UNIVERSITÉ FRANÇOIS - RABELAIS

DE TOURS

ÉCOLE DOCTORALE Santé, Sciences et Technologies

Laboratoire d'Informatique
Equipe « Bases de données et Traitement des langues naturelles »

THÈSE présentée par :

Tonio Wandmacher

soutenue le : **22 octobre 2008**

pour obtenir le grade de : **Docteur de l'université François - Rabelais**

Discipline/ Spécialité : Informatique

Titre

Adaptive word prediction and its application in an assistive communication system

Adaptive word prediction and its application in an assistive communication system

von
TONIO WANDMACHER

Philosophische Dissertation
angenommen von der Neuphilologischen Fakultät
der Universität Tübingen
am 22. Oktober 2008

Osnabrück

2008

Gedruckt mit Genehmigung der Neuphilologischen Fakultät
der Universität Tübingen

Summary

The present work investigates the capacities of adaptive methods for word prediction, which represents a central strategy in the context of alternative and augmentative communication (AAC) for speech- and motion-impaired persons. First an introduction to the respective research fields of AAC and word prediction is given. Here we address in particular stochastic language models, which have been shown to be very appropriate for the task of prediction. We then explore two major classes of adaptive methods: In a first part we consider strategies enabling to adapt to the lexical and syntactic preferences of the user of an AAC system. Here we investigate the recency promotion or cache model, an auto-adaptive user lexicon and the dynamic user model (DUM), which integrates every input of the user.

The second class of methods aims to adapt to the semantic or topical context. Here we focus in particular on *Latent Semantic Analysis* (LSA), a vectorial model establishing semantic similarity from distributional properties of lexical units in large corpora. A difficult aspect however arises from the integration of LSA-based information to the general prediction model; for this reason we discuss and evaluate here several integration methods.

In the last part of this work an assistive communication system is presented that implements the most successful of the previously investigated adaptation methods. After a description of the user interface as well as the communication enhancing components we report results from the application of this system in a rehabilitation center, where it has been in use for several years.

Zusammenfassung

Die vorliegende Arbeit untersucht die Kapazitäten adaptiver Methoden für die Wortprädiktion, einer zentralen Strategie im Kontext der unterstützenden Kommunikation (UK) für sprach- und bewegungsbehinderte Menschen. Zunächst wird eine allgemeine Einführung in die Forschungsgebiete der UK einerseits und der Wortprädiktion andererseits gegeben. Dabei wird insbesondere Bezug genommen auf stochastische Sprachmodelle, welche sich für die Prädiktion als sehr geeignet erwiesen haben. Die Untersuchung adaptiver Methoden erfolgt daraufhin entlang zweier Achsen: In einem ersten Abschnitt werden die Verfahren betrachtet, welche die Adaption an die spezifische Lexik und Syntax des Benutzers eines UK-Systems erlauben. Dabei werden insbesondere das sogenannte *Cache*-Modell, ein selbstlernendes Benutzerlexikon sowie das *dynamische Benutzermodell* (*dynamic user model, DUM*) diskutiert.

Auf der zweiten Achse werden nun Modelle untersucht, welche auf eine Adaption der Wortprädiktion an den semantischen bzw. topikalischen Kontext abzielen. Hier rückt vor allem eine Methode in den Mittelpunkt: die Latent-Semantische Analyse (LSA), ein vektorraumbasiertes Verfahren, welches semantische Ähnlichkeit durch distributionelle Eigenschaften lexikalischer Einheiten etabliert. Als schwierig erweist sich jedoch die Integration der LSA-basierten semantischen Information in das allgemeine Prädiktionsmodell, weshalb hier verschiedene Integrationsmöglichkeiten untersucht werden.

Im letzten Teil dieser Arbeit wird ein UK-System vorgestellt, welches die erfolgreichsten der zuvor betrachteten Adaptionsverfahren implementiert. Nach einer Präsentation der verschiedenen kommunikationsfördernden Komponenten wird von der Anwendung dieses Systems berichtet, welches seit mehreren Jahren in einem Rehabilitationszentrum eingesetzt wird.

Résumé

Ce mémoire étudie les capacités de méthodes d'adaptation pour la prédiction de mots, une stratégie importante dans le contexte de l'aide à la communication pour personnes handicapées. Notre première partie présente le domaine de la communication assistée et de la prédiction de mots. Dans ce cadre, nous avons abordé en particulier les modèles de langage stochastiques, qui se sont avérés très appropriés dans le contexte de la prédiction. Ensuite, nous avons étudié deux groupes d'approches adaptatives. Le premier groupe de méthodes traite de l'adaptation aux préférences lexicales et syntaxiques de l'utilisateur d'un système de communication assistée. Au sein de ce groupe de méthodes, nous avons étudié le modèle cache, le lexique auto-adaptatif et le modèle d'utilisateur dynamique (MUD), intégrant toute saisie de l'utilisateur.

Le deuxième groupe de méthodes rassemble des approches qui ont pour objectif d'exploiter le contexte sémantique. Dans ce contexte, nous avons en particulier étudié l'analyse sémantique latente (ASL), un modèle vectoriel qui se base sur les propriétés distributionnelles d'unités lexicales. Cependant, l'intégration de l'information sémantique dans le modèle de prédiction général présente un aspect difficile. Pour cette raison, nous avons développé et étudié plusieurs possibilités d'intégration.

Dans la dernière partie du mémoire, nous présentons un système d'aide à la communication, dans lequel nous avons implémenté les méthodes d'adaptation les plus prometteuses. Après une description de l'interface utilisateur ainsi que des composants de communication améliorée, nous avons exposé quelques expériences réalisées avec ce système, qui est utilisé depuis plusieurs années dans un centre de rééducation fonctionnelle.

Preface

Interestingly, it was an Englishman who pronounced the following words, one year after the Second World War had ended: *"The first step in the re-creation of the European Family must be a partnership between France and Germany.*[1]*"* This Englishman – Winston Churchill – was convinced that the ceaseless nationalistic rivalries on this continent could only be overcome by creating a supranational structure. And his endeavors were rewarded; 60 years later, most Europeans use the same currency, live and work wherever they want, they travel without border-controls, and they take all this for granted.

In this development the German-French partnership has indeed attained particular importance. The political, economic and cultural bonds between the former *archenemies* have become exceptionally strong; France and Germany are nowadays often referred to as the *heart* or *engine* of the European Union. This is however not a matter of course, it results from a multitude of political efforts, like the Elysée friendship treaty from 1963, but also from setting up numerous exchange programs between youth of both countries. The work of organizations like the *Franco-German Youth Office* (DFJW/OFAJ), the *French-German University* (DFH/UFA) and many other municipal and academic initiatives have notably consolidated these efforts.

The present thesis is likewise a product of this integration process: It has been prepared in a so-called *cotutelle* framework, implying a co-supervision by tutors from both countries. I had not only the chance to have two supervisors but also to work in both countries, to become acquainted with two different academic systems and research traditions, as with two different ways of living. Establishing this binational project was somehow a metaphor on the overall process: It was difficult and tedious. But in the end I can say that it was more than worthwhile.

[1]Winston Churchill: *Speech to the academic youth*, Zürich, 19 September 1946.

Acknowledgements

I acknowledge that this thesis would never have been possible without the organizational and intellectual, the financial, technical and emotional support of many. In the first place I want to thank my two supervisors, Jean-Yves Antoine and Uwe Mönnich, who contributed a lot to my work in each of these aspects. Not only in the beginning of this project, many bureaucratic obstacles had to be overcome, proposals and reports had to be written, technical problems to be solved, the direction of my work had to be traced, and everything worked out perfectly. Considering the question whether I could have been supervised in any better way my answer is no.

In this respect I also want to mention Karin Krüger-Thielmann, who is responsible for coming up with the initial idea for this project. I will not forget the moment when she asked me – en passant – in March 2004 if I could imagine going to France ... But she did not only have this wonderful idea, she also helped a lot in making it come true.

This leads me to financial aspects: I am indebted to the German Academic Exchange service (DAAD) for according me a graduate scholarship for the time of my stay in France. Back in Germany I started to work at the Institute of Cognitive Science, Osnabrück in a research project funded by the German Research Council (DFG). And for all the journeys between the countries I received a travel grant from the French-German University (DFH/UFA).

Speaking of technics I am very grateful to Pierre-François Laurand at the IUP in Blois as well as to the system administrators in Osnabrück (in particular to Martin Schmidt, Egon Stemle and Udo Wächter) who all helped me a lot to handle and process my masses of data and resolve programming problems.

The application of the work developed would not have been possible without the help of Jean-Paul Départe as well as of the speech therapists at the rehabilitation center of Kerpape. Several valuable features of our system are a direct result from discussions with them. And in this respect I want to thank especially Margaux and Quentin for opening my eyes. They showed me how much one can communicate without words.

From an emotional perspective I want to thank all my co-thésards and colleagues at the IUP in Blois who warmly welcomed me and provided me a pleasant and joyful atmosphere; the same applies for my colleagues in Osnabrück. Here I have to mention in particular my boss, Kai-Uwe Kühnberger, who always left me enough freedom to pursue my PhD-related work. Special thanks also to Stefan Evert, Katja Ovchinnikova, Gabriel Pickard, Konstantin Todorov and Daniel Weiller for valuable comments, hints and corrections.

And finally – even though personal concerns might not belong into research – I have to express my particular gratitude to my parents and to Anne-Lise, who supported, who encouraged and put up with me not only over the past four years.

I hope to have made clear that many people contributed to this work; so, whenever the first person plural will be used in the following, I want to refer to each of these persons. They are excluded however with respect to all errors, imprecisions and other aspects that the reader might criticize.

Osnabrück, June 2008

Contents

Notation

X	Random variable
X_1^n	Variable sequence of length n
x	Any event or outcome of X (e. g. occurrence of a symbol)
$p(x)$	Probability of an event x
$P(x, m)$	Probability estimate for an event x with respect to a model m (normally m is omitted)
$C(x)$	Occurrence frequency of x in a given data set (corpus)
$P^*(x)$; $C^*(x)$	Adapted or smoothed probability estimate; frequency for an event x
$H(X)$	Entropy of variable X (cf. section 3.1)
$PP(X_1^n, m)$	Perplexity for a sequence of length n with respect to a model m (cf. section 3.1)
w_i	Any word of the given vocabulary
$\vec{w_i}$	A vector representing word w_i in a semantic space
c_i	Any word class or part of speech (of a given tag set)
h_i	Current history or context (i. e. all symbols to the left from the current prediction position)
V; $\|V\|$	Vocabulary; vocabulary size
α, β	Constant scaling factor or backoff weight
λ	Interpolation coefficient
ν	Normalization function
θ	Threshold value
ksr_n	*Keystroke saving rate* with respect to a list of n words (cf. section 2.5.2, eq. 2.5)
ASD	Average scanning distance (cf. section 2.5.1)

Chapter 1

Preliminaries

denn das Menschlichste, das wir haben,
ist doch die Sprache

THEODOR FONTANE
(*Unwiederbringlich*, 1891)

1.1 Introduction

This thesis connects two research areas which are – at first glance – rather distinct: On the one hand we discuss issues of *word prediction*, which represents a more or less classical problem in *natural language processing* (NLP); on the other hand we deal with questions of *alternative and augmentative communication* (AAC), a subfield of disability research that aims to enhance the communicative facilities of speech- and motion-impaired persons.

The aim of NLP as a discipline is to create computer systems that are able to perform human language tasks in a robust, efficient and computationally tractable way. In this regard it has to be considered a branch of artificial intelligence, and it should be distinguished from computational linguistics (CL), which Crystal (1991) defines as *"a branch of linguistics in which computational techniques and concepts are applied to the elucidation of linguistic and phonetic problems"*. So, whereas CL aims to study human language using computational tools, NLP develops computational tools that process human language, without making rigorous claims on linguistic or cognitive adequacy.[1]

Typical NLP tasks are speech recognition, machine translation or optical character recognition. In such tasks an incomplete, mutilated or simply altered input signal is to be transformed into a linguistically well-formed output

[1]But as Cunningham (1999) points out, *"there is crossover and blurring of these definitions in practice"*.

signal, i.e. given all information provided by the input, the most likely output has to be predicted. The major achievements in the past thirty years of NLP basically concern this input-output mapping and the way in which the input can be analyzed in an optimal manner. As stated above, optimality in NLP is to be understood in terms of robustness, efficiency and computational complexity, not in a linguistic sense. Yet it can be claimed that a large range of sophisticated models and approaches has been developed in this area, at every level of linguistic description. And in many of these models prediction of characters, phonemes, morphemes, words or even larger constituents is involved.

Research in NLP has yielded fruit: While far from perfect, large-scale machine translation systems nowadays provide translated manual pages for software companies and internal reports for political institutions (e.g. the European Commission), speech recognizers and dialog systems inform passengers about train connections, grammar and spell checkers are built into almost every text processor.

Research in AAC on the other hand "*involves attempts to study temporary or permanent impairments, activity limitations, and participation restrictions of individuals with severe disorders of language production or comprehension, including spoken and written modes of communication*" (cf. ASHA, 2002). Persons affected by neuro-muscular disorders or cerebro-vascular accidents can be unable to control their speech organs or to communicate otherwise (e.g. by using sign language); they are dependent on an external aid – be it human or artificial – in order to be able to communicate at least partly.

This kind of disability is not as rare as one might think: According to the *American Speech-Language-Hearing Association* (ASHA) approximately two million Americans are unable to use speech and/or handwriting to meet their daily communication needs, representing between 0.8% and 1.2% of the U.S. population (ASHA, 2002). Similar numbers are reported for the UK: Newell *et al.* (1998) estimate that over one million people in the UK have severe language impairment, and more than 300.000 cannot communicate using speech. Such counts suggest a considerable social importance for research in this field, and this importance is growing, due to the demographic trend in many developed countries: With the aging of the population, the number of people in need of assistance and assistive technology will increase significantly (cf. European Commission, 2007)[2].

