svårare att detektera när en andraspråksinlärare sätter sig och tar hjälp av språkgranskning i till exempel en ordbehandlare för svenska. För att hitta de feltyper som inte detekteras i ett språkverktyg för svenska, när användaren är språkinlärare med ett annat modersmål än svenska, behövs det specifika regler i de datorstödda språkinlärningsverktyg som fångar de feltyper andraspråksinlärare gör. Dessa feltyper ser nämligen helt annorlunda ut än de feltyper som en som sitter och lär sig strukturen i sitt modersmålspråk gör (Öhrman, 2007). Det är även så med grammatiken. Modersmålsstalare av ett språk kan identifiera avvikelser i grammatiken hos andraspråkstalare som verkar vara gemensamt för alla andraspråkstalare oavsett modersmål (Abrahamsson, 2009). Med hjälp av datorstödd andraspråksinlärning kan andraspråksinlärares feltyper detekteras. Dessutom har andraspråksinläraren möjlighet att öva det nya språket var den än befinner sig (då det även finns mobilapplikationer för språkinlärning) samt att fortsätta öva på det nya språket efter avslutad SFI-kurs (svenska för invandrare). I och med att de datorstödda språkinlärningsverktygen finns tillgängliga på webben och via mobiltelefon blir det flexibelt och effektivt att lära sig ett nytt språk.

2.3 Grammatikövningar på webben

Genom en marknadsundersökning av vilka typer av grammatikövningar som finns tillgängliga på webben, fick vi nya idéer för utformning av övningar men också en uppfattning av vad som i praktiken inte fungerar för vår målgrupp, till exempel brist på snabb respons. Två av de system som inspirerade oss, Digitala Spåret och WERTi System, fick vi ta del av via kontakter till företaget. Övriga applikationer som vi tittat på letade vi fram på webben, men det var få som faktiskt stack ut.

2.3.1 Digitala Spåret

Digitala spåret¹ är ett projekt som avslutades 2007, vars mål har varit att stimulera språkträning för SFI (svenska för invandrare) och SAS (svenska som andraspråk). Projektet utfördes av lärare på Botkyrka vuxenutbildning och initierades av CFL, Centrum för flexibelt lärande, vars verksamhet övergått till

¹<http://www.digitalsparet.se/>

skolverket. Övningar är baserade efter nivåerna; Grund, B, C, D, SAS och GY (Svenska som andraspråk - gymnasiet) och består av läs, skriv och hör-övningar för olika kategorier som till exempel grammatik, samtal och tal, ordkunskap med mera. Webb-baserade övningar utifrån läroböcker är också tillgängliga. Några övningstyper som inspirerar är:

- Fyll i tomraden där användaren ombes att fylla i det ord som saknas på den tomma raden i en text.
- Fyll i tomraden där användaren ombes att fylla i det ord som saknas på den tomma raden med hjälp av en bild, till exempel välja att sätta en/ett framför ett givet ord med en bild föreställande ordet.
- Fyll i tomraden där användaren ombes att fylla i det ord som saknas på den tomma raden i en text med hjälp av en lista med alternativ.
- Drag n' Drop där användaren ska dra ett objekt till dess rätta position, till exempel placera ett ord på rätt plats i en mening.
- Flervalsfråga där användaren ska välja ett eller flera av de givna alternativen som passar in i texten.

Digitala spåret har ett brett utbud av grammatikövningar och intressanta övningstyper med tydliga förklaringar till övningarna. Dock bygger alla övningar på färdiga texter som också är nivåbaserade något som skiljer sig från det system vi arbetar i.

2.3.2 WERTi System

WERTi System² är ett så kallat ICALL (intelligent computer-assisted language learning) system som använder sig av NLP teknik (Metcalf och Meurers, 2006). Det är ett fritt tillgängligt webb-baserat system. I WERTi kan användaren välja att träna på engelska determinerare eller prepositioner och välja en verklighetsbaserad artikel som finns tillgänglig på internet (Ott och Meurers, 2009). Användaren börjar med att välja ordklass och sedan skriva in ett sökord och välja en artikel innehållande ämnet. Därefter ska användaren välja en av 3 övningstyper:

- Läs artikel där vald ordklass är markerade med en färg.
- Läs artikeln och samtidigt försöka hitta vald ordklass genom att klicka på orden. Rätt ord markeras med grönt och fel ord med rött.
- Läs artikeln och samtidigt fylla i de tomma raderna. Genom att klicka på ett frågetecken bredvid den tomma raden visas det rätta svaret.

Kombinationen av att välja sin egen text och samtidigt träna ordklasser med de ovan nämnda övningstyper gör att systemet påminner om den applikation vi arbetar med. Likheterna gör att WERTi System blir en väldigt intressant inspirationskälla.

Idag utvecklas systemet med bland annat Gerundium (Ott och Ziai, 2008).

²<http://prospero.ling.ohio-state.edu/WERTi/index.py/main>

2.3.3 Studentlitteratur

En annan webbtillgänglig grammatikövning är Studentlitteratur.se³. Denna sida har ett stort test på ca 80 frågor, inriktat på grundskole- och gymnasienivå, ej nybörjarnivå. Övningsfrågorna är flervalsfrågor inbyggda i färdiga meningar, innehållande alla typer av svensk grammatik. Det negativa med sidan är att alla frågor kommer upp på en gång, inte en och en, samt att inläraren därigenom inte får snabb återkoppling om de svarat rätt eller fel, då rättningen inte sker förrän användaren har svarat på alla frågor/övningar.

2.3.4 Gramte

Gramte har utvecklats vid Lunds Universitet och ger en möjlighet till repetition av svensk grammatik och grammatiska termer (Svensson, 2008). Gramte finns fritt tillgängligt att ladda hem på webben. Programmet erbjuder tre olika övningstyper (Lastow och Håkansson, 1997):

- Ordklasser – Användaren kan träna substantiv, adjektiv, pronomen, verb, adverb, adjektiv och adverb, prepositioner och partiklar, konjunktioner samt alla ordklasser. För substantiv till exempel får användaren några ord och ska fylla i om det är singular eller plural samt om det är obestämt eller bestämt. Dessutom finns det ett spel för varje övning där programmet stegar igenom en text och användaren ska klicka på en knapp varje gång programmet markerar en specifik ordklass i texten. Varje rätt klick ger ett antal poäng.
- Ordbildning – Användaren får en stam, till exempel "rök" och ska lägga till ändelser till stammen för olika ordklasser. I spalten för substantiv skulle användaren kunna skriva "rökare" och "rökelse", i adjektivspalten "rökande" och i verbspalten "röka".
- Satser/Satsdelar – Användaren ska fylla i om meningar består av en huvudsats och/eller bisats.

Det mest inspirerande var övningen med ordbildning där användaren ska träna ändelser.

2.4 Språkteknologiska metoder

För att skapa ett program som genererar luck-övningar i form av stavningsövningar på meningsnivå använder vi oss av tre språkteknologiska metoder, enkel mönstermatchning med reguljära uttryck, stoppfilter och lexikonuppslagning med ordklasstagning.

2.4.1 Mönstermatchning med reguljära uttryck

För att undersöka om en delsträng förekommer i en sträng används den enklaste formen av reguljära uttryck vilket är av en sekvens av tecken. Ken Thompsons editore av reguljära uttryck för sökning i text inkluderade kommandot

³<http://www.studentlitteratur.se/o.o.i.s/2529?csid=22&mp=8380>

”g/regularexpression/p”, ”Global Regular Expression Print”, grunden för Unix-kommandot ”grep”.

Sekvensen av tecken är det mönster som ska matchas i strängen. Mönstren har vi valt att definiera med snedstreck (/) precis som Jurafsky och Martin (2009, s. 51-59) gör och de reguljära uttrycken är av den typen Perl (Covens, 2000) använder. Vid implementationen har vi dock använt oss av Unix-kommandon⁴ som är snarlika de kommandon som används i Perl. Eftersom reguljära uttryck är känsliga för versaler och gemener matchar alltså inte mönstret /myskoxen/ ordet ”Myskoxen”. Istället måste klamrar användas som betecknar disjunktioner. [mM] matchar då mönster innehållande m eller M, som till exempel ”Myskoxen”. För att slippa problemet med versaler och gemener är det ett alternativ att använda en sträng som innan är konverterad till gemener.

För hitta mönstret /en/ i meningsslut används dollartecken (\$) som betecknar ordslut. Mönstret /en\$/ matchar då ”en” i strängen ”pågående utställning/en/”. Om mönstret /k/ ska matchas i meningbörjan i en sträng används brygga (^) framför mönstret. / ^ k/ matchar då det första k:et i strängen ”/k/onkursboet”.

Ett mönster bestående av en teckensekvens kommer att matcha alla ord innehållande mönstret. Mönstret /sjön/ matchar inte bara strängen ”sjön” utan även ”sjöns” och ”sjöng”. För att enbart matcha strängen ”sjön” används det reguljära uttrycket \b som matchar ord. Mönstret /\bsjön\b/ matchar då enbart strängen ”sjön”. För att enbart matcha en rad innehållande strängen ”Sjön.” används det reguljära uttrycket / ^ Sjön\.\$/ där taket (^) står för radbörjan, backslash (\) punkt (.) definierar att det är en punkt som ska matchas och inte vilket tecken som helst (som punkt också står för), samt dollartecken (\$) som står för meningsslut.

Ytterligare reguljära uttryck som brukar vara användbara för att hitta mönster, dessa är av typen Perl:

- [a-z] matchar en gemen: ”/n/ej det finns inte”
- [0-9] matchar en siffra: ”jörgen andersson /5/3 år från gimo”
- [^ A-Z] matchar inte en versal (när ^ förekommer inom klamrar): ”D/e/t är modernt”
- [^ Mm] matchar varken ”M” eller ”m”: ”/A/ntropologisk metod”
- t_o_m matchar: ”på en eftermiddag eller /t_o_m/ ännu kortare tid”
- den | det matchar: ”den” eller ”det”
- .* . är ett tecken, vilket som helst följt av * som är noll eller flera av det föregående tecknet

2.4.2 Stoppfilter

Att använda stoppfilter eller listor är vanligt förekommande för att hindra att ett program genererar för mycket, till exempel för många övningar eller fel övningar. Stoppfiltren är listor innehållande högfrekventa ord eller ord som

⁴<http://www.elektronen.kungsbacka.se/alf/unix/urval.html>

inte ska genereras i ett program av någon annan anledning. När stoppfilter används drivs det ofta av två motivationer, ett är att sortera bort ord med hög frekvens, det andra är att kunna sortera bort funktionsord, som är semantiskt lätta och har lite eller ingen betydelse för kontexten, för att spara utrymme och tid i programmet. Exempel på funktionsord är "på" och "i" (grammatiska ord). Problemet med att använda stoppfilter är att det är svårt att söka på sekvenser innehållande flera ord, fraser, då enskilda ord i den förekommande frekvensen kan förekomma i stoppfilteret, ett exempel som Jurafsky och Martin (2009, s.806) tar upp är frasen "to be or not to be" där endast "not" skulle genereras.