Within AAC research the development of electronic communication aids takes a prominent position, for two main reasons: the loss of communicative abilities, being unable to express one's thoughts and to interact with others is normally perceived as the most severe kind of impairment, leading to social exclusion and to complete dependence on speech-abled persons. Therefore every means enabling to reduce the dependence of such persons is of eminent importance. Moreover, the development of computer-based communication

[2](European Commission, 2007), p. 2: "[...], *the number of older persons in need of care will more than double by 2050, increasing the demand for formal care services.*"

technologies increases the need for accessibility to these new media, especially by disabled persons who are in particular isolated from other forms of interaction.

The main objective of assistive communication systems is to enhance the communicative abilities of their users. To achieve this, several factors have to be considered in their development. Newell *et al.* (1998) list the following:

1. reduction of the physical input necessary to produce an utterance
2. reduction of the cognitive load on the user
3. increase in the speed of communication
4. reduction in delay between the user formulating what they want to say and the device articulating the appropriate words.

In many AAC systems these problems are addressed by two complementary strategies: the first is to offer an efficient item selection procedure which can be controlled by a binary command. Here, each of the items (characters/words/icons/phrases) is successively activated and can then be selected by the user (a method normally referred to as *scanning*). The other strategy tries to predict the element that the user intends to insert. If the wanted element is guessed correctly, the user can simply confirm the prediction and thereby saves effort and time. The problem of prediction can again be traced back to the input-output mapping problem: How can the input (number of items to insert) be reduced as far as possible, so that the output, as intended by the user, can still be generated?

This is the point at which the two disciplines meet: In NLP a class of models has been developed that is well-suited to handle the mapping problem: Stochastic language models (SLM) encode probabilistic distributions of language symbols and they are able to determine the most probable symbols to follow a given context (at least to some extent). Computationally these models are equivalent to stochastic finite state automata, which means that they can only reflect linear dependencies, moreover the context they exploit is limited to a few words (usually two or three). Still, they have some very attractive properties: they are very robust, i.e. they will always return a result, and the computational effort, compared to other, more structural models, is very low. In addition, despite their inability to describe complex linguistic structures, they seem to capture already a large deal of regularities in human language. In any case SLM are nowadays successfully applied in a wide range of NLP tasks; for example most large-scale speech recognizers incorporate knowledge from an SLM at some point.

As stated above, these models calculate probabilities for a given symbol after a given context. However, it is a truism of linguistics that a word or a phrase to be uttered does not simply depend on the n previous words. Apart from the information to be transmitted (which is of course unpredictable) humans choose their words based on a large range of situational factors: the current style and topic of the conversation, the interlocutor, the purpose of the message, their sex, social status and age, and, as we deal with human beings, they might simply have personal preferences for certain words or syntactic constructions that are beyond any rational explanations. As a result the lexical, syntactic and semantic characteristics of the generated messages vary considerably. But these factors are not arbitrary; during message composition people make extensive use of this situational knowledge. A good example is *alignment*: interlocutors tend to align their ways of expression: after a few sentences only they start using the same vocabulary and syntactic structures (Pickering & Garrod, 2004; Branigan & Pearson, 2006). This phenomenon happens unconsciously and without any explicit negotiation; people adapt their way of communication to each other.

Statistical language models in the above described form are unable to adapt. Their probabilistic properties are fixed, once they have been estimated on a training corpus. This means that they cannot make use of a vital part of human language, which implies a considerable loss of performance. But especially in the context of AAC, intelligent, adaptive models are needed in order to support as far as possible the communication process of a person who is unable to express her- or himself otherwise. The investigation of such models will be the central topic of this work.

1.2 Research questions and objectives

It has long been known that adaptation is a powerful strategy to improve a system's performance in changing environments. Adaptive approaches can nowadays be found in various domains, from economics via cybernetics to many problems in artificial intelligence (for an overview cf. Holland, 1992). Therefore, even though this alone might not be a sufficient argument, it is consistent with the current scientific development to investigate the use of adaptive techniques in AAC.

We intend to study two families of adaptation strategies: One focuses on the lexical and syntactic preferences of the user; here we will investigate three major models: (i) the *cache model*, (ii) the *user lexicon*, and (iii) the *dynamic user model*, a language model which dynamically integrates all user input. The other family of adaptation methods exploits the semantic context of the message to be typed. Semantic knowledge plays an important role for the prediction of content words (e.g. consider the probability of *merger* to occur in a stock trading context versus a loose conversation about football). However this kind of knowledge is obviously rather elusive and difficult to formalize. In this context we want to make use of *Latent Semantic Analysis* (LSA) a vectorial technique based on distributional properties of words which has been successfully employed in a variety of very different tasks, including information retrieval, cognitive psychology (lexical acquisition) and automated summarization. However, as the problem of integrating information from LSA with a standard language model turns out to be non-trivial, we will propose several strategies how this integration can be achieved.

All these different methods will be implemented and evaluated in detail with respect to their adaptive and predictive capacities. For the assessment we will apply *cross-register* evaluation, i.e. we will compare the results of several test corpora from distinct registers (or domains) in order to get a more reliable picture of the performance variance in general and the adaptivity of the respective model in particular. Furthermore, to assure at least some cross-linguistic comparability the assessment will be performed on three languages: French, German and English.

Apart from the (rather theoretic) investigation of adaptive models, the second main objective of this thesis concerns the integration of such models into an assistive communication system. It could be observed for many years in the domain of AAC as well as in other subfields of artificial intelligence that only small-scale, showcase applications were developed in order to prove the feasibility of the approach. However, too often such applications included what Cunningham (1999) calls a "*toy problem syndrome*": As soon as the approach is scaled up to a real-world task, it turns out to be inappropriate, due to problems of complexity or usability (which is more often the case). As a result a lot of probably thorough and long-lasting work is futile.

In order to avoid such a mismatch with the real world we aim to develop a system which is not only applicable but applied. Fortunately, we do not have to start from scratch here, we can base our development on the SIBYLLE system, an assistive communication system, which has evolved from the work of Igor Schadle, who obtained his PhD from the *Université Bretagne-Sud* (France) in 2003. At that time Igor also had established contact with the rehabilitation center of Kerpape (Morbihan, Brittany), where SIBYLLE has been applied with communication-impaired users since. The unique value of a direct contact with these users cannot be overestimated for the development of such a system.

A third major aspect of our work concerns questions of language portability. As mentioned above, we assess our models on three languages; however we also want to enable our system to work with these languages. This implies on the one hand that we develop a generic and modular architecture, but on the other hand it requires addressing language-specific peculiarities such as productive compounding in German.

Our final, overall objective is to create a well-performing, plurilingual AAC system that makes use of the adaptive strategies investigated and that is able to palliate the communicative impairment of its users as far as possible.

1.3 Thesis organization

At times the enforced linearity of a textual document can be impedimental. This is particularly true in the case of a thesis operating from two rather distinct points of view. For this reason we have to ask the reader to consider both the chapters 2 and 3, being introductions to their respective domain, as being on par with regard to the text structure.

Chapter 2 presents the research field of *Alternative and Augmentative Communication*. Here we discuss the major challenges and objectives of this discipline, give a short outline of its development and introduce some of the prevailing research methods. Furthermore, an overview of the most influential systems and projects in AAC will be given, and in the end the evaluation methodology of such approaches is discussed in detail.

Chapter 3 on the other hand gives an introduction to the theory of communication in general and to approaches of language modeling in particular. We outline the problem of prediction from an information-theoretic perspective and introduce the basic notions of Markov processes and stochastic (n-gram) language models. In a longer section we then discuss the problem of data sparsity and present methods to estimate the probability of unseen events (so-called *smoothing* models). Afterwards some more practical issues like parameter estimation or storage formats for language models are depicted, and in the end we present two other families of language models: those based on linguistic considerations (e.g. *chunk parsers*) and maximum entropy models, a promising approach that enables to combine all types of information in one model.

After having presented the state of the art in the relevant domains, the chapters 4 and 5 will be devoted to the problem of adaptation in language modeling. Chapter 4 focuses on methods enabling a model to adapt to the user's lexical and syntactic preferences. We here discuss problems of training dependency and adaptation in language modeling, and we present in detail the three already mentioned adaptation techniques. We then introduce our evaluation paradigm as well as the training and test corpora we have made use of. In the end we present results of a detailed evaluation of all adaptation methods.

Chapter 5 then deals with the other family of adaptation models, namely those that enable to exploit the current semantic or topical context. After motivating the use of semantic information in our task we will present several models that enable to capture such information from the context and exploit it for prediction. We will then concentrate on *Latent Semantic Analysis*, a method that has been shown to provide reliable information on long-distance semantic dependencies between words in a context. After a detailed presentation of the approach we introduce several methods enabling to integrate the semantic information captured by LSA into a prediction model. As before the chapter ends with an in-depth quantitative evaluation of the predictive capacities of a language model adapted by information from LSA.

The last major chapter (6) is finally devoted to the presentation of the AAC system that has been developed in the context of this thesis: SIBYLLE. Its latest version incorporates the adaptive word prediction models that have been presented in the previous chapters as well as a number of other features that have proven useful for AAC users. We will present the system as it is, explain the major functionality of the user interface and the key selection strategies included, and we disclose some details of how the system was implemented. After an evaluation of the different key selection techniques and the word predictor, using the integrated adaptation models in combination, we close this chapter by reporting findings from the application of SIBYLLE in the Kerpape rehabilitation center. In the final chapter we try to make a conclusion of the results achieved and give an outlook on further research efforts that we deem reasonable within the context of this thesis.

Chapter 2

Alternative and Augmentative Communication

where words are scarse,
they are seldome spent in vaine

WILLIAM SHAKESPEARE
(*Richard II*, 1597)

This chapter provides an introduction into the research field of *Alternative and Augmentative Communication*. First, the main objectives and research questions of this young discipline are presented (2.1), followed by a historic outline of its development (2.2). We will then present the most important strategies (2.3), and we will give an overview of the most influential projects and developments in this area (2.4). In the end of the chapter, we will discuss the question of how a communication system can be evaluated (2.5), which is a fundamental concern for the following chapters.

2.1 Aims and challenges of AAC

In his *Politics*, Aristotle described two elementary properties of man: Being a *'zoon logon echon'*, an intelligent animal, possessing the gift of language, and being a *'zoon politikon'*, which is commonly translated as 'social animal'.[1] Social interaction is one of the most fundamental and essential human activities, and it is based on communication. Being deprived of one's communicative abilities is normally considered to be the most severe kind of impairment; many legal systems know *solitary confinement* as the most rigorous form of punishment, and psychological studies have shown that prisoners being deprived of

[1]cf. Aristotle: *Politics - A treatise on government*, translated by William Ellis, published by J. M. Dent & Sons Ltd, London, Toronto, 4th reprint, 1928.

the ability to communicate for a longer time often develop mental or psychosomatic disorders.

Loosing one's communication facilities however can also be due to a number of severe medical conditions, affecting not only the speech organs but also the limbs, so that alternative communication channels (e.g. by sign language) are not accessible either. This can be caused by a range of neuro-muscular diseases, by cerebro-vascular accidents or by brain lesions. In the following we will give a short overview of the different conditions that can lead to such a state:

Cerebral palsy is a non-progressive disease acquired before birth or in early childhood, leading to various kinds of motoric dysfunctions. It can result in a variety of clinical patterns: diplegia, tetraplegia, athetosis or dyskinesia, sometimes combined with cognitive disability, which can also include aphasia. The incidence of this condition in the industrial countries is approximately 0.6 - 2 cases per 1000 births.

Amyotrophic lateral sclerosis (*ALS*, also called *Maladie de Charcot*) causes a progressive degeneration of the motoric nervous system including the peripheral nerves and the respiratory system. ALS usually strikes people between 40 and 60 years of age, men are affected more often than women. It gradually leads to complete loss of muscular functions, while cognitive abilities remain unaffected. The disease is quite rare; it is estimated that 1 to 2 persons per 100,000 become affected every year.

Myopathy, in particular *Duchenne dystrophy* is a hereditary disease leading to degeneration of muscular tissue, while the nervous system and cognitive functions remain intact. Its onset is in early childhood, patients have a life expectancy of less than 30 years, death is usually caused by respiratory weakness. The incidence of Duchenne dystrophy is 1 case per 3500 births.

Another reason for such an impairment may be cerebral lesions caused by apoplexia or head traumatisms (from an accident), leading to a disconnection between the cerebrum and the brain stem. This includes the loss of nearly all voluntary muscular functions, whereas the higher cognitive capacities may not be affected at all. A person suffering from such a lesion may be aware and awake while she or he is not able to move, feeling locked within her or his own body. For this reason such a state is referred to as the *Locked-In Syndrome* (LIS).

Whether suffering from a disease or a brain lesion, all these persons have in common that they have lost the ability to communicate with their environment while their cognitive facilities remain (mostly) unaffected. Treatment of the above described conditions is very limited or impossible, these persons therefore have to adjust their life with this severe handicap. Apart from all other implications of such a condition, finding some means of communicating is of first and foremost importance, and this is the research ground of *Alternative and Augmentative Communication* (AAC).

The probably oldest and most basic AAC device consists of a *communica-*

tion board, i.e. a table of icons, letters or words printed on a card board from which the person can successively select the intended items by either pointing or letting the interlocutor point to them, depending on her or him to compose and interpret the full message. Figure 2.1 shows such a non-electronic communication board.

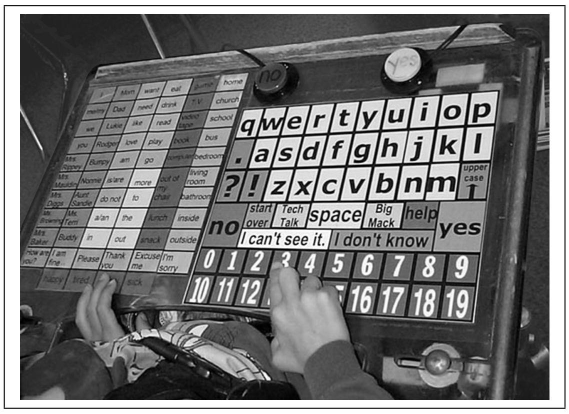


Figure 2.1: Example of a communication board

Electronic devices can emulate the same kind of functionality, but the task of pointing and composing can then be performed by the system, which implies a larger extent of independence. Nevertheless, the communication rate of these devices remains extremely reduced. Whereas people produce up to 200 words per minute in oral communication, a person using an AAC device cannot express more than a few words (see Table 2.1, cf. Le Pévédic, 1997; Newell *et al.*, 1998). This slowness of message input does not only imply loss of time, it also has a substantial impact on how the person can interact with others. In colloquial communication, long delays are rarely accepted. This leads to an asymmetric form of communication and does not allow the AAC user to control the course of the discussion and to express what she or he intended to say. Communicating at a rate of less than 10 words per minute means that many goals of spoken conversation, such as gabbling, swearing, telling a story or simply making a joke cannot be achieved anymore.

Communication mode	Speed (words per minute)
Oral communication	150 – 200
Skilled typing (10 fingers)	40 – 100
Handwriting	12 – 25
Unskilled typing (one finger)	10 – 30
AAC device	< 10

Table 2.1: Communication rates for different modes

Another limiting factor in AAC is the selection process itself. Selecting each symbol separately from a list implies great effort and is therefore rapidly tiring. Whereas unimpaired people are able to speak for hours with no sign of fatigue, a speech-deprived person may feel exhausted after entering only a few sentences.

These two aspects, *communication rate* and *cognitive effort*, span the dimensions along which AAC research has to be oriented (and evaluated): How far can an AAC device enhance the communicative abilities of its user? How "fluent" is the person when communicating with the device? And how much cognitive effort has to be implied? How usable is the device in the adapted environment of the handicapped person?

Regarding the standard architecture of AAC systems, we can distinguish four basic components: First of all there is a physical input device, which is adapted to the remaining capacities of its user (see Figure 2.2). If the person is still able to control slight arm movements she or he can profit from the larger degree of freedom offered by an adapted trackball. When her or his abilities are further restrained, a single-switch device has to be used. Since the functionality of the muscles controlling the eye-glimpse or the respiratory apparatus is usually least affected, eye or breath sensors are often applied. For patients suffering from cerebral palsy, head movements are normally best controlled, therefore these users often use sensors triggered by head movements. In the (very rare) cases of a complete loss of muscular control, the application of a *brain-computer interface* (e.g. *thought-translation device* (TTS), cf. Birbaumer *et al.*, 2000) exploiting cortical activity (i.e. slow cortical potentials) may become necessary.

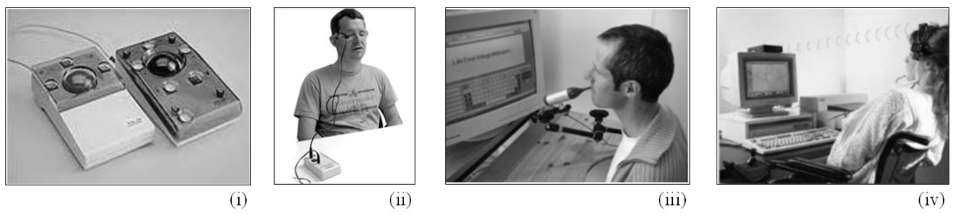


Figure 2.2: Input devices for an AAC system: (i) gross-motor trackball, (ii) eye glimpse sensor, (iii) breath sensor, (iv) head movement sensor

The second standard component of an AAC system is a software interface offering functions to insert messages. Depending on the capacities and preferences of the user it displays a table of characters, phonemes or icons and it allows the user to select an item through an iterative scanning technique: Each key is successively activated and can then be selected via the user's input device (i.e. the communication channel is reduced to *binary* or *yes/no* input).

The last two components are an editor which assembles the selected symbols and – if the AAC system is also used for oral communication purposes – a speech synthesis module, allowing to read out the composed message.

2.2 Historic development

Considering the development of AAC it is important to distinguish between icon-based and text-oriented systems. Iconographic communication is a field that has developed almost separately, and complex theories and models were developed here. However, since the focus of this work is on text-based AAC, we will present iconographic communication systems only briefly in the beginning and then explain the development of AAC relying on (alphabetic) text.

2.2.1 Iconographic AAC

Some persons are unable to communicate using an alphabetic system, either due to mental impairment (e.g. aphasia) or because they have not (yet) learned to read and to write. As an alternative to text-based communication a number of iconic or visual languages were designed, of which we will present three:

The most prominent one is the *Bliss* symbol system (Blissymbolics)[2] (cf. (Bliss, 1965). *Blissymbolics* was developed by Charles K. Bliss in the 1940s, and it was at first intended to remove language barriers and problems of ambiguity. Today *Bliss* is one of the most widespread visual languages, even some websites offer a bliss-based access. The system of *Blissymbolics* is composed of over 3000 symbols, which can be combined in order to create new symbols (see Figure 2.3, left). The language has a grammar which allows for sentences not only in the present tense, but past and future tense as well. It also contains markers for questions, commands, plurality, and possession.

Another iconographic communication system is *Minspeak*®[3], which was developed in the beginning of the 1980s (Baker, 1982). It is a pictorial system that allows to express various meanings by combination of a small set of pictures that each bear multiple meanings (also referred to as "semantic compaction"). A message can be composed by selecting and arranging sets of icons.

Finally, *Makaton*[4] is based on a vocabulary of manual signs and gestures as well as of graphic symbols. It was developed in the mid 1970s in the UK for communication with residents of a large hospital who were both deaf and intellectually disabled. The language comprises a core vocabulary of app. 450 central concepts and an open-domain vocabulary of over 7000 concepts, represented as pictograms (example sentence in Figure 2.3, right).

[2]http://www.blissymbolics.org/
[3]http://www.minspeak.com/
[4]http://www.makaton.org

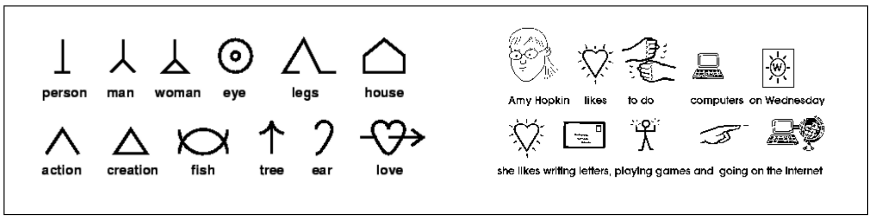

person
man
woman
eye
legs
house
action
creation
fish
tree
ear
love
Amy Hopkin likes to do computers on Wednesday
she likes writing letters, playing games and going on the Internet

Figure 2.3: Examples of iconic communication systems: *Bliss* symbols (left) and *Makaton* pictograms (right)

At first these communication systems were used without electronic devices, i.e. by pointing to the respective icons on a cardboard. Nowadays however many electronic systems exist, which are based on one of the above described languages, offering selection, browsing and editing facilities as well as speech synthesis of the composed message. For example, *Minspeak*® based systems[5] are available for many operating systems and kinds of computers (workstations, handhelds); *Bliss* fonts can be installed and used on any computer, so that *Bliss* messages can be composed with a normal editor.

2.2.2 Text-based AAC

The first and simplest AAC devices for speech-impaired but literate persons were cardboards onto which the alphabetic letters and possibly some frequent words or phrases were printed (cf. Figure 2.1). While these cardboards are simple and straightforward to use, they also have some serious disadvantages: The interlocutor has to be able to observe the selection process and to compose the message from the selected symbols; when the disabled person is not able to point to the cardboard, the interlocutor has to read out the symbols or words until the wanted element is reached, which the disabled person then confirms by some visible reaction (i.e. an eye blink or any other movement).[6] As mentioned earlier on, this way of communication is not only painfully slow and tedious, it also means that the disabled person is completely dependent on the interlocutor.

The first research efforts in AAC therefore concentrated on developing electronic devices that could replace the letter board and to reduce the dependency on a human partner by providing automated symbol scanning and composition techniques. One of the first examples of such a device was the *Tufts Interactive Communicator* (Foulds, 1973), comprising a letter display from

[5]Developed by the *Prentke Romich Company*, http://www.prentrom.com/

[6]The book "*Le scaphandre et le papillon*" (The diving bell and the butterfly) by Jean-Dominique Bauby (1997) was entirely written in this fashion. Bauby, former chief editor of the fashion magazine *Elle*, suffered from a *Locked-In Syndrome* after a cerebro-vascular accident and was only able to control the blink of his left eye.

which each letter could be selected by a line-column scanning process (cf. Figure 2.4). As most other computer systems in that time the device consisted of a rather large and clumsy box which was not easy to handle by a handicapped person.

The *Talking Brooch* (Newell, 1974) was the first portable communication device for speech-impaired persons. It allowed the user to type characters or words on a miniature keyboard which was connected to an LED display, pinned to the chest of the person.

Soon however, researchers as well as practitioners recognized that computing facilities can be of more help than simply replacing the letter board. In the beginning of the 1980s the first text-to-speech systems were developed to provide non-speaking persons with vocal facilities (cf. Hunnicutt, 1981). Moreover, some approaches were presented to accelerate message composition by anticipating the text to be entered (cf. Heckathorne & Childress, 1983; Hunnicutt, 1986).

2.3 Strategies to enhance communication

AAC research mostly concerns the design and the functionality of the interface component. Most approaches to enhance communicative facilities of the user can be traced back to one of two strategies: Either enhancing the selection rate or reducing the number of selections necessary to enter a message. The former mostly tries to optimize the arrangement of the keys on the interface as well as the selection mode, the latter aims to reduce the selection effort either by constraining expressivity (i.e. limiting the number of possible messages) or by anticipating redundant symbols.

2.3.1 Optimizing key selection

The first question that arises when we try to optimize the selection process on a keyboard is the degree of freedom a user possesses. If a person is able to control a mouse or a trackball, the problem is substantially different than if she or he can only control a binary input device. In the following we will concentrate on the latter case.

If a person is only able to control a single switch (e.g. an eye or a breath sensor, cf. 2.2) the following selection methods are possible:

- **linear**: Every item on the keyboard is highlighted one after the other. One selection step is needed.
- **line-column**: The keyboard is scanned line-wise until a line is selected, then column-wise (cf. Figure 2.4). Two selection steps are needed.

- **block-wise**: The keyboard is divided into a number of blocks of keys, which can recursively contain sub-blocks. A key is chosen by selecting the corresponding blocks in a tree-like fashion until the wanted item is reached; the number of selection steps depends on the depth of the block arrangement.

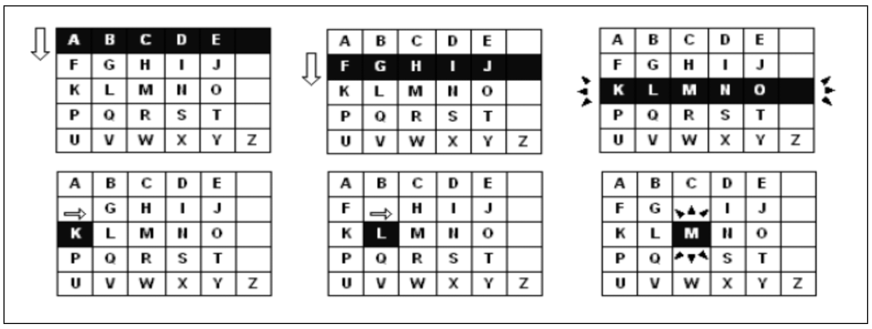


Figure 2.4: Line-column scanning on an alphabetic keyboard

Another important dimension for optimizing the selection process is the arrangement of the keys.[7] It is well-known that the arrangement of a standard QUERTY keyboard is far from optimal; many more optimal solutions have been developed in the mean time (e.g. the *Dvorak* keyboard, cf. Dvorak *et al.*, 1936). But even though these arrangements would allow faster typing, less fatigue, and less strain injuries, they were not accepted on the large scale, because human beings are obviously very prone to stick to familiar environments.

Likewise we often find in AAC systems sub-optimal keyboards. While most of the time an arrangement according to letter (or phoneme) frequency would be optimal for linear scanning (i.e. the most probable symbols are scanned first), alphabetic or QUERTY settings are also found.

Static frequency ordering is however not the only strategy to accelerate the selection process. Since we deal with a virtual keyboard, the arrangement of the keys does not have to remain fixed. It can be dynamically rearranged after every selection step, so that the most likely characters (or phonemes) for a given context appear first on the keyboard. The occurrence of *'x'* for example is much more probable after *'e'* than after *'t'* or *'j'*. In order to achieve an optimal rearrangement, a letter prediction component has to analyze the current context (i.e. the last typed characters) and to determine the probability of each character to follow. An example for such an approach is the work of Schadle *et al.* (2001), also explained in chapter 6 (6.1.2).

[7]Since the work presented here focuses on a text-based AAC system, only letter keyboards are considered; however most of the design decisions can be transferred to iconographic systems.

Yet another possibility for speeding up key access is to make the keyboard ambiguous, i.e. to assign several characters to one key. This obviously reduces the number of keys to be scanned, but creates the problem that the ambiguity has to be resolved. A typical example for an ambiguous keyboard is the numeric keypad of a mobile phone, where each number also comprises 3 or 4 letters (1 $\rightarrow$ *a,b,c*; 2 $\rightarrow$ *d,e,f*; ...). The ambiguity can be resolved either by multiple selection (e.g. selecting a key twice for typing the second character, *multi-tap approach*), or by automated disambiguation of the key sequence. A well-known disambiguation strategy is offered by the T9™ system[8], which is installed on almost every mobile phone. It predicts the most likely word from an ambiguous input, using frequency tables of words.

Another ambiguous keyboard is the *Uniglyph* system (Belatar & Poirier, 2007) working on three keys only. Each of the keys corresponds to a graphic primitive (glyph), grouping a number of characters that contain it (Figure 2.5).