2.4.3 Lexikonuppslagning med ordklasstaggar

Ordklasstagning är en process där ord får etiketter i form av ordklass eller/och andra syntaktiska klasser i en korpus (Jurafsky och Martin, 2009, s.167). Före ordklasstagning görs en tokenisering som separerar orden. Skiljetecken och interpunktionstecken som till exempel punkt (.), kommatecken (,), kolon (:), semikolon (;) och talstreck (-) separeras också från orden och märks upp.

Ordklasstagning är en disambigueringsuppgift eftersom ord kan vara ambigüosa. Problemet med ordklasstagning är att lösa ambigüiteterna, att välja rätt ordklass för kontexten (Jurafsky och Martin, 2009, s.167). Ordklasstagare är antingen regelbaserade eller stokastiska. Regelbaserad taggare för automatisk tilldelning av ordklass använder lexikon och en mängd handskrivna disambigueringsregler (Ahrenberg, 2006) som avgör vilken ordklass ett ord tillhör genom att se till kontexten. En stokastisk taggare använder en uppmärkt träningskorpus (Ahrenberg, 2006) och beräknar sannolikheten för ett ord och en given tagg i en kontext med hjälp av markovmodellen (HMM - Hidden Markov Model).

Vi har använt lexikonuppslagning baserat på tokeniserade ord som har tilldelats en ordklasstagg. Detta är en förenklad metod till en fullständig ordklasstagning eftersom endast vissa ord (förekommande i vårt program) kommer att tilldelas en ordklasstagg. En fullständig ordklasstagning skulle tagga alla ord vilket vi anser kan ta längre tid. Metoden kommer att avgöra om detta är en tillräcklig metod eller om företaget istället bör implementera sin ordklasstagare.

Exempelmening med lexikonuppslagning baserad på SUC:

"och den pågående utställningen i kungshamn är hans första på västkusten sedan mitten på 60 talet"

Exempelmening på hur en mening kan se ut efter lexikonuppslagning med ordklasstaggar, de ord som finns är taggade:

och	
som	
den	dt
pågående	
utställningen	nn

i
kungshamn
är
hans
första
på
västkusten pm
sedan
mitten nn
på
60
talet nn

Lexikonet är baserat på 90% ur SUC (Ejerhed m.fl. (1992); Källgren (2006)) bortsett från 101 meningar vilka vi utvärderar med (testdata). Frekvenslistor till stoppfilter är baserade på de övriga 10% av SUC.

SUC - Stockholm Umeå Corpus

Under 1990-talet lades texter till SUC som följde principerna i The Brown Corpus och The Lancaster-Oslo/Bergen Corpus (LOB). Texterna kommer från olika genrer som reportage, recensioner, biografier, vetenskapliga artiklar med mera, och förekommer i genre-ordning. SUC består av 1 miljon ord som är taggade och annoterade med ordklasstagg, morfologisk böjningsform och lemma.

Version 1.0 utvecklades genom projektet "Korpusbaserad utveckling av modeller för datoranalys av löpande svensk text" mellan Stockholms Universitet och Umeå Universitet och gjordes tillgängligt 1997 av Institutionen för Lingvistik på Stockholms Universitet (Ejerhed m.fl., 1992).

Version 2.0 gjordes tillgänglig 2006 av Sofia Gustafson-Capková och Britt Hartmann 2006 vid Institutionen för Lingvistik på Stockholms Universitet. Den innehåller samma texter som SUC 1.0 men här är även punkter, citattecken, förkortningar och namngivna enheter annoterade. Dessutom innehåller SUC 2.0, TIGERSUC, en konvertering till TIGERxml samt STORSUC, extra textmaterial (Källgren, 2006).

SUC skapades som grund för utveckling, träning och testning av olika analyserare för obegränsade svenska texter och är fritt tillgängligt för forskningsändamål.

3 Tekniska utgångspunkter

I detta kapitel kommer vi att berätta om det system vi har som utgångspunkt för arbetet. Vi har tagit del av programmet för luck-övningar och sett över andra befintliga övningar i systemet ur användarperspektiv samt sett över LanguageTool och Granska, den morfologiska analysator och ordklasstagare som företagets system använder sig av.

3.1 Befintliga övningar

Användare av applikationen tränar sin svenska ordförståelse med hjälp av olika övningar som applikationen automatiskt skapar utifrån en övning, till exempel luck-övningar.

3.1.1 Program för luck-övningar

Programmet som genererar grammatikövningar är skrivet i Java och är ett samarbete mellan företaget och en utvecklare på KTH (Kungliga Tekniska Högskolan) i Stockholm. Det program vi tagit del av skapar luck-övningar i form av fråga-svar för fem ordklasser: substantiv, adjektiv, verb, determinerare och prepositioner. För varje ord förekommer också en ledtråd i form av ordets lemma. Övningarna skapas i XML-filer. Programkoden är uppdelad i fem klasser som hanterar olika delar för skapandet av grammatikövningar:

- **ExerGen** är den körbara filen som tar en textfil. Övningar skapas utifrån den ordklass som inläraren väljer att göra övningar på. Det går enkelt att utöka ordklasserna och information till gränsen av vald POS-taggar. Ordklasserna benämns endast med ordklasstaggen, till exempel "nn" för substantiv. Ordklasserna anropas som heltal 1-5 i programmet när övningarna skapas och de skrivs och sparas till en XML-fil.
- **ExerciseGenerator** skapar själva övningarna för en given text och ordklass. En hantering görs på de ord som inte ska generera övningar, till exempel frekvenshantering. Här ansluts även vald POS-taggar.
- **SentenceExercise** gör om formatet på texten till ord_lemma_ordklass för varje ord, separerade med mellanslag, till exempel "hundarna_hund_nn skäller_skälla_vb". Sammansatta särskrivningar som till exempel "New York" hålls samman med hjälp av #, "New#York". Ledtråden genereras för ordklasserna substantiv, verb och adjektiv, eftersom lemma för determinerare och prepositioner är själva ordet som efterfrågas. Övningarna returneras i så kallade ts-taggar <ts>, se 3.1.3.

- **WordList** ansluter till en vald tokeniserare och byter ut versaler till gemener och sparar ord, lemma och ordklass i en ordlista. Frekventa ord beräknas och filtreras bort när dessa anropas av ExerciseGenerator. Här finns också möjligheter för hantering av ordlistan.
- **WordData** lagrar den information om ordet som behövs nämligen lemma, ordklass, olika ord som tillhör ett lemma samt frekvens för lemmat. Hantering av ordklass i form av SUC-taggar sker, till exempel "vb" för verb.

3.1.2 Användarperspektiv

Här förklaras några fler av de befintliga övningarna sett ur användarperspektiv.

- **Luck-övningar** Användaren får en ledtråd till exempel "Ett organ som du tänker med" till en fråga som kan vara "Forskare vill upptäcka hur _ fungerar", och ska sedan skriva rätt ord på den tomma raden. När den tomma raden fyllts i, finns det en knapp som heter "acceptera" som användaren kan klicka för att avsluta uppgiften och kontrollera svaret.
- **Ordklass** Användaren får först ett ord, till exempel "hjärna" och dess definition "Ett organ på insidan av huvudet som vi tänker med" på skärmen och genom att klicka på knappen "visa" kommer ett svar fram i form av en ordklass, till exempel "substantiv". Användaren ska sedan välja om svaret är rätt eller fel genom att klicka på endera knapp.
- **Böjning** Användaren får ett ord i sin grundform till exempel "blomma" och dess definition "Planteras i kruka". Genom att klicka på knappen "visa" kommer ett svar fram i form av en böjning, till exempel "-or". Användaren ska sedan välja om svaret är rätt eller fel genom att klicka på endera knapp.
- **Valens** Användaren får ett ord till exempel "absorbera" och frågan hur ordet används i en mening. En definition till ordet ges också, till exempel "suga upp". Genom att klicka på knappen "visa" kommer ett svar upp på dataskärmen i form av ett ord och användaren ska välja om det är rätt eller fel genom att klicka endera knapp. En högtalarikon ger användaren möjlighet att lyssna på ordet och svaret. Användaren har möjlighet att gå bakåt till början av uppgiften.
- **Flervals-text, översättning** Användaren får ett ord och dess definition till exempel "blomma" och "Växer i rabatt". Ytterligare får användaren en lista med ord till exempel "floor", "flour", "flower", där användaren har möjlighet att välja rätt översättning. Genom att klicka på knappen "acceptera" kan användaren kontrollera om ordet användaren valt är rätt. Detta sker med en jämförelse mellan det ord användaren valt och det korrekta ordet som uppgiften genererar.
- **Flervals-textfråga** Användaren ombes att utifrån en given text till exempel "Vilken färg på trafikljus visar att det är okej att köra", välja det

korrekta svaret i en lista med alternativ, till exempel "grön", "guld", "choklad". Efter valet trycker användaren på knappen "acceptera" och då visas användarens svar tillsammans med det rätta svaret.

- **Kontext** Användaren får en fråga och en mening med ett tomrum, till exempel "Citroner växer i Medelhavsområdena. Jag tycker inte om _". Användaren ska fylla i tomrummet med vad användaren tror är rätt ord, genom att klicka på knappen "acceptera" kontrolleras svaret som användaren gett genom en jämförelse med det korrekta ordet.
- **Flervals-textfråga med kontext** Användaren får en multimedialadtråd, en fråga till exempel "En bok är.." och en lista med svar/ord som även kan vara ord på andra språk. Användaren ska välja ett av ordalternativen i listan för att sedan klicka på knappen "acceptera", för att jämföra sitt val med det korrekta svaret, som genereras automatiskt.

3.1.3 XML definition

Övningarna applikationen tillhandahåller definieras i XML-format. När användaren väljer lektion skickar servern den specifika XML-filen innehållande data till övningen. Här är några exempel på hur övningar kan definieras:

Ordklass-övningen:

```
<?xml version="1.0"?>
<ts version="1">
  <form>brain</form>
  <definition>An organ inside the head of an animal that enables thinking
    and feeling things such as heat or pain</definition>
  <pronunciation></pronunciation>
  <answer>noun</answer>
</ts>
```

Form-taggar innehåller den ordform som efterfrågas. Innanför definition-taggar finns texten som definierar det ord som användaren ska gissa ordklass på. Pronunciation-taggar är tomma eftersom det inte är något ljud kopplat till uppgiften. Till sist har vi answer-taggar som innehåller den rätta ordklassen.

Flervals-textfråga:

```
<?xml version="1.0"?>
<ts version="1">
  <form></form>
  <question>Which color of the semaphore shows that you may start driving?</question>
  <multichoice>
    <selection>
      <alt>red</alt>
      <alt>blue</alt>
      <alt>yellow</alt>
      <alt correct="TRUE">!green</alt>
```

```
<alt>light green</alt>
...
</selection>
</multichoice>
<pronunciation></pronunciation>
</ts>
```

Form-taggar är tomma eftersom ingen given ordform ska anges. Innanför question-taggar finns den fråga användaren ska svara på. Taggen multichoice innehåller flera alternativa svar som definieras som section och varje alternativ definieras i sin tur inom alt-taggar. Det rätta alternativet definieras som korrekt genom att bli tilldelad "TRUE". "pronunciation"-taggar är tomma eftersom det inte är något ljud kopplat till uppgiften.