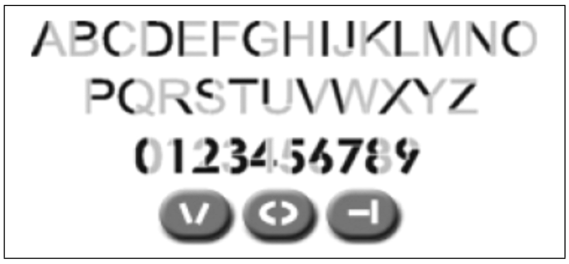


Figure 2.5: The three keys of the *Uniglyph* ambiguous keyboard, representing character primitives (cf. Belatar & Poirier, 2007)

To enter a word using *Uniglyph*, the user simply has to select a suite of glyphs, which are then disambiguated by the system. If the suite cannot be disambiguated, a list of alternatives is presented to the user, who then has to select the wanted element.

It is obvious that not all of the keyboard arrangements, scanning and selection strategies can reasonably be combined with each other. For example, while it might theoretically be optimal to dynamically rearrange an ambiguous keyboard, a user will soon give up due to the tremendous cognitive load implied.

2.3.2 Reducing the number of keystrokes

The recent advances in Natural Language Processing (NLP) had very stimulating effects on the development of AAC techniques and systems. Especially

[8]*T9* refers to "Text on 9 keys", and it is developed by *Nuance Communications*, http://www.nuance.com/t9/

with regard to the enhancement of message composition, various NLP techniques have been applied that exploit the regularities of natural language. This can be done at different levels of linguistic analysis, on the lexical and syntactic, but also on the semantic level. To illustrate how NLP can in principle support an AAC user, let us consider a typical example with a human interlocutor: Suppose a non-speaking person manages to select the word onset *'san'* from a communication board. As little as this could suffice a human interlocutor to complete the word to *'sandwich'*[9]. Since *'sandwich'* is a count noun, and refers to an object that is normally eaten, and we can suppose that the person has a certain intention to utter this word, it would make sense to form a full phrase like *"I would like to eat a sandwich"*. Up to this point we have exploited information at the three mentioned linguistic levels: Lexical probability, knowledge about the syntactic well-formedness of a sentence, and semantic knowledge on the mentioned concept *'sandwich'*, being an edible object. Current AAC systems are of course far from the level of abstraction depicted here, but they do try to exploit the same kind of information in order to reduce the number of selection steps to be performed. We can grossly distinguish between five kinds of approaches:

- Language generation
- Text retrieval
- Top-down composition
- Bottom-up composition
- Abbreviation expansion

Language generation techniques allow the user to enter a few keywords (or select a number of icons), from which the system tries to construct a meaningful and grammatical text message. Considering the already mentioned example, a user could enter *"eat sandwich"* or select the corresponding icons, which the system then translates into a full message. While these approaches are in principle able to considerably reduce the user effort, they normally have trouble to deal with ambiguity. And a wrong translation of the intended message is very penalizing, since the user then has to delete and re-enter the message (in a different manner), which implies a loss of time as well as additional cognitive effort.

The underlying idea of *text retrieval* methods is to provide the user with a large set of predefined messages or text snippets which can quickly be retrieved during communication. Such approaches either define a number of typical contexts or perspectives (like the TALK system, cf. Todman *et al.*, 1999) which can be browsed for, or they rely on techniques from classical full text

[9]We must not neglect the effects of contextual knowledge here: It is very easy to guess 'sandwich' from 'san' around lunch time.

retrieval, matching one or more query terms in a set of messages. Whereas standard IR approaches however aim more at achieving high precision in the retrieved documents, text retrieval methods in AAC try to establish a high recall with minimal input (number of key words).

Top-down approaches offer a set of communication scenarios, i.e. a constrained set of message components, from which a full message can be composed. With respect to our example, a user could first select the subject (e.g. ***I**, You, He, Mary, ...*) then the predicate (e.g. *am, have, would like*), then a continuing verb phrase (e.g. *to sleep, to drink* [x], *to eat* [x]) and finally choose from a list of food objects the intended one. Whereas this kind of approach anticipates a lot of information and can therefore lead to a considerable reduction of selection steps, it severely constrains the user's possibilities of expression, since only those phrases can be composed whose components are included in the selection sets. It is therefore mainly applied in task-oriented communication situations such as describing medical conditions (e.g. *"I feel nauseous"*, *"My* [x] *hurts"*).

Bottom-Up composition on the other hand does not impose any kind of restriction on the intended message. Each symbol is selected separately, which implies a greater degree of freedom, but usually also more selection steps. To reduce the effort in bottom-up composition, various techniques are applied that try to predict the following textual element (usually a word), based on the already entered message part. Many AAC systems incorporate a text prediction component by displaying a list of words, from which the intended one can be selected; others try to complete the current word directly in the editor.

Finally, *abbreviation expansion* (also known as *'disabbreviation'*) techniques are based a set of predefined abbreviation/equivalent pairs. If the user enters an abbreviation, it is replaced on the fly by the corresponding word or phrase. While this is potentially very helpful, the set of abbreviations has to be known to the user, which implies that their number is rather limited. For this reason abbreviations are normally stored only for frequently recurring forms or phrases, e.g. *"gm"* → *"Good morning!"*. There have been attempts to automatically generate abbreviations, based on a set of rules (cf. Stum & Demasco, 1992; Moulton *et al.*, 1999)), but this includes the risk of unwanted expansion, which is again very penalizing. Abbreviation expansion has been shown to be very helpful, if the user is able to define her or his own abbreviations, because they are adapted to the user's needs, and they are usually better memorized.

2.4 AAC systems and research projects

2.4.1 Scientific efforts in AAC research

The following sections will give an overview of the most important research projects and systems in the field of (text-based) AAC. It is not meant to be exhaustive, but to give insights in the methods applied and the principal design decisions. In order to keep an evolutionary perspective, the projects are presented (mostly) in chronological order. In the end, some of the major commercial systems in this area are also mentioned.

PAL and Syntax PAL

The *Predictive Adaptive Lexicon* (PAL) project (Swiffin *et al.*, 1987a,b), developed at the University of Dundee, UK, follows a bottom-up strategy of message composition (cf. 2.3.2). It is one of the first systems offering a word prediction component in an AAC text entering system. Word prediction is integrated in form of a list which is updated after every insertion. The predictor makes use of lexical frequencies and it can be extended with new words.

Syntax PAL (Morris *et al.*, 1992) has been developed on top of the PAL architecture. The previous model is augmented by a syntax-oriented component using transition probabilities of parts-of-speech (PoS), which makes the prediction sensitive to the current context.

Compansion

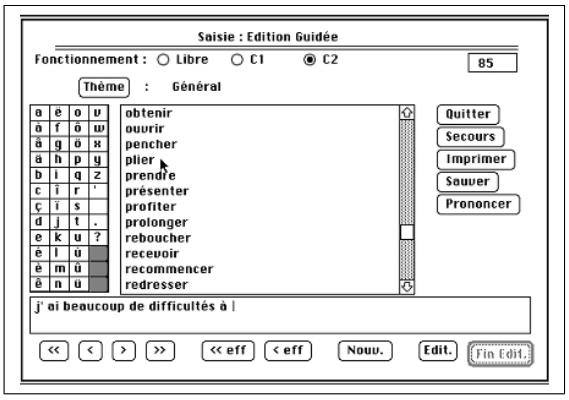
Developed at the University of Delaware, the *Compansion* system (Demasco & McCoy, 1992) follows a language generation strategy: The user can insert or select telegraph-like input, consisting of a few uninflected words, from which the system constructs a well-formed sentence. The system comprises three major components: (i) a word order parser, relying on a syntactic grammar, groups the inserted words into larger constituents and adds further syntactic information; (ii) a semantic parser then reasons about the intended meaning of the content words and constructs a semantic interpretation; finally (iii) a translator component generates a well-formed sentence from the previously constructed representations. To give an example (cf. McCoy & Demasco, 1995):

Input:	`5 apple eat John`
Output:	`5 apples were eaten by John`

While in principle the system is able to reduce typing effort, it is obvious that expressivity is largely restricted for the user, who is limited to the predefined 1000-word vocabulary of the system and to the implemented phrase structures. Another questionable aspect is structural ambiguity, which often cannot be resolved automatically.

KOMBE

The European *KOMBE* project (Guenthner *et al.*, 1992) aimed at developing a communication system for ALS patients (cf. 2.1). The architecture is based on a guided sentence composition (i.e. top-down) strategy, where a given message is composed by selecting each component (word or constituent) one after the other. The approach relies on a grammar, conceptual rules and a lexicon in order to suggest meaningful continuations for a given message beginning. The vocabulary is especially tailored to medical purposes in order to facilitate the dialogue between an ALS patient and a medical practitioner.


Saisie : Edition Guidée

Fonctionnement : Libre C1 [✓ C2] 85

Thème : Général

obtenir
ouvrir
pencher
plier
prendre
présenter
profiter
prolonger
reboucher
recevoir
recommencer
redresser

j' ai beaucoup de difficultés à |

Figure 2.6: The interface of the KOMBE AAC system

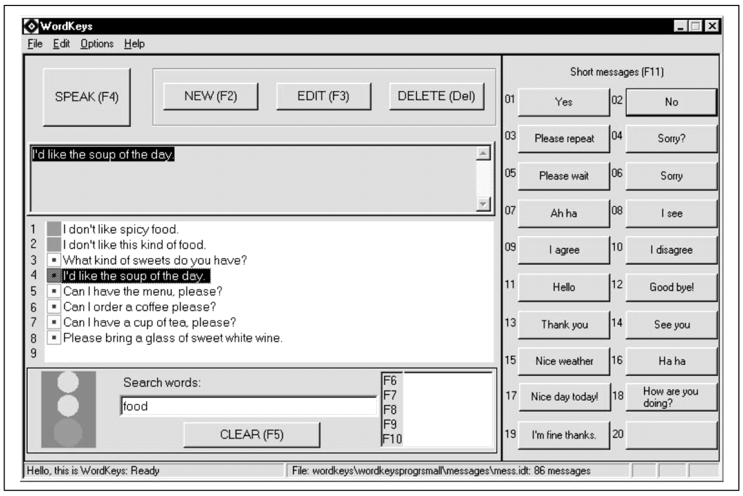
The interface of *KOMBE* consists of a virtual keyboard, a word list, from which the the wanted items can be selected, and an editor for composing the message. Another component of *KOMBE* was designed for illiterate persons, using a pictographic communication system (cf. section 2.2.1); it enables select a set of icons, which are then translated into a textual message.

Predict / Profet

Since the early 1980s researchers at the KTH in Stockholm have been working in the field of AAC. The *Predict* system (Hunnicutt, 1986, 1989) consisted of a writing aid that could be used on a normal PC together with a speech synthesis. The word to be typed was predicted using a frequency-ordered lexicon and a list of previously used words; moreover new words could be integrated. The system underwent several extensions, and it was tried to incorporate more complex linguistic information such as deeper syntactic and semantic structures (cf. Hunnicutt, 1989). *Profet* then (Carlberger *et al.*, 1997) is a further development of *Predict*. It combines frequency and syntactic information, and it offers possibilities to manually modify the lexicon or to create new lexica.

WordKeys

WordKeys (cf. Langer & Hickey, 1998) is a communication aid incorporating predefined messages and a phrase retrieval system. The user can insert new messages which are then automatically indexed and can be retrieved via a search window (cf. Figure 2.7). The retrieval component makes use of a semantic lexicon which expands the query by synonymic and hyponymic expressions, moreover the query as well as the indexed messages are morphologically analyzed. The ranking of the retrieved messages is then performed according to their morphological and semantic similarity with the query. In a user evaluation it could be shown that most of the time one key word is sufficient for retrieving the intended message.



I'd like the soup of the day.

1 I don't like spicy food.
2 I don't like this kind of food.
3 What kind of sweets do you have?
4 I'd like the soup of the day.
5 Can I have the menu, please?
6 Can I order a coffee please?
7 Can I have a cup of tea, please?
8 Please bring a glass of sweet white wine.
9

food

01	Yes	02	No
03	Please repeat	04	Sorry?
05	Please wait	06	Sorry
07	Ah ha	08	I see
09	I agree	10	I disagree
11	Hello	12	Good bye!
13	Thank you	14	See you
15	Nice weather	16	Ha ha
17	Nice day today!	18	How are you doing?
19	I'm fine thanks.	20	

Hello, this is WordKeys: Ready
File: wordkeys\wordkeysprogrsmall\messages\mess.idt: 86 messages

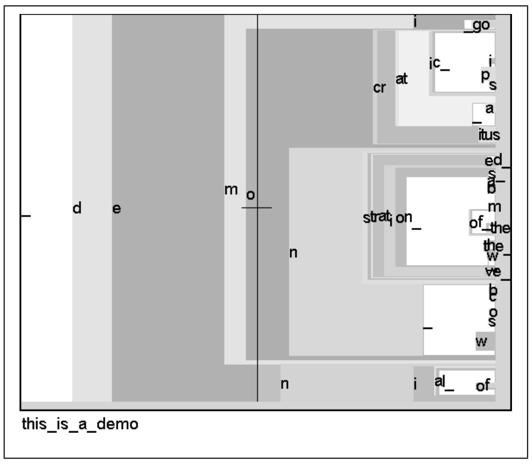
Figure 2.7: The interface of the *WordKeys* system

HandiAS

The project *HandiAS* (Le Pévédic, 1997; Maurel *et al.*, 2000; Maurel & Le Pévédic, 2001), developed at the universities of Nantes and Tours (France), primarily focuses on the improvement of the word prediction process. It is based on a hybrid architecture, combining weighted token automata, morphosyntactic dictionaries and statistics on word usage. The model analyzes the left context (i.e. the already entered part of a message) for morphological and syntactic features in order to predict the syntactically most appropriate lexical items, which are then proposed to the user. The approach includes promising NLP techniques, however it was only prototypically implemented and could not be evaluated in a real-life situation.