3.2 LanguageTool

En morfologisk analysator bygger på lexikon och formella regler (Jurafsky och Martin, 2009, s. 87). När en morfologisk analys "MA" görs identifieras ordklass, böjningsform och lemma för ord utan kontext. En morfologisk analys av okända ord görs av ett program som hittar lemma och ändelse för det okända ordet. Språkgranskningsprogrammet LanguageTool används som morfologiska analysator i applikationen och bestämmer ordklass om ett ord är ambiguöst och har flera olika taggar, till exempel "var" som både taggas "nn" (substantiv) och "vb" (verb) i SUC beroende på kontext.

Under 2003 skrevs LanguageTool (Naber, 2003) i programspråket Python av Daniel Naber vid Universitetet i Bielefeld, Tyskland. Programmet utförde då en regelbaserad stil- och grammatikkontroll för engelska. Idag är det ett fritt tillgängligt källspråkgranskningsverktyg bland annat för svenska. Fördelen är att programmet hittar fel som vanliga språkgranskningsverktyg inte hittar, t.ex. om användaren blandat ihop ord som är lika i form. Den hittar också grammatikfel. Felen upptäcks med regler som är definierade i en XML-konfigurerad fil, men för att hitta mer komplicerade fel krävs en java-applikation. Den senaste versionen 1.0.0 är tillgänglig men fungerar bara med Java 5.0 och OpenOffice.org 3.0.1 eller senare version.

3.3 Granska

Under senare delen av 1990-talet utvecklades Granska¹, ett datorstött språkgranskningsprogram för svenska, på KTH i Stockholm, och vidareutvecklades 2001. Tillgängliga funktioner i programmet är språkgranskning, hjälpsystem med skrivregler samt sökning av ordklasser. Även tillgång till grundläggande ordbehandlingsfunktioner finns. Text behandlas genom analys och taggning med markovmodellen och SUC kompletterat med ord från SAOL (Svenska Akademiens Ordlista). I Granska matchas en ordsekvens i en text mot regler för att hitta eventuella fel. Feltyper som Granska detekterar är stavfel, felaktigt

¹<http://www.csc.kth.se/tcs/projects/granska/>

skrivna tecken, stilavvikelser och grammatikfel. Granska använder sig av regler där en av källorna är Svenska språknämndens skrivregler och ord märks upp (taggas) med ordklass och böjningsform. Granska används som POS-taggare i systemet och ordklasstaggar de texter som matas in i systemet för att generera ordklassövningar.

4 Skapandet av ett nytt program

Eftersom företaget eftersökte nya övningar som även skulle bidra till variation i systemet valde vi att skapa ett nytt program och utforma övningar utifrån förslaget om borttagning av bokstavskluster, vilket kom från en kund via företaget. Ord kan ha olika betydelser men låta lika med olika stavningar (Saeed, 2003, s.63). Vi har därför valt att bygga stavningsövningar på meningsnivå samt utöka antalet borttagningar (luckor) i övningarna. Idéer till utformningen har vi fått från det befintliga programmet som vi tagit del av. Först presenterar vi övningstyperna och sedan det program som vi skapat som genererar övningarna.

4.1 Övningarna

Vi har skapat tio nya övningar delvis baserat på en mindre undersökning gjord av Johanna Vitikka (Vitikka, 2009, s.2-4) där hon presenterar svenska språkljud som kan vara svåra för andraspråksinlärare. Övningarna har i grunden samma uppbyggnad, det som skiljer övningarna åt är borttagningen av olika bokstavskluster. Efter flertalet körningar har vi kommit fram till att sätta krav på omgivningen kring bokstavsklustrena. Dessa bokstavskluster har vi valt ut eftersom det kan vara svårt att skilja språkljuden åt i klustrerna.

1. **Suffixövning** för substantivets olika böjningsformer (-en, -et, -arna, -nas, -ar) och några av adjektivens, adverbens och participens böjningsformer (-re, -ande). Syftet med övningen är att träna ordens böjningsformer i kontext.
2. **Alfabetövning** där en bokstav på en specifik plats i meningen tas bort. Här är syftet att träna det latinska alfabetet med svenska tilläggen å, ä, ö. Förekommer bokstaven på flera platser i meningen tas även dessa bort.
3. **ck/k övning** där användaren ska träna skillnaden på dessa bokstavskluster som dock inte förekommer i ordbörjan och innan "s".
4. **sje-ljud övning** där användaren ska träna skillnaden på bokstavskluster som sch, ch, sh, sj och stj i ordbörjan och ordslut.
5. **ng-ljud övning** där användaren ska träna skillnaden på bokstavskluster som ng och gn i ordslut.
6. **e/ä övning** där användaren ska träna ä-ljud efter "r", "t" och "v" samt e-stavning före "n" i ordbörjan.
7. **o/å övning** där användaren ska träna å-ljud efter "m", "n", "b" och "p" samt o-stavning före "r" i ordbörjan, före "m" och efter "l".

8. **å/ä/ö övning** där syftet är att träna att använda bokstäver med prickar och ringar. "m", "n", "b" och "p" med efterföljande å, "v", "t", "g" och "m" med efterföljande ä och "r" och "f" med efterföljande ö samt ö med efterföljande "v".
9. **l/r övning** för den fonemiska variationens skull för till exempel de kantonesisktalande inlärarna. Klustrena är l med efterföljande "e" och "a", "e" med efterföljande r och r med efterföljande "ä".
10. **b/p övning** är två olika konsonantfonem som kan uttalas relativt lika i vissa sammanhang. Vi ställer inga krav på omgivning vid förekomster av b och p eftersom bokstäverna inte förekommer så frekvent. Däremot ska p inte matcha "ps" och "ph" i ordbörjan.

Exempel på dessa övningar kommer under avsnitt 4.2.2.

Motivationen till övningarna grundar sig i svenskans ganska komplicerade förhållanden mellan ljud och bokstäver som andraspråksinlärare kan få problem med om de inte fått möjligheten att lära sig svenska uttals- och skriftspråksregler (Vitikka, 2009, s.1). Ord kan låta lika men stavas olika. Dessa ord kan samtidigt ha olika betydelser och det är här valet att generera övningar på meningsnivå kommer in. Till exempel med orden *kål* och *kol* måste användaren se ordet i sin kontext för att avgöra vilken bokstav som ska in i luckan i o/å övningen. Eftersom det för inlärare är svårt att lära sig svenskans stavning av å-ljudet och sje-ljudet¹ samt avvikande uttal av ng-ljudet (Engstrand, 2009) valdes dessa som övningar. Syftet med övning 8 är att öva de extra bokstäverna i det svenska alfabetet, "å", "ä", "ö", där prickar och ringar ovanför bokstäverna ofta glöms bort. Övning 9 är till för inlärning av "l" och "r" för till exempel den fonemiska variationens skull för de kantonesisktalande inlärarna (Vitikka, 2009, s.7), samt övning 10 med "b" och "p", två olika konsonantfonem som kan uttalas relativt lika i vissa sammanhang, exempelvis i "snabbt" och "spik". Suffixövningen är skapad med inspiration från en idé om en kongruensövning.

4.2 Programbeskrivning

Programmet Exercises innehåller övningstyperna och huvudprogrammet. Alla övningar är definierade i separata metoder som läser in texten och returnerar luckövningar. Till suffixövningen går det också att generera övningar baserade på lexikonuppslagning. Då anropas ett lexikon innehållande ord som är ordklasstagade (tagna ur SUC). Övningen baserad på lexikonuppslagning utgår från samma principer som övningen utan ordklasstagning. Det finns dock ytterligare en metod som tar en sträng och ger en bokstav på en specifik plats. Bokstaven blir ytterligare ett parametervärde deklarerat som en character. Metoden är ett tillägg till alfabetsovningen för att kunna returnera den saknade bokstaven som svar.

4.2.1 Övningstyper

Övning med suffix

¹[http://www.langsci.ucl.ac.uk/ipa/IPAchart\(C\)2005.pdf](http://www.langsci.ucl.ac.uk/ipa/IPAchart(C)2005.pdf)

```

algoritm suffix(text)
  för varje ord i text
    ersätt suffixen "en" → "_",
    "et" → "_",
    "arne" → "_",
    "nas" → "_",
    "re" → "_",
    "ar" → "_",
    "ande" → "_"
returnera text

```

Övning med suffix med lexikonuppslagning

```

algoritm suffix(text)
  för varje ord i text
    hitta ord och tagg i lexikon
    ord → ord/tagg
  för varje ord/tagg i text
    ersätt suffixen "en/tagg" → "_",
    "et/tagg" → "_",
    "arna/tagg" → "_",
    "nas/tagg" → "_",
    "re/tagg" → "_",
    "ar/tagg" → "_",
    "ande/tagg" → "_"
returnera text

```

Övning med alfabet

```

algoritm alfabet(text)
  låt x vara tredje bokstaven i text
  för varje förekomst av x
    x → "_"
returnera text

```

Tar fram den saknade bokstaven i alfabetövningen

```

algoritm bokstaven(text, bokstav)
  låt x vara tredje bokstaven i text
  tredje bokstaven = x
returnera x

```

Övning med ck/k

```

algoritm ck(text)
  låt x vara bokstaven k eller ck förutom i ordbörjan
  för varje förekomst av x
    x → "_"
returnera text

```

Övning med ck/k med modifiering

algoritm ck(text)

låt x vara bokstaven k eller ck förutom i ordbörjan och efter s
för varje förekomst av x
x → "_"

returnera text

Övning med sje-ljud

algoritm sje(text)

för varje ord i text

ersätt sje-ljuden Ordbörjan "sch" → "_",
Ordbörjan "ch" → "_",
Ordbörjan "sh" → "_",
Ordbörjan "sj" → "_",
Ordbörjan "stj" → "_",
Ordslut "sch" → "_",
Ordslut "sh" → "_",

returnera text

Övning med ng/gn

algoritm ng(text)

för varje ord i text

ersätt ng-ljuden Ordslut "ng" → "_",
Ordslut "gn" → "_",

returnera text

Övning med e/ä

algoritm eä(text)

för varje ord i text

ersätt ä-ljuden Ordbörjan "en" → "_n",
"rä" → "r_",
"tä" → "t_",
"vä" → "v_",

returnera text

Övning med o/å

algoritm oå(text)

för varje ord i text

ersätt å-ljuden Ordslut "or" → "_r",
"om" → "_m",
"lo" → "l_",
"må" → "m_",
"nä" → "n_",
"bå" → "b_",
"på" → "p_",

returnera text

Övning med å/ä/ö

```

algorithm åö(text)
  för varje ord i text
    ersätt "må" → "m_",
           "nä" → "n_",
           "bå" → "b_",
           "på" → "p_",
           "vä" → "v_",
           "tä" → "t_",
           "gä" → "g_",
           "mä" → "m_",
           "rö" → "r_",
           "fö" → "f_",
           "öv" → "_v",
  returnera text

```

Övning med l/r

```

algorithm lr(text)
  för varje ord i text
    ersätt "le" → "_e",
           "la" → "_a",
           "er" → "e_",
           "rä" → "r_",
  returnera text

```

Övning med b/p

```

algorithm bp(text)
  för varje ord i text
    ersätt "b" → "_",
           "p" → "_",
  returnera text

```

Övning med b/p med modifiering

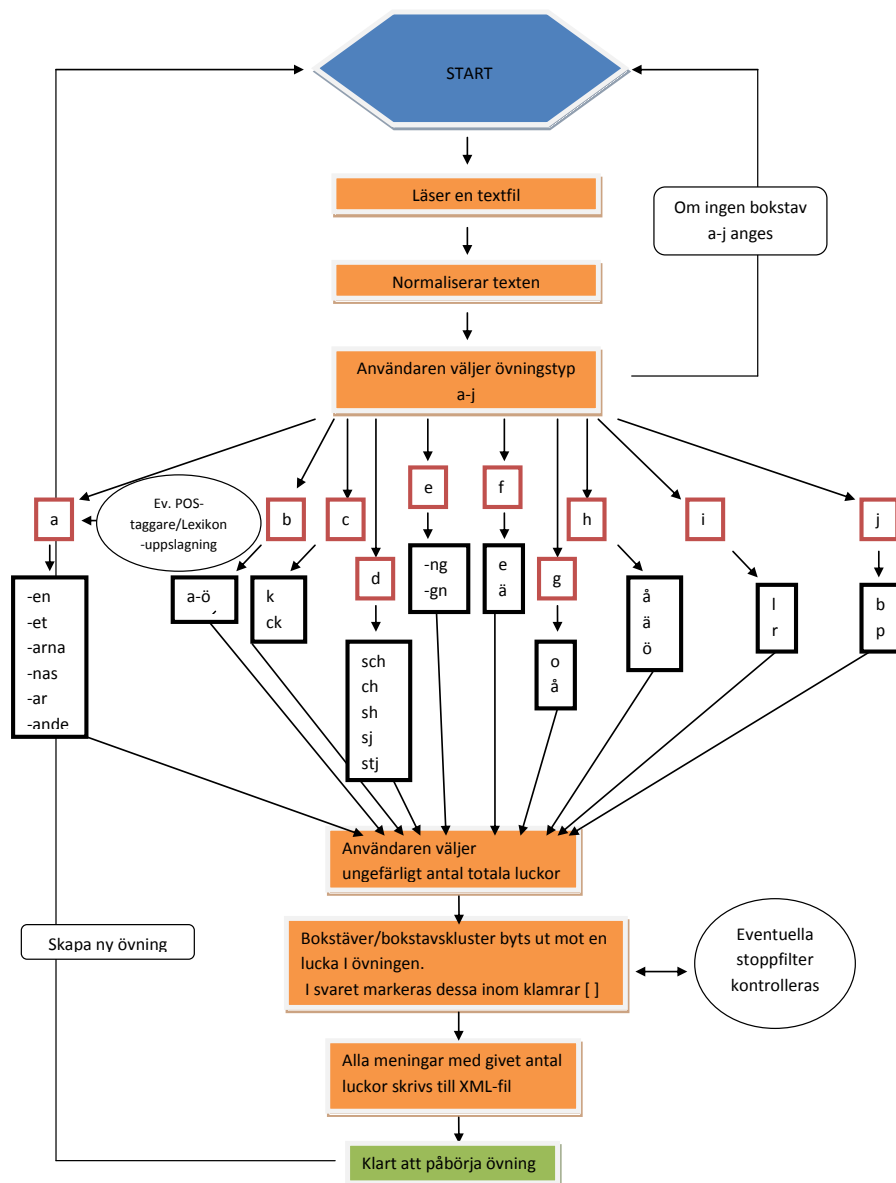
```

algorithm bp(text)
  för varje ord i text utom p framför s och h i ordbörjan
    ersätt "b" → "_",
           "p" → "_",
  returnera text

```

4.2.2 Huvudprogrammet

Programmet tar en valfri svensk textfil av formatet ".txt". Det första steget i hanteringen av textsträngen är en meningsindelning samt en normalisering av versaler till gemener, interpunktionerna utropstecken (!) och frågetecken (?) blir punkt, medan andra specialtecken blir blankslag. I programmet har vi använt oss av enkel mönstermatchning i liknande form av reguljära uttryck där specifika delsträngar av en mening matchas och om dessa mönster finns, byts dessa ut mot en lucka. Svaret hanteras genom att mönstren markeras med hakparenteser i svarssträngen. I de fall där mönstren ersatts med luckor skapas



Figur 4.1: Flödesschema för stavningsövningsprogrammet

en XML-fil innehållande övningstyp, uppgiftsbeskrivning, övningssträngen och svarssträngen, se bild 4.1.

För att starta programmet anropas programmet och namnet på den textfil som det ska genereras övningar på. En meny visas med de olika övningstyperna listade a-j. Användaren ombes att välja en av övningstyperna a-j och även ett ungefärligt antal luckor som ska genereras totalt, se bild 4.2. Användarens val av luckor blir ett ungefärligt antal eftersom när sista luckan genererats genere-

Please select exercise type a-j:

- (a) Suffix
- (b) Alphabet
- (c) k/ck exercise
- (d) sch/ch/sh/sj/stj exercise
- (e) ng/gn exercise
- (f) e/ä exercise
- (g) o/å exercise
- (h) åäö exercise
- (i) l/r exercise
- (j) b/p exercise

a

Please enter total number of blanks: 10

```
Suffix_exercise is saved in sentence1.Suffix_exercise.xml
Suffix_exercise is saved in sentence5.Suffix_exercise.xml
Suffix_exercise is saved in sentence6.Suffix_exercise.xml
Suffix_exercise is saved in sentence7.Suffix_exercise.xml
Suffix_exercise is saved in sentence8.Suffix_exercise.xml
Suffix_exercise is saved in sentence9.Suffix_exercise.xml
Suffix_exercise is saved in sentence10.Suffix_exercise.xml
```

Exercises Done for:

7 / 106 sentences

13 / 145 blanks in whole text.

5 Unknown words with blank.

Figur 4.2: Användarens vy i kommandotolk när 10 luckor för Suffixövning (a) är vald med lexikonuppslagning och XML-filer är skapade. "Unknown words" är de ord som inte funnits i lexikonet

ras också eventuella övriga luckor i samma mening. Har användaren bett om 10 luckor kan alltså 11 luckor genereras något som programmet tydligt talar om. Detta för att vi vill generera övningar på hela meningar. Om suffixövningar skapas med lexikonuppslagning kan också antalet okända ord (unknown words), ord som inte funnits i lexikonet, beräknas. De genererade övningarna inklusive svarsmeningar sparas i XML-format, i vilka användaren inte fysiskt kan fylla i svaret i luckorna. Programmet ska importeras till företagets server där de genererade övningarna visas en och en och användaren kan fylla i luckorna för att sedan visa svaret.

Exempel på genererade övningar

Några exempel på de genererade övningarna och deras svar tagna ur vår testdata. Det skiljer lite i hur meningarna som genereras ser ut. Det bör nämnas att testmängden innehåller längre och mer avancerade meningar än vad som är tänkt för andraspråksinlärare. Detta för att få så täckande texter som möjligt till utvärdering då testmängden ändå är relativt liten. I suffixövningen visar vi hur programmet tar specifika ordslut på ord som har tilldelats ordklasstagg av lexikonet och hur svaret ser ut, att samma mening genereras med svaren

uppmärksatta med hakparenteser. För fler exempelmeningar se 6.4.

Suffixövning

Övning: *uppsättning_ blev om jag rätt kommer ihåg den en lys_ uppvisning av tvillingpar_ philip zandén och krister henriksson men rest_ en mer löst hopfogad inventering av ensamhetens tvåsamhetens och abnormitetens många aspekter*

Svar: *uppsättning[en] blev om jag rätt kommer ihåg den en lys[ande] uppvisning av tvillingpar[et] philip zandén och krister henriksson men rest[en] en mer löst hopfogad inventering av ensamhetens tvåsamhetens och abnormitetens många aspekter*

Alfabetsövningen visar däremot inte hela meningen som svar utan endast den borttagna bokstaven.

Alfabetsövning

Övning: *ka_ske också e_ skogsbra_d _ågo_sta_s*

Svar: *n*

ck/k-övningen är lik suffixövningen, men skillnaden är att vi har valt att övningen inte ska genereras på ordbörjan, då "ck" aldrig förekommer i början av orden. Det blir därmed för självklart och enkelt att tillåta övningar genererade på "k" i ordbörjan då "k" är det enda svarsalternativet. Men svaret hanteras lika som i suffixövningen.

ck/k övning

Övning: *socialtjänsten ska med hänsyn till människans ansvar för sin och andras sociala situation inri_tas på att frigöra och utve_la enskildas och grupper resurser*

Svar: *socialtjänsten ska med hänsyn till människans ansvar för sin och andras sociala situation inri[k]tas på att frigöra och utve[ck]la enskildas och grupper resurser*

4.2.3 Implementering av stoppfilter

Eftersom utvärderingen (se avsnitt 5.2) av enkel mönstermatchning visar att övningar genereras på ord med fel ljudning och fel ordklass kom vi fram till att ytterligare implementationer var en nödvändighet för att se om dessa höjde kvaliteten. Genom att tillämpa stoppfilter kan förhoppningsvis felgenererade ord stoppas. Genom tillämpningen av stoppfilter väljer vi dessutom att filtrera bort frekventa ord eftersom företaget själva inte genererar övningar på för frekventa ord.

När vi har implementerat stoppfiltern har vi implementerat två olika filter, ett som återställer frekventa ord i övningen, och ett som återställer ett redan genererat svar, då övning och svar ser olika ut och inte kan hanteras på samma sätt.

Vi valde att använda 10% ur SUC (var tionde mening, med och utan

ordklasstag) för att hitta frekventa ord, vi kallar detta vår frekvensdata. Tanken är att dela upp de övriga 90% till testdata och lexikon. På så sätt kan stoppfilter och lexikon även användas på andra texter.

Val av stoppord

För varje övning har vi tittat på de tio mest frekventa orden innehållande respektive bokstav/bokstavscluster. Alla ord med en frekvens på 50 och mer filtreras bort. Detta för att inte generera övningar på alltför frekventa ord något som företaget uttryckt som önskemål eftersom det då kan genereras flertalet övningar på ett och samma ord.

Vi har även valt att uttöka frekvenslistan om vi ser att det tionde mest frekventa ordet har en frekvens på över 100. Då har vi tittat på ytterligare tio ord, för att inte missa de mest frekventa orden.

För suffixövningarna valde vi att även se över frekvenserna med den ordklasstagade mängden för att kunna filtrera bort både vanligt förekommande ord och vanligt förekommande ord av fel ordklass.

För alfabetsövningen uteblir stoppfilter, detta för att texten redan till stor del normaliserats vid inläsning av textfilen.

För övningarna med ck/k, sje-ljuden, ng-ljuden, e/ä och o/å, åäö, l/r och p/b gäller det att uttalen är korrekta mot respektive ljud samt avvikande ord med hög frekvens filtreras bort.

Korrekta ljudningar listas nedan:

- "k" och "ck" ska ljudas /k/
- 'gn" och "ng" ska ljudas nasalt-glottalt
- "ch", "sh", "sch", "sj" och "stj" ska ha sje-ljudning ljudas
- "o"/"å" ska ha å-ljudning
- "e"/"ä" ska ha ä-ljudning
- "l"/"r" ska ljudas efter respektive bokstav
- "b"/"p" ska ljudas efter respektive bokstav

Nedan presenteras de övningar som kommer att bli aktuella för tillämpning av stoppfilter.