Dasher

The interfaces of the already presented systems all followed the same paradigm: A list of lexical items is presented to the user, who can select from it the intended element, which is then added to some editor. *Dasher* Ward *et al.* (2000) follows a very different strategy: It offers a dynamically modified interface in which the user can browse by controlling a mouse or using eye movements. Like in a somewhat old-fashioned computer game, all letters of the alphabet arrive from the right-hand side; in order to form a word, letters are to be threaded by moving the cursor towards the intended symbol; as the movement continues, new letters arrive from the right. The arrangement of the letters as well as the size of their selection fields is estimated by letter and word statistics, so that the most probable items are the easiest to select.



this_is_a_demo

Figure 2.8: User Interface of *Dasher*

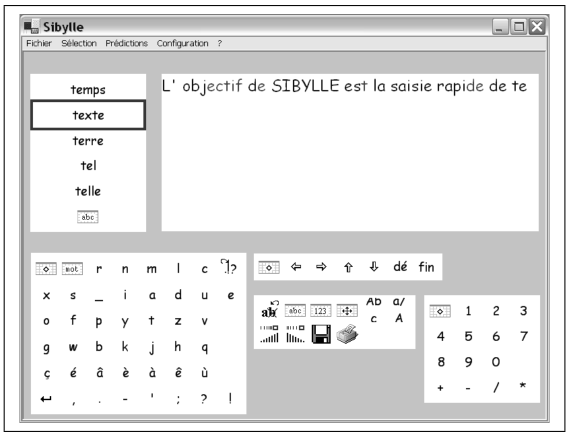
According to the developers, communication rates of up to 34 words per minute can be reached using *Dasher* with a mouse, but reports on the cognitive load as well as on selection by eye tracking have not been made. So, this interface might be a helpful solution for some users, who still have fine motor control, but not for those having uncontrolled movements (e.g. in the case of cerebral palsy).

VITIPI

The system *VITIPI* (*Versatile Interpretation of Text Input by Persons with Impairments*) has been developed at the IRIT in Toulouse, France (Boissière, 2000). While also being an unrestricted, bottom-up system, *VITIPI* follows a different completion strategy: Instead of displaying a list of predicted words, *VITIPI* proposes word parts to be selected by the user. If a word can be unambiguously terminated, the ending is directly inserted to the text. Since the lexicon of *VITIPI* is morpheme-based, unknown words can be completed as well, as long as they terminate on a common ending. Moreover, *VITIPI* offers an automated correction of the inserted text, which can be very useful for persons with loss of fine motor control, who frequently make selection errors. According to an evaluation study, the system is able to correct 72% to 75% of all typing errors (cf. Boissière & Dours, 2000).

SIBYLLE 2003

The project *SIBYLLE* has started at the *Université Bretagne-Sud* in Vannes, Brittany in 2001; it was developed in the context of the PhD thesis of Igor Schadle (Schadle, 2003).[10]. The initial goal was to realize two complementary communication aids within one system. The first one, *SibyLettre*, focuses on enhancing the (binary) letter selection process by dynamically rearranging the keyboard after every insertion, according to the left context (cf. sections 2.3.1 and 6.1.2). In this way, the most probable letters appear first on the keyboard and can be selected quicker.



L' objectif de SIBYLLE est la saisie rapide de te

Figure 2.9: User Interface of *Sibylle 2003*

The second component is *SibyMot*, a word predictor, based on a model performing a partial syntactic analysis of the left context. It presents the predictions in a word window, right next to the editor (cf. Figure 2.9).

Fasty / EMU

The EU funded project *Fasty* has been realized at the Technische Universität, Vienna (cf. Zagler *et al.*, 2003; Beck *et al.*, 2004; Trost *et al.*, 2005); *EMU* is its commercial offspring[11]. As for *Profet* (cf. 2.4.1), its interface only consists of a word prediction list, which can be used with any text editing device and follows the actual position of the text cursor; words are selected via the function keys.

The prediction engine is based on a combined model taking word sequence statistics as well as part-of-speech and contextual information into account, the

[10]Since it is the precursor of the system presented in this work, our main intention here is to distinguish the contributions of Igor Schadle from ours. A detailed description of the present system will follow in chapter 6.

[11]http://www.is.tuwien.ac.at/emu/

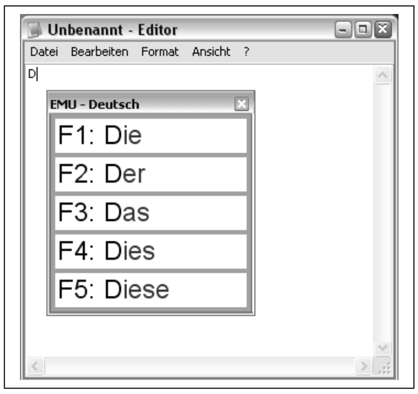


Figure 2.10: User Interface of *Fasty/EMU*

lexicon can also be modified to the user's wishes. Being a European project, its objective was to develop a generic approach, in which many European languages can be treated. Therefore, one of the aims was to deal with the problem of compounding in northern European languages (cf. also section 3.5.4 later on).

Aegys-PCA

An ongoing project is *PCA* (Plateforme de Communication Alternative), being developed at the Université Aix-Marseille, France (Blache & Rauzy, 2007b,a). It offers a text-based as well as an iconographic input system together with different selection and scanning modes; text input is supported by a word prediction module. This module includes a personal, auto-adaptive lexicon and a morpho-syntactic predictor, which analyzes the current context. The iconographic module comprises a reformulation system that constructs from the sequence of selected icons a well-formed sentence.

2.4.2 Commercial systems

Since AAC became not only a research field but also an important market, many companies have focused on the development of communication systems for the disabled. Several of these systems also include word prediction, combined with various scanning techniques in order to facilitate text selection. Among them we find: *Aurora Suite* (Aurora Systems), *Co:Writer* (Don Jonston Inc.), *WordQ* (QuillSoft), *EZ Keys* (Words+), *WordPower* (Prentke Romich Inc.), *DynaWrite* (Dynavox / Mayer-Johnson) and the already mentioned *EMU* (Fortec).

With the rapid development of mobile communication technologies, however, another, much larger market was found for word prediction techniques:

They are becoming used for reduced keyboards such as the numeric keypads of mobile phones or PDA touch screens. The already mentioned *T9*™ predictive text software for example nowadays is installed in more than 2 billion mobile phones world wide. As the development of mobile communication devices is constantly growing, it can be expected that in the near future many other mobile interfaces will include word prediction techniques in order to speed up text entry.

2.5 Evaluation aspects

Considering the variety of approaches it becomes obvious that the standardized evaluation of an AAC system is difficult. The main characteristic is certainly the enhancement of the communication rate, but it strongly depends on the user's cognitive and motoric capacities, which are obviously not comparable.

As for many other examples of human-machine interaction there are two major evaluation paradigms for an AAC system (cf. Garay-Vitoria & Abascal, 2006):

- Environmental evaluation (human testing): based on the observations and the typing speed of several users
- User emulation: an emulation device enters a test corpus and thereby calculates a standardized evaluation measure.

The advantages and disadvantages of these paradigms are well known. On the one hand, human testing provides results that include the influence of human factors like writing errors, fatigue, learning time etc. But these observations depend strongly on the recruited users, what restricts the evaluation to individual case studies. Moreover, human testing is a very time-consuming task, which makes it inappropriate for many optimization processes, where sometimes a large number of parameters have to be tested.

On the other hand, while emulation is fast and yields a reproducible and objective evaluation measure, it completely ignores human factors. It produces only theoretical results which have to be interpreted with care.

Since an AAC system normally comprises several parts and features, some aspects can be distinguished from one another in the evaluation. For example, the key selection process can be evaluated separately from the keystroke reduction strategy. However, since these methods obviously interact, we also have to perform an environmental evaluation of the system as a whole, especially considering its usability and configurability.

In the following we will describe each of these aspects separately. Again, the focus here is on text-based AAC systems; especially the quantitative evaluation methods presented cannot directly be applied to iconographic systems.

2.5.1 Key selection

As already mentioned in section 2.3.1, cognitive aspects play an important role in the key selection process. Theoretic estimations as well as user emulation however also help to characterize the performance of a key selection method. The theoretically optimal setting needs the least number of scanning steps, i.e. the *Average Scanning Distance* (ASD) is lowest. Let us first consider the simplest case, a linear scan operating on a keyboard with n equiprobable keys. On the average we will need to scan half of the keys until the intended one is reached; the ASD is then:

$$ASD = \frac{n+1}{2} \tag{2.1}$$

For a line-column scan on a keyboard with l lines and c columns we get:

$$ASD = \frac{l+1}{2} + \frac{c+1}{2} \tag{2.2}$$

It can easily be shown that the lowest ASD for a line-column scan is always obtained when l = c ($= \sqrt{n}$), so a quadratic keyboard arrangement would be optimal. It can also be shown that this generalizes to a block-wise scanning procedure: The optimal block arrangement for d levels would be $\sqrt[d]{n}$ keys per minimal sub-block. The ASD is then:

$$ASD = \frac{\sqrt[d]{n}+1}{2} \times d \tag{2.3}$$

For example, if we arrange a keyboard of 64 keys in 3 block levels of $4\times4\times4$, the ASD would be $7,5$ scanning steps, which is the lowest possible value for three levels.

The situation becomes a little more complex, if we give up the assumption of equiprobability. We then have to determine the probability $P(k_i)$ and the position $D(k_i)$ of every key k_i ($i = 1 \ldots n$). The *Average Scanning Distance* (ASD) then is calculated as follows:

$$ASD = \sum_{i=1}^{n} P(k_i) \times D(k_i) \tag{2.4}$$

This formula can also be applied to dynamic rearrangements, we then however have to redetermine the position $D(k_i)$ for each key after every selection.

Table 2.2 gives an overview of the *Average Scanning Distance* for different scanning modes and keyboard arrangements, based on a keyboard of 64 keys (lower and upper case + blank + special characters). The estimations for the frequency-based settings are taken from (Schadle *et al.*, 2001) , the dynamic rearrangement is based on the *SibyLettre* letter predictor (for French).

scan:	linear	line-col.	block-wise	linear	linear
arrangement:	equiprob.	equiprob.	equiprob.	frequency	frequency
ordering:	static	static	static	static	dynamic
ASD:	32,5	9	7,5	7,1	2,9
Sel. steps:	1	2	3	1	1

Table 2.2: Average scanning distance and selection steps for different keyboard arrangements and selection modes

It can be seen in Table 2.2 that the arrangement as well as the selection procedure have an important influence on the ASD, but it is also clear that some of the gains shown here have to be traded off against additional cognitive load, induced either by more selection steps or by a permanently changing key arrangement.

2.5.2 Keystroke reduction

Several objective metrics have been proposed to assess the ability of a prediction component to speed up a communication aid. Some of them are directly related to human testing. Soukoreff & MacKenzie (2003) for instance use a *KeyStrokes Per Character* measure, which is a good indicator for the rate of typing errors. Likewise, measures of communication speed (Koester & Levine, 1994) are strongly related to the motor and cognitive abilities of the recruited users.

For the assessment by emulation Fazly & Hirst (2003) list a number of evaluation metrics, which are all closely related:

- **Hit rate (HR)**: The percentage of times that the intended word appears in the prediction list. A higher hit rate implies a better prediction performance. It gives a clear idea of how much the system is able to aid the user.
- **Keystrokes until Completion (KuC)**: The average number of keystrokes that have to be entered until the intended word appears in the prediction list. Here, a lower value indicates a higher performance.
- **Accuracy (ACC)**: The percentage of words that could be predicted (and completed) before the user reached the end of the word.
- **Keystroke Saving Rate (ksr)**: The percentage of keystrokes that could be saved by using the predictor.

Fazly & Hirst (2003) showed that these measures strongly correlate, however the first three make the assumption of a word prediction technique, which is not the only possibility to reduce keystrokes (cf. *language generation* or *abbreviation expansion*, section 2.3.2).

The *Keystroke Saving Rate* on the other hand does not make any prior imposition on the way of keystroke reduction; as its name implies, it simply determines the number of keystrokes that could be saved by applying a given technique. This makes it probably the most intuitive one of the measures described, and it is most commonly applied to evaluate writing aids (cf. Carlberger *et al.*, 1997; Lesher *et al.*, 1999; Trost *et al.*, 2005; Trnka *et al.*, 2006, among others). The *ksr* is formally defined as follows:

$$ksr_n = \left(1 - \frac{k_{red}}{k_{all}}\right) \cdot 100 \tag{2.5}$$

with k_{red}, k_{all} being the number of keystrokes needed on the input device when typing a message with (k_{red}) and without reduction method (k_{all} = number of characters in the text that has been entered).

If the ksr is calculated for a technique using a prediction list (as in most word prediction approaches), we have to specify its number of items n, because the ksr depends obviously on the list's size. The more words are presented the better the *ksr* will be, however the cognitive load on the user rises as well. To find out about this relationship we have calculated the ksr using our word predictor for sizes of the prediction list from 1 to 20 on a French newspaper corpus (cf. section 4.4); the results are shown in Figure 2.11. As the curve in Figure 2.11 clearly shows, the dependency between *ksr* and list size is non-linear: whereas we observe an increase of more than 13% from ksr_1 to ksr_5, ($ksr_1 = 44,4\%$; $ksr_5 = 57,9\%$) the gain between ksr_5 and ksr_{10} is only $3,6\%$ ($ksr_{10} = 61,5\%$). For this reason, a list size between 3 and 7 items seems a reasonable trade-off between saving rate and cognitive load.

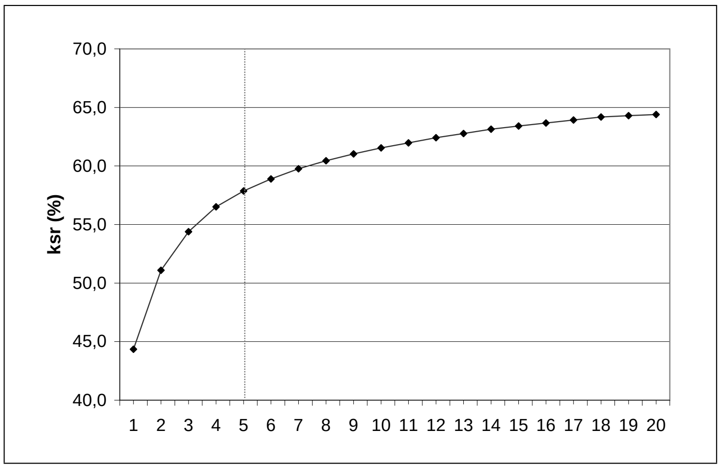


Figure 2.11: Keystroke saving rate against prediction list size

As in most other works applying ksr (cf. Trost *et al.*, 2005; Trnka *et al.*, 2007), the keystroke saving rates presented throughout this document are based on the assumption that one additional keystroke is required to jump to the word selection list and that a space is automatically inserted afterwards. This is a rather hypothetic assumption, because it implies that a predicted word can be selected just as a normal character. However, in many systems the prediction list is separate from the keyboard, which means that actually two selection steps are needed in order to select a word. However, since this is an intrinsic property of the system under consideration, the *ksr* is defined with one additional keystroke.

The flattening of the curve in Figure 2.11 suggests that there is an upper limit for the *ksr*. With respect to a word prediction approach one (theoretical) limit is obvious: Since at least one selection per word is needed, the minimum number of keys selected equals the number of words. The average word length in English (and French) is $4,5$ letters, to which the (automatically inserted) empty space can be added. This gives us a theoretical ksr maximum of $(1 - (1/5,5)) \times 100$ = $81,8\%$. German words are slightly longer (average length = $5,3$), so the maximal ksr for German is $84,1\%$.

State-of-the-art word predictors arrive at a ksr_5 of approximately 50% to 55% (cf. Copestake, 1997). Interestingly, experiments have shown that keystroke savings of human guessers lie in the same range (cf. Lesher *et al.*, 2002). Even though there appears to be room for improvement, it seems increasingly difficult to go beyond this empirical limit. In the first part of the following chapter we will return to this question from an information-theoretic point of view.

Another question that arises with regard to such a theoretic measure is if an improvement in *ksr* really translates into an enhancement of the communication rate of real users. Some researchers have questioned this (cf. Anson *et al.*, 2005). In an experimental study however, Trnka *et al.* (2007) could show that advanced prediction methods have a clear positive effect on communication rate with respect to no prediction (59,9% improvement), but also to simple prediction, based on a word list (44,4% improvement). In this light it seems worthwhile to put more effort in the development of advanced prediction methods for AAC systems.

2.5.3 Environmental evaluation: Usability

While it is certainly reasonable to use the above mentioned emulation-based measures to optimize the components of an AAC system, it is also essential to consider the system as a whole. Research in human-computer interaction has shown for a long time that the design of a user interface is the crucial factor for the resulting performance of the user; countless aircraft accidents for example can be related to misestimations in the cognitive preconditions of human users[12]. Therefore, the notion of *usability* has come to the fore in the past years; standardized methodologies and evaluation patterns were developed in order to determine this rather vague property of a man-machine system (cf. for example the ISO 9126 standard[13]).

However, as these standardized approaches also presuppose a rather standardized user, they are only partially applicable to an AAC system. AAC users surely have common cognitive abilities and demands, however, as every medical condition is different, it is not obvious to define common criteria of usability for such a system (cf. Boissière & Dours, 2002).

A further difficult aspect with respect to user-centered testing of an AAC system is the way of assessment: While standard usability tests largely rely on user feedback and standardized interviews, this is – for obvious reasons – not easily applicable to AAC users. Moreover, while researchers make a lot of assumptions about the communication needs of these persons, only few efforts aimed to make them explicit, such as the French project ESAC-IMC, in which we were also partly involved (cf. section 6.4).

Since users and their requirements are so various, only one *a priori* criterion for the quality of an AAC system can be defined, namely its degree of configurability: The more configuration possibilities, keyboard settings and selection modes a system offers, the better it will suit the abilities and preferences of its respective user.

In the following chapters the major means of evaluation will be the *keystroke saving rate* (for the above mentioned reasons of objectivity and reproducibility), however we hope to have made clear that this cannot be the only criterion. The ”real-life” evaluation of the AAC system as a means of communication remains indispensable; we will return to this aspect in chapter 6.

2.6 Conclusion

The chapter presented the field of Alternative and Augmentative Communication. This discipline aims to provide communicative facilities to persons who are unable to speak or to communicate otherwise. The main challenge of AAC is to enhance the extremely reduced communication rate of such persons. In the past 30 years a considerable number of AAC systems, implementing various methods to support the user's communication, have been proposed. In most of these approaches two major strategies for accelerating text input can be distinguished: (i) Improving the key selection process, and (ii) reducing the number of keystrokes necessary to insert a message. Key selection methods can be evaluated by measuring the *average scanning distance* (*ASD*), the

[12]For a comprehensive introduction to this subject cf. Hawkins (1993)

[13]http://www.issco.unige.ch/projects/ewg96/node13.html

most important evaluation measure for keystroke reduction techniques is the *keystroke saving rate* (*ksr*). An often neglected factor in the evaluation of AAC systems is however also usability. Especially since AAC users have rather heterogeneous demands it is crucial to design an AAC system as configurable as possible.

Chapter 3

Prediction and language modeling

It is this resolution of uncertainty
which is the aim and outcome of communication

JOHN R. PIERCE
(*Symbols, Signals and Noise*, 1961)

Whereas the last chapter focused on the application side of AAC, this part is supposed to provide the theoretical foundations of the models and techniques on which the major AAC approaches are based. The first section (3.1) aims to consider communication and language from an abstract perspective and to answer the question of how prediction can be achieved at all. Basic notions of probability estimation and information theory are also explained. The following two sections (3.2 and 3.3) outline statistical language modeling, a field of research which has become very important not only for AAC but for most other areas of natural language processing. In section 3.4 more practical issues arising from the application of language models (like storage and parameter estimation) are discussed. The last two sections then present two other families of language models, those based on linguistic knowledge (3.5) and maximum entropy models (3.6).

3.1 Information and redundancy in natural language

3.1.1 Communication as a discrete noiseless system

Human language can certainly fulfill many aims or intentions; the most obvious one however is to transport information. In this regard it seems at first astonishing that it should at all be possible to predict any element of a language signal on the sole basis of what has already been said, because this means that the predicted element is superfluous or redundant, it does not carry information. But why should humans spend so much effort and time on transmitting redundant symbols? The commonly given answer is that redundancy protects the language signal from possible errors during transmission (cf. Pinker, 1994); therefore, even though a piece of the message is distorted or lost, it can still be reconstructed.

Even though its doubtful connotation, redundancy is a vital means of every natural language as well as of most artificial languages. And it seems to be the key concept for understanding the prediction of a language signal and for the development of optimal prediction algorithms. But how can this rather fuzzy and invisible concept be characterized and quantified?

In order to understand redundancy it is helpful to first get a better understanding of its antipode 'information' and the nature of communication itself. The landmark event for the formal description of communication was the development of *communication theory* (later also called *information theory*) by Claude Shannon at the *Bell Laboratories* in 1948 (cf. Shannon, 1948). Shannon describes a communication system in its most general form: (i) An information source (or sender) selects a message bearing the intended information from a set of possible messages. (ii) This message is encoded into a form making it suitable for transmission. (iii) The encoded message is transmitted via a communication channel (a medium) and arrives at its destination (or receiver), who has a decoding device in order to translate the message into an internal representation (iv). A schematic overview of this process is given in Figure 3.1.

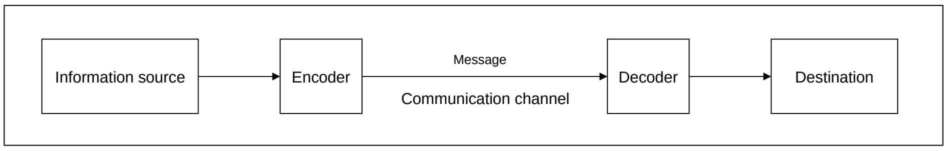


Figure 3.1: Description of signal transmission on a noiseless channel (from: Shannon, 1950)

Following this model, communication can only succeed if two preconditions are fulfilled: First of all, the form of encoding has to be suitable for the given channel and secondly the transformation performed at the encoding step has to be the exact inverse of the operations performed at decoding, i.e. the encoding standard has to be known by both communication partners.[1]

According to the kind of message transmitted, communication systems can be classified into three main categories: discrete, continuous and mixed. Whereas in a discrete system, both the message and the transmitted signal consist of a sequence of discrete symbols, in a continuous system they are treated

[1]While violations of the first condition are quite rare, the world is full of violations of the second, especially if someone who does not speak Mandarin tries to order a meal somewhere in rural China.

as continuous functions, and the mixed system makes use of both kinds of representation.

And yet another aspect has to be characterized: Is the signal on its transmission affected by noise or not? If noise can alter or mutilate the message on its way, the system has to provide a means of restoring the original message. But for the moment let us consider the most simple case: a discrete, noiseless system:

A discrete information source can emit a message by generating a symbol sequence from a finite set of s (discrete) symbols. A first naive guess would be that the amount of information to be transmitted now depends on the number of different possible symbols, i.e. transmitting one of two symbols should imply less effort than transmitting one out of 1.000. This becomes especially clear if we convert n into a binary format: the outcome of two possible symbols can be coded by one bit, whereas we need 10 bits to code one out of 1.000 symbols ($log_2 1000 \approx 9,97$). Shannon's big leap was to show that the transmission effort does not depend on the number of possible symbols but on their probability distribution. He therefore introduced a new term which he borrowed from thermodynamics: *entropy*, which describes the amount of a thermodynamic system's energy to do work.[2] Shannon defines the entropy H of a single variable X (x being the possible outcomes of X) as follows:

$$H(X) = -\sum_{x \in X} p(x) \times log_2 p(x) \tag{3.1}$$

There are several ways to understand this measure: it can be seen as the average uncertainty about the outcome of a given event. Mostly however it is considered as the amount of information that is transmitted in a communication process. To illustrate this by a short example, let us imagine a variable with 8 different outcomes (e.g. sending one out of 8 symbols). If all outcomes are equiprobable, the entropy gives us the expected value of 3 bits ($H = -\sum_1^8 \frac{1}{8} \times \log_2\left(\frac{1}{8}\right) = -\log_2 \frac{1}{8} = 3$). However if p_1 becomes more probable than $p_2, \ldots, p_8$, the entropy gets smaller and smaller. Table 3.1 shows different probability distributions and the respective entropy over $p_1, \ldots, p_8$.

Whereas in the equiprobable case (1) we have to transmit 3 bits in order to report the outcome, only 0,56 bits need to be transmitted in case 4, because most of the time the outcome is the same (p_1). For this case one could design a kind of *emergency code*, that transmits only if the outcome is *not* p_1. This code would transmit in only 7% of the cases, so the average number of bits over all outcomes is highly reduced.

However, measuring entropy does not (directly) help constructing an optimal code, it has to be understood as the lower bound for the number of bits

[2]Note that the term of entropy has also been defined in statistical mechanics, where it describes the number of microscopic configurations that result in the observed macroscopic description of the thermodynamic system. The definition of entropy by Ludwig Boltzmann has strong similarities with the information-theoretic definition: $S = k \times logW$.

$Dist.$	p_1	$p_2 \ldots p_8$	Entropy
1	$\frac{1}{8}$	$\frac{1}{8}$	3
2	0,3	0,1	2,85
3	0,65	0,05	1,92
4	0,93	0,01	0,56

Table 3.1: Entropy for different probability distributions

that have to be transmitted in order to pass a message. The most common strategy for creating an optimal code is to assign short bit sequences to frequently used symbols and longer sequences to symbols of lower probability. One of the most efficient encoding strategies, developed by Huffman (1952), does exactly this, and in human language we find the same idea: Usually the most frequently used words are very short, whereas less frequent words are longer.

So far, the entropy is defined for only one variable (i.e. one symbol is transmitted). But most of the time we want to transmit symbol sequences. Calculating the entropy of a symbol sequence ($X_1^n = x_1, x_2, \ldots, x_n$) is straightforward:

$$H(X_1^n) = -\sum_1^n p(x_1^n)\ \log p(x_1^n) \tag{3.2}$$

We can then calculate the entropy of a single symbol (entropy rate) by dividing it by n:

$$\hat{H}(X_1^n) = -\frac{1}{n}\sum_1^n p(x_1^n)\ \log p(x_1^n) \tag{3.3}$$

But this would only give us a measure for sequences of length n; if we consider a language, we would not want to limit a communication process to n words; the length of our sequences should be infinite.

If we define a *language* L (in a very broad sense) as a symbol sequence of infinite length, the entropy of that language is calculated as:

$$H(L) = -\lim_{n\to\infty}\frac{1}{n}\sum_{X\in L} p(x_1, x_2, \ldots, x_n)\ \log p(x_1, x_2, \ldots, x_n) \tag{3.4}$$

Conditions become easier if we conceive of all possible sequences X as only one that is long enough that it comprises all others. This is the underlying idea of the *Shannon-McMillan-Breiman* theorem, but it requires two assumptions:

The information source generating that language has to be both *stationary* and *ergodic*. *Stationarity* means that the probability assigned to any subsequence does not change over time, i.e. it does not matter at what point we consider a sub-sequence, because its dependency on past symbols is limited. *Ergodicity* refers to the condition that the statistical properties of any two sub-sequences be the same, i.e. that we find on the average the same number of symbols x_i in each of them.

The *Shannon-McMillan-Breiman* theorem then defines the entropy of a language L using the infinite sequence of symbols $(x_1, x_2, \ldots, x_n)$ as follows:

$$H(L) = -\lim_{n\to\infty} \frac{1}{n} \log p(x_1, x_2, \ldots, x_n) \tag{3.5}$$

So far we have assumed that we have full knowledge of the probability distribution from which a sequence is produced. But in natural language this is rarely the case. Instead we apply a model m to approximate the true probability distribution. We then can define the *cross entropy* for a given model m simply by replacing the true probability distribution p by probability estimates P_m as provided by m:

$$H(P, m) = -\lim_{n\to\infty} \frac{1}{n} \log P_m(x_1, x_2, \ldots, x_n) \tag{3.6}$$

Of course, this does not help much by itself, but it can be shown that for any model m approximating p the entropy determined over m is an upper bound for the real entropy, i.e. $H(p) \leq H(P, m)$.