Suffixövning

För att ta fram frekventa ord med suffixen sökte vi igenom hela frekvensmängden och tog först fram en lista över de tio mest frekventa orden med de specifika suffixen. Efter iakttagelser i den frekvenslista vi fått, valde vi att även titta på den ordklasstagade delmängden och specifikt filtrera bort det mest frekventa orden av fel ordklass. Vi väljer att filtrera bort dessa för att minska antalet felgenererade luckor, då övningarna ska genereras på specifika ordklasser. Stoppfilter har tillämpats på alla ord innehållande suffix utom -arna, -nas och -ande eftersom dessa tre suffix förekommer i ord som hamnar under vår gräns för vad som anses vara frekventa ord, det vill säga under 50 förekomster.

Exempel på ord som filtrerats bort:

öppen
nämligen
tvungen
mycket
eget
finnas
har
klar
mindre
vore
förhållande

ck/k

För att övningarna inte ska upprepas alltför många gånger, har vi valt att titta på de tio mest frekventa orden där "k" förekommer. Ord med "k" som begynnelsebokstav behöver vi inte ta hänsyn till då programmet inte genererar övningar dem.

Vi tittar även enskilt på de tio mest frekventa orden med "ck". För att se om vi missat något högfrekvent ord i bortfiltreringen. De mest frekventa orden med "ck" visar sig vara detsamma som för "k".

Exempel på ord som filtrerats bort:

olika
vilket
också
mycket
fick

e/ä

Frekventa ord med kluster innehållande "e" och "ä" filtreras bort.

Exempel på ord som filtrerats bort:

en
enligt
eller
rätt
väl

o/å

För de olika klusterna innehållande "o" och "å" har vi sett över de 10 mest frekventa orden. För "om" filtrerar vi bort de nio mest frekventa orden. För klusterna "lo" och "or" i ordbörjan behövs inga stoppfilter då frekvenserna inte alls var höga. För "må", "nå", "bå" och "på" filtreras enstaka ord bort innehållande klusterna.

Exempel på ord som filtrerats bort:

som
om

måste
någon
på

å/ä/ö

Vi har tittat på de tio mest frekventa orden innehållande "å", "ä" och "ö" i klustrena "må", "nä", "bå", "på", "tä", "vä", "gä", "mä", "rö", "fö" och "öv".

Exempel på ord som filtrerats bort:

väl
gäller
för
över

l/r

Vi har tittat på de tio mest frekventa orden innehållande "l" med efterföljande "e" och "a" samt "e" med efterföljande "r" och "r" med efterföljande "ä". Endast "rä" hade såpass låg frekvens att vi endast kunde stoppa ett ord.

Exempel på ord som filtrerats bort:

eller
skulle
alla
under
efter
rätt

b/p

När vi tittar på de tio mest frekventa orden med "b" eller "p", kan vi se att frekvenserna är så låga att vi inte behöver utöka frekvenslistan.

Exempel på ord som filtrerats bort:

blir
på
upp
knappast
fick

4.2.4 Lexikonuppslagning med ordklasstaggar

Vi har valt att använda lexikonupplagning bestående av ett lexikon innehållande ordklasstaggade ord för de relevanta suffixen, eftersom suffixövningen ska generera övningar på specifika ordklasser. Med lexikonuppslagningen matchas ett ord i en mening mot ett ord och en ordklasstagg i lexikonet. Ordet tilldelas då en tagg och baserat på ordets suffix och ordklasstagg avgör programmet om en lucka ska genereras samtidigt som ordklasstaggan tvättas bort. Detta anser vi är en tillräcklig metod eftersom alla ord inte behöver taggas. Lexikonet omfattar 24165 unika ord ur specifika ordklasser. Lexikonuppslagningen sker som

ett försteg till enkel mönstermatchning och stoppfilter när användaren väljer suffixövningen. Detta är ett tredje utvecklingssteg eftersom övningen ska genereras på specifika ordklasser. Nedan listas de relevanta ordklasserna för varje suffix:

- -en – substantiv, pronomen, adjektiv och determinerare.
- -et – substantiv, pronomen, verb och determinerare.
- -ar – substantiv och verb.
- -arna – substantiv.
- -nas – substantiv och verb.
- -re – substantiv, adverb och adjektiv.
- -ande – substantiv, adjektiv, particip och adverb.

Dessa är de ordklasser som kan generera suffixövningar. För att kunna generera en lucka måste ordet finnas med i lexikonet tillsammans med korrekt tagg, till exempel "stolar/nn" genererar en lucka eftersom att taggen "nn" annoterar ett substantiv. Vi vill även försöka generera luckor på ord av rätt ordklass som inte finns med i lexikonet och därför måste lexikonet även innehålla ord av fel ordklass för att inte generera lucka på fel ord. Till exempel så kommer "har/vb" inte att generera en lucka eftersom att "har" finns med i vårt lexikon med taggen "vb" som annoterar ett verb. Vi använder oss inte av morfologisk analys som skulle kunna hitta lemma och suffix för okända ord (Carlberger och Kann, 1999) (ord som inte finns med i lexikonet) och därmed även kunna stoppa generering av luckor på fel ord som inte finns med i lexikonet. Om till exempel ordet "myskoxen" inte förekommer i lexikonet kommer -en ändå att generera en lucka, detta för att inte missa någon lucka samtidigt som det finns en liten risk att ett ord av fel ordklass genererar lucka om det inte finns med i lexikonet.

Robustheten skulle öka vid implementering av en ordklasstaggar men med risk för minskning av effektivitet, eftersom det tar längre tid att tagga alla ord än bara de relevanta orden för övningen.

Tanken är att företaget kan använda sin egen ordklasstaggar om vi anser att lexikonuppslagning inte är en tillräcklig metod.

Disambiguering

Lexikonuppslagning med ordklasstaggar bestående av ord innehållande de suffix vi baserar övningar på kommer ge bekymmer med ambiguiteten. Lexikonet kommer att avgöra ordklass på ett ord efter dess placering i lexikonet, det första ordet i lexikonet som matchar ett ord i texten. Om ordet "var" påträffas i texten och "var/nn" ligger före "var/vb" i lexikonet kommer ordet få "nn"-tagg och generera en lucka.

Dock kommer vi lösa ambiguiteten i många fall då vi inte behöver hantera alla ord och dels beroende på vårt val av ordklasser som ska generera suffixövningar. Om vi gör ett stickprov i vårt lexikon ser vi till exempel att ordet

Tabell 4.1: Omfatning av lexikonet med taggar

Suffix	Totalt antal ord	Ordklasstagg	Antal ord/ordklasstagg
-en	10540	nn	9696
		pm	660
		jj	184
-et	4686	nn	4447
		pm	165
		jj	73
		vb	1
-ar	3769	nn	2441
		vb	1328
-arna	1347	nn	1347
-nas	641	nn	614
		vb	27
-re	1372	nn	1015
		jj	307
		ab	50
-ande	1810	pc	1261
		nn	410
		jj	97
		ab	42

"mindre" med suffixet -re både taggas som adjektiv (jj) och adverb (ab). Eftersom vi tillåter generering av övning både för adjektiv och adverb på suffixet -re behövs ingen disambiguering.

5 Utvärdering

För att veta om de övningar vi skapat verkligen gör det de ska använder vi kriterier och måtten precision, täckning (recall) och F-score (Jurafsky och Martin, 2009, s.489). Utvärderingen sker med alla luckor som genereras på respektive övningstyp och sker stegvis för varje tillämpad språkteknologisk metod på vår testdata (var 67e mening ur 90% ur SUC, vilka bearbetades om till 106 meningar i programmet). Vi har valt att använda SUC eftersom den innehåller en stor mängd av varierat textmaterial och är ordklasstaggad. På grund av upphovsrätten kan vi inte ta vilket textmaterial som helst och har därför valt att använda vår licens¹ för SUC.

5.1 Kriterier

För att utvärdera de övningar vi skapat har vi satt upp vissa kriterier för hur övningarna ska se ut, vad som är korrekt och inte vid tillämpning av varje språkteknologisk metod.

Enkel mönstermatchning – mönstren med korrekt ljudning och ordklass ska matchas.

Stoppfilter – mönstren med korrekt ljudning och ordklass ska matchas och dessutom inte matcha frekventa ord.

Lexikonuppslagning med ordklasstagg – mönstren på korrekt ordklass ska matchas.

En övning ska uppnå F-score på 90% för att anses användbar. Vi ser helst att F-score i slutändan når 100% men vi kommer aldrig att kunna uppnå 100% på alla övningar eftersom vissa av ljudningarna är dialektala och därför anser vi att 90% ska visa på en godtagbar övning. Måtten för utvärderingen blir följande:

$$\text{Precision} = \frac{\text{Number of correctly generated blanks}}{\text{Total number of generated blanks}}$$

$$\text{Recall} = \frac{\text{Number of correctly generated blanks}}{\text{Total number of potentially correct blanks}}$$

¹<http://www.ling.su.se/staff/sofia/suc/suc-license.html>

$$F - \text{score} = \frac{2 * (P * R)}{(P + R)}$$

Kriterier som är specifika för respektive övning:

- **Suffixens** kriterier för vad som är en korrekt övning är att ändelserna -en, -et, -arna, -nas och -ar enbart ska genereras på substantiv. Suffixen -re ska enbart generera övningar på adjektiv eller adverb medan -ande ska generera övningar på adjektiv, adverb eller particip.
- **Alfabetsövningen** utvärderas enligt kriteriet att programmet ska generera övningar på bokstäver ur det svenska alfabetet.
- **å/ä/ö** utvärderas enligt kriteriet att programmet ska generera luckor på dessa.
- **Stavningsövningarna med de olika bokstavsklustrerna** utvärderas utifrån kriteriet att det ska ha respektive ljudning för hela klustrer. Se 4.2.3 detaljerad beskrivning av respektive korrekt ljudning.

5.2 Utvärdering med enkel mönstermatchning

För varje övning undersöker vi antalet korrekta luckor och antalet felgenererade luckor mot de kriterier vi satt upp att korrekt ljudning och ordklass ska matchas. För alfabetsövningen tittar vi istället på antalet meningar då övningen tar bort alla förekomster av en och samma bokstav i en övning (eftersom alla luckor i en mening ger samma svar), annars genereras ingen övning. Resultaten av utvärderingen presenteras i tabell 5.1.

5.3 Utvärdering med stoppfilter

I tabellen 5.2 redovisar vi resultatet från vår andra utvärdering, när vi tillämpat stoppfilter för ord som har en frekvens på 50 förekomster eller mer (bland de 10-20 mest frekventa orden i frekvenslistan), uttal (med kriteriet om rätt ljudning) och ordklass.

5.4 Utvärdering med lexikonuppslagning

I tabellen 5.3 redovisar vi resultatet från vår tredje utvärdering som endast gjorts på suffixövningen. Tanken är att företaget ska implementera sin ordklasstaggare om vi anser att lexikonuppslagning med ordklasstaggar ökar antalet genererade luckor på korrekt ordklass.

Tabell 5.1: Utvärderingsresultat av enkel mönstermatchning

Övningstyp	Precision	Täckning	F-score
Suffix	(212/282) 75,2%	(282/282) 100%	85,8%
Alfabet	(95/95) 100%	(95/95) 100%	100%
ck/k	(186/206) 90,3%	(206/206) 100%	95%
ng/gn	(19/19) 100%	(19/19) 100%	100%
sje-ljud	(12/12) 100%	(12/12) 100%	100%
e/ä	(58/78) 74,4%	(78/78) 100%	85,3%
o/å	(127/135) 94%	(135/135) 100%	97%
å/ä/ö	(158/158) 100%	(158/158) 100%	100%
l/r	(280/292) 95,9%	(292/292) 100%	97,9%
b/p	(243/246) 98,8%	(246/246) 100%	99,4%

Tabell 5.2: Utvärderingsresultat av stoppfilter

Övningstyp	Precision	Täckning	F-score
Suffix	(192/232) 82,8%	(232/232) 100%	90,5%
Alfabet	–	–	–
ck/k	(118/119) 99,2%	(119/119) 100%	99,6%
ng/gn	–	–	–
sje-ljud	–	–	–
e/ä	(55/59) 93,2%	(59/59) 100%	96,5%
o/å	(37/44) 84%	(44/44) 100%	91,3%
å/ä/ö	(85/85) 100%	(85/85) 100%	100%
l/r	(228/228) 100%	(228/228) 100%	100%
b/p	(198/198) 100%	(198/198) 100%	100%

Tabell 5.3: Utvärderingsresultat av lexikonuppslagning med ordklasstaggar

Övningstyp	Precision	Täckning	F-score
Suffix	(132/138) 95,7%	(138/138) 100%	97,8%
Alfabet	–	–	–
ck/k	–	–	–
ng/gn	–	–	–
sje-ljud	–	–	–
e/ä	–	–	–
o/å	–	–	–
å/ä/ö	–	–	–
l/r	–	–	–
b/p	–	–	–

6 Diskussion av utvärderingsresultat

I detta kapitel diskuterar vi resultaten från utvärderingarna av övningstyperna.

6.1 Enkel mönstermatchning

Vi börjar med att presentera de övningar som gav bäst resultat.

Alfabetsoövningen ger en F-score på 100% och genererar endast luckor på bokstäver. *Baserat på vårt stickprov krävs inga ytterligare språkteknologiska metoder.*

ng/gn ger en F-score på 100%. Övningen genererar totalt 19 luckor. *Baserat på vårt stickprov krävs inga ytterligare språkteknologiska metoder.*

sje-ljud har en F-score på 100% och uppfyller kravet om en F-score på 90%. Det genereras totalt 12 luckor. *Baserat på vårt stickprov krävs inga ytterligare språkteknologiska metoder.*

å/ä/ö ger även den en F-score på 100%. 158 luckor genereras totalt. Klustren genererar luckor på ord som anses frekventa, till exempel "på" och "för" vilket egentligen är tillåtet. Detta kan förhindras med stoppfilter vilket vi också testat för att förbättra övningen ytterligare.

b/p ger en F-score på 99,4% vilket är över väl godkänt. Övningen genererar 246 luckor. Några ord som genereras fel är "philip", "phillipe" och "psykolog", dessa typer av p-förekomster går också att filtrera bort med stoppfilter för att förbättra övningen. Vi kan också se över frekventa ord.

l/r ger en F-score på 97,9%. Den genererar 292 luckor. Frekventa ord innehållande klustrena och kluster som inte ljudas rätt som till exempel när /r/ förekommer framför "s" och "t" kan filtreras bort med stoppfilter.

o/å ger en F-score på 97%. Övningen genererar totalt 135 luckor. Till större del genereras luckorna på 'o' och 'å' med å-ljudning, vilket är enligt våra kriterium. Många av luckorna genereras dock på alltför frekventa ord, vilka kan uteslutas genom att använda stoppfilter för de frekventa orden.

ck/k ger en F-score på 95%. Övningen genererar 206 luckor. De felgenererade övningarna kan förhindras genom stoppfilter av ord där inte /k/

ljudas, till exempel "sk". Vi kan också se över frekventa ord.

Suffixövningen når inte riktigt kravet på ett F-score på minst 90% utan hamnar på 85,8%. Övningarna genererar totalt 282 luckor. Felen ligger i att övningar genereras på ord av fel ordklass. Till exempel genereras luckor på en del verb med suffixet -ar, en del substantiv på suffixet -re och andra ordklasser än substantiv på suffixet -en. Ett stoppfilter baserat på frekventa ord med suffix av fel ordklass kan minska antalet felgenererade suffix. Vi kan också se över frekventa ord av rätt ordklass.

e/ä har en F-score på 85,3%. Totala antalet luckor är 78. Stoppfilter behövs för ord där inte /ä/ ljudas. Vi kan också se över frekventa ord.

För en överblick av resultatet, se tabell 5.1.

6.2 Tillämpning av stoppfilter

Vi börjar med att presentera de övningar som gav bäst resultat.

å/ä/ö ger en F-score på 100%. Övningen genererar totalt 85 luckor efter bortfiltrering av frekventa ord, till exempel "på" och "för". *Baserat på vårt stickprov krävs inga ytterligare språkteknologiska metoder.*

b/p ger en F-score på 100%. Övningen genererar totalt 198 luckor efter bortfiltrering av ph- och ps- i ordbörjan samt enstaka frekventa ord. *Baserat på vårt stickprov krävs inga ytterligare språkteknologiska metoder.*

l/r ger en F-score på 100%. Den genererar totalt 228 luckor. Nu genereras inga övningar på de "r" som inte ljudas /r/. *Baserat på vårt stickprov krävs inga ytterligare språkteknologiska metoder.*

ck/k ger en F-score på 99,6%. Övningen genererar 119 luckor varav endast ett fel. Det är när "k" förekommer i ordet "bekämpning". *Baserat på vårt stickprov krävs inga ytterligare språkteknologiska metoder.*

e/ä ger en F-score på 96,5%. "enskildas" och "ensamhetens" är exempel på ord som genererar luckor men som inte ska generera luckor. Även om övningen är acceptabel så styr användarens dialekt uttalet.

o/å övningen genererar totalt 44 luckor. Frekventa ord som "om" och "som" genererar ingen lucka på "o" eftersom de ligger i stoppfilter. Eftersom bland annat orden "om" och "som" innehåller "o" med /å/ ljudning minskar antalet luckor och därmed antalet korrekta luckor. Samtidigt kvarstår felet med "o" där inte /å/ ljudas som till exempel "blod" och "blomma" samt andra fel som är mer dialektala till exempel "antropologi", vilket ger en minskning av F-score till 91,3%.

Suffixen når nu kravet för F-score nämligen 90,5%. Övningarna genere-

rar nu totalt 232 luckor med 7 olika suffix. För att ytterligare försöka höja F-score kan ordklasstagning vara användbart och vi kommer att tillämpa lexikonuppslagning med ordklasstag för ord innehållande suffixen för att undersöka om övningen kan förbättras (se avsnitt 6.3).

För överblick över utvärderingsresultatet, se tabell 5.2.

6.3 Lexikonuppslagning med ordklasstaggar

Vi har tillämpat lexikonuppslagning med ordklasstaggar på suffixövningen eftersom det är den enda övning som har ordklasskriterium. Detta är ett förenklat tillvägagångssätt som i längden kanske inte är användbart även om det borde vara effektivare för vårt program. Tanken är att företaget kan implementera deras egen ordklasstaggar.

Suffixen genererar totalt 138 luckor och 37 av de luckorna är märkta som okända ord vilket visar att orden inte förekommer i vårt lexikon. Om vi bortser från de luckor som genererats på okända ord får vi 101 luckor som genererats med hänsyn till ordklasstag, alla korrekta.

Det ambigüosa ordet "var" genererar ingen övning eftersom lexikonet slår upp ordet som ett verb. Tittar vi på de luckor som genererats på okända ord är 31 av 37 egentligen korrekt genererade och därmed har vi inkluderat dessa som korrekta luckor.

Övningen når en F-score på 97,8% vilket är godtagbart och visar att både stoppfilter och lexikonuppslagning behövs för suffixövningen.

För en överblick av resultatet, se tabell 5.3.

6.4 Sammanvägning av resultat och exempel på genererade övningar

Vi har sammanvägt resultaten från utvärderingarna med tillämpning av de språkteknologiska metoderna och tagit eventuella förbättringar i procent räknat från enkel mönstermatchning till tillämpning av eventuella stoppfilter och/eller lexikonuppslagning i tabell 6.1.

Nedan presenterar vi exempel på hur genererade övningar inklusive svar ser ut för användaren. Meningarna är tagna ur vår testdata. Först tillämpas enkel mönstermatchning på alla övningar. Sedan har vi tillämpat stoppfilter som vi skapat med vår frekvensdata, för ord som inte ljudas korrekt, är av fel ordklass och frekventa ord på alla övningar utom alfabetsövningen, ng-övningen och sje-övningen. Dessutom har vi tillämpat lexikonuppslagning med ordklasstaggar för suffixövningen.

Suffixövning

ENKEL MÖNSTERMATCHNING

Tabell 6.1: Samlade utvärderingsresultat

Övning	Mått	Mönstermatch.	Stopppfilter	Ordklasstagging	Förbättring
Suffix	F-score	85,8%	90,5%	91,8%	6%
Alfabet	F-score	100%	–	–	–
ck/k	F-score	95%	99,6%	–	4,6%
ng/gn	F-score	100%	–	–	–
sje-ljud	F-score	100%	–	–	–
e/ä	F-score	85,3%	96,5%	–	11,2%
o/å	F-score	97%	91,3%	–	-5,7%
å/ä/ö	F-score	100%	–	–	–
l/r	F-score	97,9%	100%	–	2,1%
b/p	F-score	99,4%	100%	–	0,6%

Exempel 1:

Övning:

d_ ansträngda statsfinansiella läg_ med väx_ budgetunderskott utesluter _ fortsatt expansion av d_ offentliga sysselsättning_

Svar:

d[et] ansträngda statsfinansiella läg[et] med väx[ande] budgetunderskott utesluter [en] fortsatt expansion av d[en] offentliga sysselsättning[en]

Exempel 2:

Övning:

m_ d_ tragiska resultat_ av denna utveckling är d_ oändliga mångfald av samfund och grupper som protestantism_ på detta sätt h_ frambringat

Svar:

m[en] d[et] tragiska resultat[et] av denna utveckling är d[en] oändliga mångfald av samfund och grupper som protestantism[en] på detta sätt h[ar] frambringat

STOPPFILTER

Exempel 1:

Övning:

d_ ansträngda statsfinansiella läg_ med väx_ budgetunderskott utesluter en fortsatt expansion av d_ offentliga sysselsättning_

Svar:

d[et] ansträngda statsfinansiella läg[et] med väx[ande] budgetunderskott utesluter en fortsatt expansion av d[en] offentliga sysselsättning[en]

Exempel 2:

Övning:

men d_ tragiska resultat_ av denna utveckling är d_ oändliga mångfald av samfund och grupper som protestantism_ på detta sätt har frambringat

Svar:

men d[et] tragiska resultat[et] av denna utveckling är d[en] oändliga mångfald av samfund och grupper som protestantism[en] på detta sätt har frambringat

LEXIKONUPPSLAGNING MED ORDKLASSTAGGAR

Exempel 1:

Övning:

det ansträngda statsfinansiella läg_ med väx_ budgetunderskott utesluter en fortsatt expansion av den offentliga sysselsättning_

Svar:

det ansträngda statsfinansiella läg[et] med väx[ande] budgetunderskott utesluter en fortsatt expansion av den offentliga sysselsättning[en]

Exempel 2:

Övning:

men det tragiska resultat_ av denna utveckling är den oändliga mångfald av samfund och grupper som protestantism_ på detta sätt har frambringat

Svar:

men det tragiska resultat[et] av denna utveckling är den oändliga mångfald av samfund och grupper som protestantism[en] på detta sätt har frambringat

Med enkel mönstermatchning genereras luckor på alla suffix, det inklusive "den", "det", "har", "men" och "en" med suffixen -en, -et och -ar som endast ska generera övning på substantiv. Med tillämpning av stoppfilter kan vi stoppa "har", "men" och "en" som är frekventa ord av fel ordklass. Trots att "den" och "det" också är frekventa ord av fel ordklass fungerar det inte lika enkelt att filtrera bort dessa. Programmet blandar ihop orden vid generering av övningar, så att "den" blir "det", något vi antar beror på ordens längd. När vi tillämpar lexikonuppslagningen med taggar når vi alla ord av fel ordklass som finns i vårt lexikon och därmed genereras inga övningar på "den" och "det" eftersom de är taggade som determinerare. Ordet "protestantismen" blir taggat okänt eftersom att ordet inte finns med i vårt lexikon. Det är dock en korrekt genererad lucka eftersom att -en ska genereras på substantiv.