If we wish to determine the cross entropy of a given language (in its broad sense, s. a.) with respect to a model, it is usually quite inconvenient to work with an infinite symbol sequence. We therefore define an approximation to cross entropy by setting the symbol sequence to a finite length N:

$$\widetilde{H}(L, m) = H(X_1^N, m) = -\frac{1}{N} \log P_m(x_1, x_2, \ldots, x_N) \tag{3.7}$$

This is the formula, which is usually referred to when people speak of *entropy*. Since true cross entropy cannot be determined they calculate an approximation to *cross entropy* (i.e. with respect to a model) on a symbol sequence of finite length X_1^N.

With the notion of cross entropy, based on a model m, we are finally able to define (information-theoretic) *redundancy*. The model with the highest cross entropy for a language L is the one assuming equal probability for every symbol (= m_0). The cross entropy of such a model, using an amount of s symbols, is always:

$$H(X_1^N, m_0) = -\frac{1}{N} \log p_{m_0}(\underbrace{\frac{1}{s} \times \frac{1}{s} \ldots}_{N\ times}) = -\frac{N}{N} \log p_{m_0}(\frac{1}{s}) = \log s \tag{3.8}$$

So, the cross entropy of our equiprobable model m_0 is $\log s$, and every other model can be compared to this baseline. The redundancy R over a test sequence X_1^N, given a model m is then simply the difference (in percent) of the model's entropy with respect to the maximum entropy over X_1^N:

$$R(X_1^N, m) = 100 - \frac{H(X_1^N, m) \cdot 100}{H(X_1^N, m_0)} \tag{3.9}$$

Cross entropy is also the basis for another (even more) popular measure, which is nowadays applied everywhere in statistical NLP: *Perplexity* (PP) is simply defined as the logarithmic inverse of this approximative cross entropy:

$$\begin{aligned} PP(X_1^N, m) &= 2^{H(X_1^N, m)} \\ &= P_m(x_1, x_2, \ldots, x_N)^{-\frac{1}{N}} \\ &= \sqrt[N]{\frac{1}{P_m(x_1, x_2, \ldots, x_N)}} \\ &= \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P_m(x_i | x_1, \ldots, x_{i-1})}} \end{aligned} \tag{3.10}$$

The perplexity measure can also be understood as the inverse geometric average of the single probabilities assigned by our model m to each symbol of the test sequence. Another way to consider perplexity is the average difficulty of a guessing task. A perplexity of 100 means that on the average we would have to guess between 100 distinct symbols, which is obviously easier than guessing between 1000 symbols. A model achieving a lower perplexity on a (sufficiently large) test sequence is therefore a better approximation to the true probability distribution and likewise a better predictor.

It is a strange matter of fact that people prefer to work with perplexity, even though cross entropy seems to correlate better with many performance measures in NLP tasks (cf. Goodman, 2001; Manning & Schütze, 1999). One reason might be that this "inverse average probability" is more intuitive to understand, another might be that many research efforts are better to sell using perplexity: A perplexity reduction of 50% corresponds to a cross entropy reduction of only 1 bit, so if someone intends to "improve results by 50%", it is much easier to base this statement on perplexity.

3.1.2 *Shannon* game playing

So far, we have only regarded a particular form of symbol sequences, namely those of which the source is *stationary* and *ergodic*. If we want to apply the notions made so far to natural language, we have to assume it to be stationary and ergodic as well. This is unfortunately not the case: Natural language is not stationary, since the probability of a word or a phrase can depend on events of arbitrary distance. And whereas the total of all speakers and writers of a given language at a given time could be regarded as an ergodic source, it is hardly possible to determine a truly ergodic subset of it,[3] moreover natural language is subject to constant changes in usage. For example, it is estimated that on the average every day 20 new words (true neologisms) can be found in German newspapers.[4], and it goes without saying that no model can provide the statistical properties for future words.

So, the question arises whether the constructs introduced above are useless when applied to natural language. Let us therefore (similar to Shannon, 1948) regard some approximations to "real language", for which the conditions of stationarity and ergodicity hold:

The probabilities have been estimated on English newspaper corpora (*The Guardian*, 1997-1998, 49 million words), the size of the vocabulary is app. 133.000 words (cf. also section 4.4). A 0-order word approximation does not apply any statistical assumptions at all, every word occurrence is considered equiprobable and independent. A 1-order approximation takes into account the single word probabilities (with respect to their frequency in a given corpus). A 2-order approximation estimates probabilities for sequences of two words (bigrams), a 3-order approximation considers sequences of length 3 (trigrams), and so on.

Order	Example	Entropy
0	*hurried than furnishing hat acknowledge exit sumptuary far*	17,02
1	*representing speedily and is good a apt or come can the here*	9,6
2	*the old man and in frontal attack on an english writer that the*	7,29
3	*there are no tables which work equally well as either black*	6,85

Table 3.2: Probabilistic approximations to English of order 0-3

Two important aspects can be observed in Table 3.2: First, whereas the 0-order approximation constitutes an arbitrary sequence, the approximations of higher orders become more and more similar to real English. Especially when we consider the 3-order approximation, which constitutes a nearly grammatical sentence fragment, it seems admissible to treat natural language as a stationary and ergodic process even if it is not. Secondly, we can observe that the *cross entropy* drops considerably by nearly 60% from 17,02 ($=log_2 133.000$) to 6,85 bits per word. It therefore seems that with the rising order of an approximation we obtain a better and better estimate of the true entropy of a language.

This was also Shannon's idea, when he designed an experiment in order to estimate the entropy of English (Shannon, 1951). He let human subjects guess a random text snippet letter by letter, and he recorded the number of trials the

[3]Remember that the statistical properties (e.g. the relative word frequencies) must remain the same.

[4]Cf. http://www.wortwarte.de

subjects took to guess each letter. A typical test result might have looked as follows (from: Shannon, 1950):

```
(A) THERE IS NO  REVERSE ON A MOTORCYCLE ...
(B) 1115112112111517111213212271111411111 ...
```

Shannon remarked that the "number of guesses"-sequence in line B can be seen as a translation of the real text, therefore the entropy of line B should be the same as that of line A. To get an estimate of the relationship between the number of trials and the size of the context (number of characters known), he transformed his guessing results (all the lines B) into a large table with the numbers of trials as lines and the number of characters known as columns. In this fashion he could determine an upper and a lower entropy bound for each approximation of rank N (N being the number of letters known). If 100 characters (of an alphabet of 27 characters) were known ($N = 100$) he measured a *per-letter* entropy of 0,6 (lower bound) to 1,3 bits (upper bound).

This famous experiment, which is nowadays known as the *Shannon game*, was repeated in some variations several times. Cover & King (1978) for example altered the experiment by having the subjects place bets on each character instead of repetitive guessing. Interestingly they arrived at the same estimate of around 1,3 bits per symbol.

If we take the average length of English words (4,5 letters) as a basis, we arrive at an upper bound for the per-word entropy of 5,85 bits. This seems to be in accordance with our cross entropy estimates in Table 3.2. The 3-gram approximation yielded a cross entropy of 6,85, which is only one bit higher. It therefore seems that such a model is already able to capture a large amount of the stochastic properties of natural language (of English at least).

3.1.3 Information theory and AAC

Some readers might ask now in what sense ergodic processes and entropy bounds relate to an AAC system, and indeed, this connection ought to be clarified. First of all, it is helpful to consider the role of an AAC system in a communication process. As explained in chapter 2, a speech and motion impaired person often has full mental capacities; considering Shannon's communication scheme (Figure 3.1) we can state that the source of the information is unaffected. However, the encoding process is disrupted, because the speech organs (or limbs) which transform the information into a message are unusable, the message cannot be encoded properly. This is point where an AAC system comes into play; it represents a workaround of the normal encoding device. It receives a reduced signal and tries to find a proper encoding; the proposed message (or message part) is presented to the user, who compares the encoded message with the intended signal. If they match, the user proceeds with the signal transmission, if not, she or he has to modify the proposed encoding by further specification. When completed, the message is then transmitted via an auditive (by speech synthesis) or a visual (by text) channel to the destined communication partner. Figure 3.2 gives a schematic overview of an AAC-supported communication process.

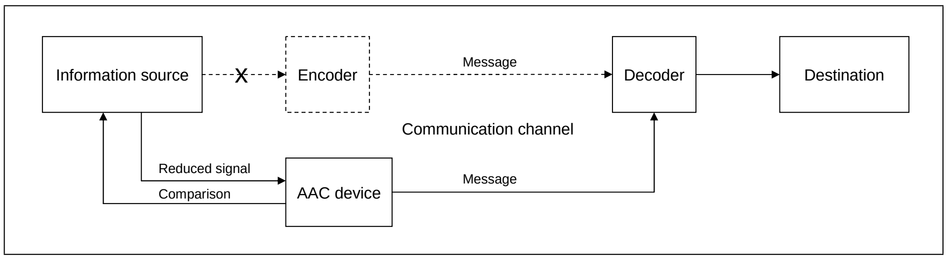


Figure 3.2: An AAC-supported communication system

In this light it becomes also clear that the most efficient AAC system is the one that needs the least input from the user in order to encode the full message; when the reduced signal is as free from redundancy as possible, the user effort is maximally reduced.

The previous parts have shown that there is a theoretical limit to this reduction, and it was also explained that an approximation to this limit can be achieved by the creation of a model that takes the probabilistic properties of the expected signal into account. Shannon showed that the capacities of such a model depend on the amount of contextual information exploited: The more of the context is considered the better the model can predict the reduced signal. So, we can claim that a word predictor plays an abstract version of the above described *Shannon game*.

The notion of signal reduction reminds strongly of an AAC evaluation measure that was already presented in the last part of the previous chapter (cf. 2.5.2): the *keystroke saving rate*. The question that arises here is how keystroke savings and the terms of cross entropy or perplexity are interrelated. Both measures describe signal reduction (or redundancy), however cross entropy is based on the probabilistic properties of the underlying prediction model, whereas keystroke savings are empirically determined.

To get a better understanding of this relation, keystroke savings as well as cross entropy and perplexity were determined for a series of 43 French corpora (from different newspaper, literature, e-mail and transcribed speech corpora, each of 2000 to 9000 words in length). The prediction was performed by a 4-gram model[5] trained on 44 million words from French newspaper (*Le Monde*, 1998-1999), the size of the vocabulary was 141.078 words (cf. also 4.4). To reduce further influences as much as possible, keystroke savings were based on a word list of length 1. Table 3.3 shows the correlation coefficients of the keystroke savings with each of the information-theoretic measures.

[5]The model uses back-off and *modified Kneser-Ney* discounting (Goodman, 2001) and is reduced by *Stolcke* pruning ($\theta = 10^{-7}$) (Stolcke, 1998). These terms will be explained in the following sections (3.3, 3.4.2).

	ksr_1-cross entropy	ksr_1-redundancy	ksr_1-perplexity
intra-register only (8 corpora)	-0,868	0,868	-0,784
all (43 corpora)	-0,678	0,678	-0,392

Table 3.3: Correlation between keystroke savings and different information theoretic measures

The correlations were calculated separately for the 8 intra-register test corpora (i.e. corpora of the same register as the training corpus, here newspaper), as well as for all corpora. Three major observations can be made from the results. The first is trivial: Since redundancy can be derived by linear scaling from cross entropy, their correlations with ksr have the same absolute value. Secondly, we observe weaker correlations for perplexity than for cross entropy. This shows once again (as mentioned in section 3.1.1 and also discussed by Goodman (2001)) that perplexity, despite its high popularity, may be not the most suitable performance measure for an AAC system as for other NLP tasks.

Thirdly, we remark an important difference between the results for the intra-register data and all corpora. This is most probably due to the fact that the non-journalistic corpora do not exhibit the same statistical properties as the data on which our model is based, i.e. they represent an even less ergodic information source (cf. 3.1.1).[6] The more the styles of the corpora differ from the style of the training data, the less precisely our model can estimate the correct probabilities.

All in all, we can state that there is an important correlation between keystroke savings and cross entropy, but it is not perfect. On the one hand this is due to the just mentioned violation of the ergodicity constraint, which affects the model quite strongly, and on the other hand we have to take into account the way word prediction is carried out: In a standard left-to-right setting (i.e. when words are predicted based on the left context) some redundancy might not be accessible for prediction. This has already pointed out by Copestake (1997): Consider an artificial language consisting of equiprobable 4-letter words that all start with the common prefix 'zzz' (i.e. *zzza*, *zzzb*, ..., *zzzz*). It is obvious that this language is at least 75% redundant, however words cannot be predicted until the prefix has been inserted. One might argue that natural language does not work like this, but the problem does become visible: For example when one tries to predict inflected French or German verbs (e.g. consider the inflected forms of *demander* ('to ask'): *demand-e*, *demand-es*, *demand-ons*, *demand-ez*, *demand-ent*, *demand-é*).

A conclusion from these considerations is that the best way to evaluate

[6]Of course, the texts of the intra-register data are not ergodic either. The deviations are however less important.

the performance of a probabilistic model remains an extrinsic one, i.e. one which takes the specific task into account (also called *'in vivo'* evaluation; cf. Spärck-Jones & Galliers, 1996). Intrinsic measures like cross entropy can be very helpful in optimization, since it can be assumed that a model yielding a lower cross entropy than another one will also yield better performance results, but extrinsic measures (like ksr) will still be more precise for the given task at hand.

3.2 Stochastic language models - Background

So far a family of models was presented that all make the same assumption, namely that the occurrence statistics of a symbol depends on a limited number of other symbols (i.e. that the context to be taken into account has finite length). The idea to constrain the context length in order to calculate symbol probabilities is attributed to Andrei Markov's work, who in the beginning of the 20th century developed the mathematical basis to predict upcoming symbols from symbol chains of fixed length (cf. Markov, 1913); for this reason the just mentioned assumption as well as the whole class of such processes today bear Markov's name.