Alfabetsoövning

ENKEL MÖNSTERMATCHNING

Exempel 1:

Övning:

ov_ är rädd om skog_ns kronwilt och _ndast vissa djur får fällas

Svar:

e

Exempel 2:

Övning:

en_äten syftade främst till att erhålla information om hur de studerande såg på sina utbildningars innehåll och arbetsformer när de befann sig i mitten av sina högs_olestudier

Svar:

k

Exempel 3:

Övning:

jo_an sprang av oc_ an med kartonger oc_ påsar

Svar:

h

Det enda som krävs för att generera godtyckliga alfabetsövningar är att texten normaliseras vilket sker i programmet, detta görs med enkel mönstermatchning som endast matchar bokstäver.

ck/k-övning

ENKEL MÖNSTERMATCHNING

Övning:

socialtjänsten s_a med hänsyn till männis_ans ansvar för sin och andras sociala situation inri_tas på att frigöra och utve_la ens_ildas och grupper resurser

Svar:

socialtjänsten s[k]a med hänsyn till männis[k]ans ansvar för sin och andras sociala situation inri[k]tas på att frigöra och utve[ck]la ens[k]ildas och grupper resurser

STOPPFILTER

Övning:

socialtjänsten ska med hänsyn till människans ansvar för sin och andras sociala situation inri_tas på att frigöra och utve_la enskildas och grupper resurser

Svar:

socialtjänsten ska med hänsyn till människans ansvar för sin och andras sociala situation inri[k]tas på att frigöra och utve[ck]la enskildas och grupper resurser

Den enkla mönstermatchningen matchar alla "k" och "ck" förutom i ordbörjan, då "ck" inte förekommer som ordbörjan i svenska språket. I ordet "människans" har k:et sje-ljudning. En modifiering av mönstret hindrar sedan att övningar genereras när "k" förekommer efter "s" och bildar sje-ljudning. Med stoppfilter filtrerar vi bort frekventa ord.

sje-övning

ENKEL MÖNSTERMATCHNING

Exempel 1:

Övning:

det var något nytt som jag tolkar som utslag av sund _älvbevarelsedrift

Svar:

det var något nytt som jag tolkar som utslag av sund [sj]älvbevarelsedrift

Exempel 2:

Övning:

den grundläggande lagen är _erman act från 1890

Svar:

den grundläggande lagen är [sh]erman act från 1890

Exempel 3:

Övning:

henry stiglund hyllar sitt ursprung i den andra ändan av sverige den som vetter mot finland med ett för det skånska temperamentet exotiskt utspel fränt luktande av snus och mus brännvin och _amanism

Svar:

henry stiglund hyllar sitt ursprung i den andra ändan av sverige den som vetter mot finland med ett för det skånska temperamentet exotiskt utspel fränt luktande av snus och mus brännvin och [sch]amanism

För att generera övningar på "sch", "ch", "sh", "sj" och "stj" byts dessa mönster ut mot luckor med enkel mönstermatchning. Kraven som vi ställt är att sje-ljuden ska vara antingen i ordbörjan och/eller ordslut, detta förhindrar att vissa kombinationer av sammansättningar som ger de olika sje-stavelserna inte genereras och behöver inget stoppfilter på grund av den låga frekvensen av ord med sje-ljudning.

ng-övning

ENKEL MÖNSTERMATCHNING

Övning:

nu stampar västerbottningarna också i gå_ nordic swi_ festival 21 22 april

Svar:

nu stampar västerbottningarna också i gå[ng] nordic swi[ng] festival 21 22 april

Med enkel mönstermatchning matchas -ng och -gn endast i ordslut för att undvika att övningar ska genereras på sammansättningar och därmed inte ljudas korrekt med nasal-velar och nasal-uvular (IPA, 2005). I detta fall kommer övningar alltid att genereras på korrekt ljudade bokstavskluster.

e/ä-övning

ENKEL MÖNSTERMATCHNING

Exempel 1:

Övning:

uppsättningen blev om jag r_tt kommer ihåg den _n lysande uppvisning av tvillingparet philip zandén och krister henriksson men resten _n mer löst hopfogad inventering av _nsamhetens tvåsamhetens och abnormitetens många aspekter

Svar:

uppsättningen blev om jag r[ä]tt kommer ihåg den [e]n lysande uppvisning av tvillingparet philip zandén och krister henriksson men resten [e]n mer löst hopfogad inventering av [e]nsamhetens tvåsamhetens och abnormitetens många aspekter

Exempel 2:

Övning:

lorenz pekade på alla svårigheter med att förutsäga v_dret och på att prognoser aldrig kommer att kunna göras för _n längre period

Svar:

lorenz pekade på alla svårigheter med att förutsäga v[ä]dret och på att prognoser aldrig kommer att kunna göras för [e]n längre period

STOPPFILTER

Exempel 1:

Övning:

uppsättningen blev om jag rätt kommer ihåg den en lysande uppvisning av tvillingparet philip zandén och krister henriksson men resten en mer löst hopfogad inventering av _nsamhetens tvåsamhetens och abnormitetens många aspekter

Svar:

uppsättningen blev om jag rätt kommer ihåg den en lysande uppvisning av tvillingparet philip zandén och krister henriksson men resten en mer löst hopfogad inventering av [e]nsamhetens tvåsamhetens och abnormitetens många aspekter

Exempel 2:

Övning:

lorenz pekade på alla svårigheter med att förutsäga v_dret och på att prognoser

aldrig kommer att kunna göras för en längre period

Svar:

lorenz pekade på alla svårigheter med att förutsäga v[ä]dret och på att prognoser aldrig kommer att kunna göras för en längre period

ä-ljudningar förekommer på både "e" och "ä" med krav på omgivningen och ett första steg är att med enkel mönstermatchning generera övningar på dessa och undersöka vilka som faktiskt ljudas korrekt. Stopppfilter stoppar sedan "e" och/eller "ä" i frekventa ord, till exempel "en" och "rätt".

o/å-övning

ENKEL MÖNSTERMATCHNING

Exempel 1:

Övning:

i stället för att förtvivla och sedan gå och tigga _m n_d hos kungen klädde han ut sig till ambulerande m_lare och reste ner till södra frankrike

Svar:

i stället för att förtvivla och sedan gå och tigga [o]m n[å]d hos kungen klädde han ut sig till ambulerande m[å]lare och reste ner till södra frankrike

Exempel 2:

Övning:

försiktig eftertänksam osäker p_ sin egen förm_ga och beroende av vuxnas stöd flera mödrar beskrev sina blyga barn så

Svar:

försiktig eftertänksam osäker p[å] sin egen förm[å]ga och beroende av vuxnas stöd flera mödrar beskrev sina blyga barn så

STOPPFILTER

Exempel 1:

Övning:

i stället för att förtvivla och sedan gå och tigga om n_d hos kungen klädde han ut sig till ambulerande m_lare och reste ner till södra frankrike

Svar:

i stället för att förtvivla och sedan gå och tigga om n[å]d hos kungen klädde han ut sig till ambulerande m[å]lare och reste ner till södra frankrike

Exempel 2:

Övning:

försiktig eftertänksam osäker på sin egen förm_ga och beroende av vuxnas stöd flera mödrar beskrev sina blyga barn så

Svar:

*försiktig eftertänksam osäker på sin egen förm[å]ga och beroende av vuxnas stöd
flera mödrar beskrev sina blyga barn så*

Enkel mönstermatching används för att generera å-ljudningsövningar på "o" och "å" med krav på omgivningen. Frekventa ord filtreras sedan bort med stoppfilter vilket gör att det inte genereras någon övning på orden "om" och "på".

å/ä/ö-övning

ENKEL MÖNSTERMATCHNING

Övning:

detta g_ller också f_r kommuner som inte ingår i n_gon landstingskommun

Svar:

detta g[ä]ller också f[ö]r kommuner som inte ingår i n[å]gon landstingskommun

STOPPFILTER

Övning:

detta gäller också för kommuner som inte ingår i n_gon landstingskommun

Svar:

detta gäller också för kommuner som inte ingår i n[å]gon landstingskommun

Med enkel mönstermatchning genereras övningar på "å", "ä", och "ö". Stoppfilter används sedan för att inte generera övningar på frekventa ord, som ordet "på". När vi utökade filtret att inte generera mer än en lucka på frekventa ord som innehåller alla tre svenska vokalerna (å/ä/ö) sker en ytterligare förbättring på ordet "ändå".

l/r-övning

ENKEL MÖNSTERMATCHNING

Exempel 1:

Övning:

*det här är den bästa affär vi kan göra för att kunna växa inte_nationellt säge_
konce_nchefen bo ramfors till tt*

Svar:

*det här är den bästa affär vi kan göra för att kunna växa inte[r]nationellt säge[r]
konce[r]nchefen bo ramfors till tt*

Exempel 2:

Övning:

*ungefär lika mycket som roge_ skul_e tjäna på de senaste husaffäre_na hade
rånaren begärt i lösen för bankpe_sona_en han tagit som giss_an samt att få rikets
mest kände brottsling c_ark olofsson ditskickad samt en flyktbil*

Svar:

ungefär lika mycket som roge[r] skul[l]e tjäna på de senaste husaffäre[r]na hade rånanaren begärt i lösen för bankpe[r]sona[l]en han tagit som giss[l]an samt att få rikets mest kände brottsling c[l]ark olofsson ditskickad samt en flyktbil

STOPPFILTER

Exempel 1:

Övning:

det här är den bästa affär vi kan göra för att kunna växa inte_nationellt säger konce_nchefen bo ramfors till tt

Svar:

det här är den bästa affär vi kan göra för att kunna växa inte[r]nationellt säger konce[r]nchefen bo ramfors till tt

Exempel 2:

Övning:

ungefär lika mycket som roge_ skulle tjäna på de senaste husaffäre_na hade rånanaren begärt i lösen för bankpersona_en han tagit som giss_an samt att få rikets mest kände brottsling c_ark olofsson ditskickad samt en flyktbil

Svar:

ungefär lika mycket som roge[r] skulle tjäna på de senaste husaffäre[r]na hade rånanaren begärt i lösen för bankpersona[l]en han tagit som giss[l]an samt att få rikets mest kände brottsling c[l]ark olofsson ditskickad samt en flyktbil

Med enkel mönstermatchning genereras övningar på alla "l" och "r" med krav på omgivningen vilket resulterar i att frekventa ord genereras samt att "r" inte alltid ljudas korrekt som i "bankpersonalen". I vissa fall kan klustret "rn" som i exemplet "husaffärerna", inte anses ha korrekt ljudning, men vi har i utvärderingen bestämt att det ska anses korrekt eftersom betoningen på "r" kan vara olika. Med stoppfilter filtreras frekventa ord innehållande "l" och "r" med specifika krav på omgivning bort, vilket vi ser med orden "säger" och "skulle" som det inte genereras någon övning på.

b/p-övning

ENKEL MÖNSTERMATCHNING

Övning:

u_sättningen _lev om jag rätt kommer ihåg den en lysande u_visning av tvilling_aret _hili_ zandén och krister henriksson men resten en mer löst ho_fogad inventering av ensamhetens tvåsamhetens och a_normitetens många as_ekter

Svar:

u[p][p]sättningen [b]lev om jag rätt kommer ihåg den en lysande u[p][p]visning av tvilling[p]aret [p]hili[p] zandén och krister henriksson men resten en mer löst ho[p]fogad inventering av ensamhetens tvåsamhetens och a[b]normitetens många as[p]jekter

STOPPFILTER

Övning:

*u_sättningen blev om jag rätt kommer ihåg den en lysande u_visning av
twilling_aret phili_zandén och krister henriksson men resten en mer löst ho_fogad
inventering av ensamhetens tvåsamhetens och a_normitetens många as_ekter*

Svar:

*u[p][p]sättningen blev om jag rätt kommer ihåg den en lysande u[p][p]visning
av twilling[p]aret phili[p] zandén och krister henriksson men resten en mer löst
ho[p]fogad inventering av ensamhetens tvåsamhetens och a[b]normitetens många
as[p]ekter*

Enkel mönstermatchning matchar och genererar övningar på alla "b" och "p". Korrekta genererade övningar är där b och p ljudas. "p" i namnet "philip" räknas som felgenererad likväl som "p" i "psykolog" som kan förekomma i andra meningar. Detta åtgärdades med en modifiering av mönstret. Stoppfilter används för att filtrera bort frekventa ord som "blev".

7 Slutsats

Vi har tillämpat och utvärderat tre språkteknologiska metoder, enkel mönstermatchning, stoppfilter och lexikonuppslagning med ordklasstaggar för att skapa ett program som genererar stavningsövningar i form av luckövningar för andraspråksinlärare. Alla övningar vi skapat når vårt kriterium på en F-score $\geq 90\%$. Övningar med ng/gn, alfabet, b/p, l/r, å/ä/ö och sje-ljud når upp till högsta F-score på 100%, och vi anser att övningarna är kompletta.

Under arbetets gång har vi sett att de tre språkteknologiska metoderna vi har använt oss av har inverkat på programmet. Enkel mönstermatchning var egentligen tillräcklig för övningarna på alfabet, ng/gn, ck/k, sje-ljud, o/å, å/ä/ö, l/r och b/p då de uppnådde kriteriet med F-score men några av övningarna genererar dock luckor på för frekventa ord, vilket inte var ett kriterium för mönstermatchning men som inte är önskvärt i företagets system. Övningarna som vi tillämpade stoppfilter på för att i bästa fall uppnå godtagbar nivå enligt F-score samt inte generera luckor på ord av fel ordklass, fel ljudning samt för frekventa ord var suffix, ck/k, e/ä, o/å, å/ä/ö, l/r och b/p. Eftersom suffixövningen har kriteriet att endast generera övningar på specifika ordklasser tillämpade vi lexikonuppslagning med ordklasstaggar för att se om mindre luckor genererades på ord av fel ordklass. Med ett större lexikon skulle vi nå fler av våra i dagsläget okända ord men problemet med disambiguering skulle kvarstå i och med att programmet inte ser till kontexten och väljer ordklass efter det första ordet som matchas i lexikonet.

Programmet skulle kunna generera övningar på mer specifika mönster för att uppnå bättre resultat. Därmed skulle nuvarande mönstren behöva utvecklas genom att utvärdera programmet på fler texter och på så sätt hitta bättre mönster. Stoppfilter behöver inte alltid ha en positiv inverkan på precisionen då det inte passar alla övningar. Stoppas många korrekt genererade ord med hänsyn till frekvens är det ibland på bekostnad av precision, detta på grund av att felen kan kvarstå.

Vi har använt lexikonuppslagning med ordklasstaggar eftersom det kändes som ett effektivare och mer avgränsat alternativ. I längden är lexikonuppslagning med ordklasstaggar inte något bra alternativ att använda för att bestämma ordklass. Att implementera en ordklasstaggar skulle lösa vårt problem med ambiguitet eftersom vi får första bästa matchning med tagg i lexikonuppslagningen. Skillnaden mellan en ordklasstaggar och lexikonuppslagning är att ordklasstaggar taggar hela texten, medan vi med lexikonuppslagning enbart matchar de relevanta orden för suffixövningen. Detta gör att programmet inte behöver normalisera taggar i en hel text utan bara de specifika orden, vilket vi tror sparar tid. Företaget har sin egen ordklasstaggar och morfologiska analysator som avgör vilken tagg som är korrekt i kontexten som vi rekommenderar för

programmet. Alltså skulle en fullständig ordklasstagning ändå vara att föredra.

Vi har anpassat programmet så att användaren kan välja ett ungefärligt totalt antal luckor på en text att generera övningar på. Detta för att möta företagets önskemål om att begränsa antalet genererade övningar. Programmet genererar inte övningar på frekventa ord, vilket inte heller företagets system gör. Eftersom företaget tillämpar ordklasstagning i sitt system har vi testat att anpassa ordklasstagning i mindre grad för en av övningarna i vårt program. Det har visat att tillämpningen ger möjlighet till förbättring av övningen. Företaget kommer dock att tillämpa egna filter som inkluderar ord som inte genererar övningar och markering av svaren med grönt eller rött beroende på om det är rätt eller fel. De kommer också att tillämpa deras egna ordklasstagare för att automatiskt generera stavningsövningarna.

Litteraturförteckning

- Abrahamsson, Niclas. *Andraspråksinläring*. Studentlitteratur, 2009.
- Ahrenberg, Lars. Grammatisk analys på dator. URL <http://www.ida.liu.se/~HKGAB9/ht06/F0re13-ht06.pdf>. Föreläsning 3, HKGAB9 Språkvetenskapliga datorlaborationer, december 2006.
- Carlberger, Johan och Kann, Viggo. Implementing an efficient part-of-speech tagger. *Software: Practice and Experience, Volume 29, Issue 9*, ss 815–832, july 1999. URL [http://onlinelibrary.wiley.com/doi/10.1002/\(SICI\)1097-024X\(19990725\)29:9%3C815::AID-SPE256%3E3.0.CO;2-F/abstract](http://onlinelibrary.wiley.com/doi/10.1002/(SICI)1097-024X(19990725)29:9%3C815::AID-SPE256%3E3.0.CO;2-F/abstract).
- Cozens, Simon. *Beginning Perl*. Wrox Press, Birmingham, 2000.
- Ejerhed, Eva, Källgren, Gunnel, Wennstedt, Ola, och Åström, Magnus. *The Linguistic Annotation System of the Stockholm-Umeå Corpus Project*. University of Umeå: Department of Linguistics. Department of Scandinavian Languages, Umeå, 1992.
- Engstrand, Olle. *Fonetikens grunder*. Studentlitteratur, Lund, 2009.
- Gimeno-Sanz, A. och Davies, G. Call software design and implementation, 2010. URL http://www.ict4lt.org/en/en_mod3-2.htm.
- Jurafsky, Daniel och Martin, James H. *Speech and language processing - An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition*. Pearson Education, 2009.
- Källgren, Gunnel. *Documentation of the Stockholm – Umeå Corpus*. In: *Manual of the Stockholm Umeå Corpus version 2.0*. Stockholm University: Department of Linguistics, Stockholm, 2006. Sofia Gustafson-Capková and Britt Hartmann (eds.).
- Lastow, Birgitta och Håkansson, Gisela. Grammatical terminology and the application gramte. Working Papers 46 (1997), 185-195, 1997.
- Levy, Mike och Stockwell, Glenn. *CALL Dimensions: Options and Issues in Computer-Assisted Language learning*. Lawrence Erlbaum Associates, Inc. Mahwah, New Jersey, 2006.
- Metcalf, Vanessa och Meurers, Detmar. Generating web-based english prepositions exercises from real-world texts. EUROCALL 2006, september 2006.

- Naber, Daniel. A rule-based style and grammar checker. Examensarbete, Technische Fakultät, Universität Bielefeld, Bielefeld, 2003.
- Öhrman, Lena. Datorstödd språkgranskning och andraspråksinlärare. Examensarbete, Institutionen för lingvistik, Stockholms Universitet, Stockholm, 2007.
- Ott, Neils och Meurers, Detmar. Authentic text icall (aticall) exercise generation and information retrieval for language learning. Presentation, workshop talk, Tübingen-Berliner Lernerkorpusstreffen. Humboldt-Universität zu Berlin. 15./16., 2009.
- Ott, Neils och Ziai, Ramon. Icall activities for gerunds vs. to-infinitives: A constraint grammar-based extension to the new werti system. URL <http://niels.drni.de/n3files/read-me/werti-gerunds.pdf>. Unpublished term paper for the course *Using Natural Language Processing to Foster Language Awareness in Second Language Learning* taught in summer 2008 at Tübingen University by Detmar Meurers, 2008.
- Saeed, John I. *Semantics*. Blackwell Publishing, United Kingdom, 2003.
- Språkrådet. Svenska som andraspråk, 2007. URL <http://www.sprakradet.se/1960>.
- Svensson, Patrik. Språkutbildningen och cyberspace. *Datorn i utbildningen*, (8), 2007. URL <http://www.diu.se/nr8-07/nr8-07.asp?artikel=s22>.
- Svensson, Patrik. *Språkutbildning i en digital värld: informationsteknik, kommunikation och lärande*. Stockholm Nordstedts Akademiska förlag, Stockholm, 2008.
- Svensson, Patrik. Samtal om språk och språkinläring, mars 2010. URL <http://www.ur.se/play/155198>.
- Vitikka, Johanna. Finns det något samband mellan andraspråksinlärares uttal och stavning i svenska? –en undersökning om hur uttal och stavning påverkar varandra. Examensarbete, Institutionen för svenska språket, Göteborgs Universitet, Göteborg, 2009.
- Warschauer, Mark. Computer assisted language learning: an introduction. *Multimedia language teaching*, 1996. URL <http://www.ict4lt.org/en/warschauer.htm>.