It was also mentioned in the previous section that the *Markov assumption* does not hold for natural language. There is no doubt that the structure of natural language cannot be described as a finite-state Markov process; our own intuition already tells us that every sentence that we utter has a deeper structure than a simple linear left-to-right alignment of words. Relative clauses can be recursively embedded into other relative clauses and there is no constraint on how many (attributive) adjectives can be attached to a noun. Moreover, there is no evident relation between the grammatical correctness of a sentence and its frequency in a corpus of any given size. For example here may be the first and only time to see the 2-gram *"vanilla chair"*, but, apart from the fact that nearly everything can be shaped by vanilla icecream, it can also refer to the chair's color or (figuratively) to a plain and simple chair without further scrolls. It is a lively and crucial property of natural language that an expression which has never been seen or uttered before, can be understood (and perceived as grammatical) without any effort, and it is obvious that this cannot be modeled by a Markov process.

This observation was already made by Noam Chomsky in his seminal paper from 1956, where he convincingly shows that a natural language expression has to be described by other means than by a Markov model. He proposes (and formalizes) the notion of a phrase structure grammar, consisting of a set of rules that are iteratively transformed in order to derive all possible sentence structures for a given language (and only these). He also describes the expressivity of a (formal) language by the kind of rules from which they are generated. In this way he creates a hierarchy of language types, of which the most restricted ones are the languages which can be generated by a Markov source (*type 3* or *regular* languages), and the most general ones are those which can be generated by a Turing machine (*type 0* or *recursively enumerable* languages). The structure of natural language can best be described by a *type 2* (context free) or *type 1* (context sensitive) formalism, but simply not by a type 3 grammar which can only describe linear dependencies.

Chomsky's formalizations can be seen as the founding stone of modern linguistics, and they had an important influence on theoretical computer science as well. Since Chomsky's first observations (cf. Chomsky, 1956), a large number of elaborate grammar formalisms and linguistically motivated models have been developed. Apart from Chomsky's works on *Generative Grammar*, we find very elaborate formalisms such as the *Lexical-Functional Grammar* (LFG; cf. Horn, 1983; Bresnan, 2001), *Head-Driven Phrase-Structure Grammar* (HPSG; cf. Pollard & Sag, 1994) or *Tree-adjoining Grammar* (TAG; cf. Joshi *et al.*, 1975).

Considering their descriptive adequacy one would expect these formalisms to describe language much better than Markovian approaches. However, we have to recognize the disturbing fact that more than 50 years after it has been shown that natural language is not regular many successful models in natural language processing are still based on Markov-style formalisms (cf. Rosenfeld, 2000; Goodman, 2000) or at least combine them with other techniques. N-gram models can be seen as the working horse of a wide range of NLP applications, especially in those domains that deal with language generation (in its broadest sense), for example in:

- Speech recognition, (cf. Jelinek, 1990; Hruska *et al.*, 2000)
- Machine translation, (cf. Brown *et al.*, 1990; Brants *et al.*, 2007)
- Optical character recognition, (cf. McQueen & Mann, 2000)
- Spelling correction, (cf. Kukich, 1992; Brill & Moore, 2000)

This puzzling mismatch between theoretical conviction and applicative success has not yet been sufficiently discussed, but it is probably due to a difference in perspectives: linguistic theories mostly care to decide grammaticality (i.e. model linguistic *competence*). From this perspective a given structure is considered as either grammatical or ungrammatical, and not as more or less probable.

NLP applications on the other hand try to model the language *performance* of humans, where structural errors occur frequently; at the same time many actually grammatical constructions do not occur, due to cognitive limitations (e.g. normally not more than three relative clauses will be nested). Performance issues have long been treated as the ugly duckling in modern linguistics; notions of probability have been regarded as useless with respect to the description of language structure.[7] However, it has to be acknowledged that

[7](cf. Chomsky, 1969, p. 57): "It must be recognized that the notion of a 'probability of a sentence' is an entirely useless one, under any interpretation of this term."

many linguistic phenomena can only be described (and explained) by referring to probability; structures can be unusual or preferred, and determining grammaticality does not help in this distinction.

Apart from this paradigmatic gap, another reason for the tremendous success of stochastic language models is their underlying simplicity. Compared to many linguistic theories, whose degree of complexity is not only intellectually challenging but also poses computational problems, it seems very easy to build a model from nothing but the frequencies of word sequences. N-gram frequencies (once they are determined on a training corpus) make the calculation of a word's probability, given a context, directly available. We simply need to divide the number of times we have seen the context by the number of times we have seen the context followed by the word. This first (and quite intuitive) way to estimate word probabilities is called *Maximum likelihood estimation* (MLE), and it is formalized as follows ($C(w_1, \ldots, w_m)$ refers to the frequency of sequence $w_1, \ldots, w_m$ in a given training corpus):

$$P_{MLE}(w_i|w_1, \ldots, w_{i-1}) = \frac{C(w_1, \ldots, w_i)}{C(w_1, \ldots, w_{i-1})} \tag{3.11}$$

The question that arises immediately from this equation is how long the size of the context should be. As already explained in section 3.1.2 the accuracy of the probability estimate grows with the length of the context that we take into account. This can be determined by the reduction of cross entropy as well as by performance-oriented measures such as the *keystroke saving rate*. The numbers in Table 3.4 have been calculated using models of a context size from 0 to 3 words that have been trained on the already mentioned French newspaper corpus of 44 million words from *Le Monde* (vocabulary size: 141.078 types), the test corpus comprised 58.000 words.

	ksr_1	ksr_5	cross entropy	redundancy	perplexity
0-gram	10,4%	20,3%	17,1	0%	141078
1-gram	29,3%	46,3%	9,6	43,7%	795,2
2-gram	41,5%	55,7%	7,3	57,1%	162,5
3-gram	43,8%	57,6%	6,9	59,8%	116,3
4-gram	44,3%	57,8%	6,8	60,4%	109,7

Table 3.4: keystroke savings, cross entropy, redundancy and perplexity for n-gram models of varying n

The results displayed in Table 3.4 confirm Shannon's observations (cf. Shannon, 1951), and they are in straight accordance with the results given of Goodman (2001), who tested models for n up to 20: the larger the context the lower the cross entropy of the sequence.

However, as the context size grows bigger we also face an increasing problem of data sparsity. For a vocabulary of size s we have to estimate frequencies for s unigrams, s^2 bigrams, s^3 trigrams and s^4 four-grams, most of which will never occur in even the largest imaginable corpora[8].

So, many of the frequencies on which our probability estimation is based, are zero, especially when we want to calculate a model of higher n. But what happens if we apply the maximum likelihood estimate to calculate a word's probability when this word has never occurred with the given context (remember the "vanilla chair" example)? The MLE assigns zero probability to all events that have not been seen in training; as its name indicates, it "maximizes" the likelihood for seen events, it adapts perfectly to the training data. But this is not what we want, because we will not work on the training corpus but on foreign data, and there a word might still be able to occur after a given context, even though it was never seen before. Estimating the probability of unseen events is however a non-trivial task, and many sophisticated methods have been developed to achieve this. Modifying the observed estimates in order to achieve a more appropriate probability distribution is usually referred to as *discounting* or *smoothing*. While the term *'discounting'* indicates that the real counts are reduced in favor of unseen word sequences, *'smoothing'* refers figuratively to the shape of the probability distribution graph which is rounded off by such methods.

The following section introduces a number of smoothing (or discounting) methods, starting from the first considerations (e.g. *Laplace* smoothing) to the state of the art in this domain, which is currently assumed by *Kneser-Ney* smoothing.

3.3 Treating unseen events - Smoothing methods

The probability estimation for unseen events has been shown to be a crucial factor in statistical language modeling. In the last 20 years an important number of smoothing paradigms have been developed, such as those proposed by Jelinek & Mercer (1980), Katz (1987), Ney & Essen (1991), Witten & Bell (1991) or Kneser & Ney (1995), of which only a few will be presented here. Longer introductions to this subject can be found in the language modeling chapters of (Manning & Schütze, 1999, ch. 6) and (Jurafsky & Martin, 2000, ch. 6); detailed descriptions and empirical comparisons are given in the works of Chen and Goodman 1999; 2001. To simplify notation all smoothing schemes are given for a trigram model, but the formulas are easily adaptable to models of higher or lower n.

[8]The (to my knowledge) currently largest n-gram models have been calculated by the *Google* machine translation group; they are trained on web corpora of up to two trillion (i.e. 2×10^{12}) tokens and comprise 1- to 5-grams, (cf. Brants *et al.*, 2007). To be found in the *LDC Corpus catalog*: http://www.ldc.upenn.edu/Catalog/.

3.3.1 Additive discounting

The oldest solution to deal with unseen events in statistics can be related to the works of Pierre-Simon Laplace in the beginning of the 19th century. According to Laplace's law, all counts are simply raised by one, before they are transformed into probability estimates, as can be seen in the following equation:

$$P_{Laplace}(w_i|w_{i-2}w_{i-1}) = \frac{C(w_{i-2}w_{i-1}w_i) + 1}{C(w_{i-2}w_{i-1}) + |V|} \tag{3.12}$$

where $|V|$ is the size of the vocabulary. This technique is simple and rather intuitive, but can alter the original probability distribution quite strongly, especially if we have a high number of unobserved sequences. In practice, Laplace smoothing does not yield good results, it *over-smoothes* the probability distribution.

A first solution to this problem was proposed by Lidstone (1920), who generalized Laplace's formula, so that not exactly one, but smaller (positive) values δ could be added:

$$P_{Lidstone}(w_i|w_{i-2}w_{i-1}) = \frac{C(w_{i-2}w_{i-1}w_i) + \delta}{C(w_{i-2}w_{i-1}) + \delta|V|} \tag{3.13}$$

A commonly applied value for δ is $0,5$; such a fractional δ improves the probability estimates significantly. However, it is still far from optimal, especially for low-frequency counts probabilities are wrongly estimated. Nowadays, additive discounting methods are not applied in language modeling, however their simplicity offers an intuitive approach to the general idea of what smoothing tries to achieve.

3.3.2 Combining models: Backing-off and interpolation

A much more clever way to overcome the sparse data problem is to combine the information from several models. It was already made clear that a model of higher order gives better estimates, but at the same time the number of unseen events raises exponentially with n. It is therefore a good idea to consult a model of order $n-1$, if the model of order n cannot help for a given sequence (because its count is zero). This idea is usually referred to as *backing off* (cf. Katz, 1987); it can be formalized as follows:

$$P_{BO}(w_i|w_{i-2}w_{i-1}) = \begin{cases} \frac{C^*(w_{i-2}w_{i-1}w_i)}{C(w_{i-2}w_{i-1})} & \text{if } C(w_{i-2}w_{i-1}w_i) > 0 \\ \alpha_{w_{i-2}w_{i-1}} P_{BO}(w_i|w_{i-1}) & \text{otherwise} \end{cases} \tag{3.14}$$

and for the second step of the recursion[9]:

$$P_{BO}(w_i|w_{i-1}) = \begin{cases} \frac{C^*(w_{i-1}w_i)}{C(w_{i-1})} & \text{if } C(w_{i-1}w_i) > 0 \\ \alpha_{w_{i-1}} P_{BO}(w_i) & \text{otherwise} \end{cases} \tag{3.15}$$

As can be seen in equations 3.14 and 3.15, the algorithm recursively applies models of a shorter and shorter history until $n = 1$. The lower order models are however only taken into account if the higher order model cannot be applied. C^* represents an already smoothed count of the n-gram under consideration, because such a backoff scheme is normally applied in combination with a standard smoothing scheme. The coefficient α is a normalizing factor (referred to as *backoff weight*), which assures that the sum of all probabilities remains unchanged.

Another way to combine information from several models is *interpolation*. Whereas in a backoff scheme we rely solely on the information given by the higher order model (if it can be applied), interpolation always mixes the estimates from all the models. In (simple) linear interpolation we therefore weight each model by a coefficient λ_i and then add the single estimate.

$$\begin{aligned} P_{LI}(w_i|w_{i-2}w_{i-1}) \quad = \quad & \lambda_1 \cdot P^*(w_i|w_{i-2}w_{i-1}) + \\ & \lambda_2 \cdot P^*(w_i|w_{i-1}) + \\ & \lambda_3 \cdot P^*(w_i) \end{aligned} \tag{3.16}$$

where $0 \leq \lambda_i \leq 1$ and $\sum_i \lambda_i = 1$. This assures that the combined probabilities also sum up to 1; P^* is meant to be any kind of estimate, it can be the MLE, but also some smoothed form. There are also other ways to interpolate models (e.g. *geometric* interpolation, cf. section 5.4.3 later on), but for the moment linear interpolation is a sufficient means to combine information from several models.

Note that most state-of-the-art smoothing techniques can be expressed in a backoff and an interpolated variant. Thereby many interpolated models seem to have slight advantages over the backoff versions (cf. Chen & Goodman, 1999). However a tricky problem in interpolation is given by the optimal estimation of the coefficients λ_i. This will be discussed in the following section (3.4).

3.3.3 Absolute discounting

The underlying idea of *absolute discounting* (Ney & Essen, 1991; Ney *et al.*, 1994) is to subtract a small constant amount from all MLE counts, based on the in-

[9]This step is analogous to the first, and it is only displayed to make clear the recursive nature of backoff algorithms. For further algorithms using backoff (e.g. 3.3.3), only the first step is presented.

tuition that high (and more reliable) counts are less affected than low counts, which are not that trustworthy anyway. Absolute discounting also makes use of lower order models, either in a backoff or in an interpolated form. The following equation displays the back-off variant of absolute discounting:

$$P_{abs}(w_i|w_{i-2}w_{i-1}) = \begin{cases} \frac{C(w_{i-2}w_{i-1}w_i)-D}{C(w_{i-2}w_{i-1})}, & \text{if } C(w_{i-2}w_{i-1}w_i) > 0 \\ \alpha(w_{i-2}w_{i-1})P_{abs}(w_i|w_{i-1}), & \text{otherwise} \end{cases} \tag{3.17}$$

where $0 \leq D \leq 1$. In (Ney *et al.*, 1994) the authors also suggest an optimal setting of D:

$$D = \frac{|C_1|}{|C_1| + 2 \cdot |C_2|} \tag{3.18}$$

where $|C_1|, |C_2|$ are the numbers of n-grams with exactly one and two counts in the given training corpus.

Despite its relative simplicity, absolute discounting has been shown to perform remarkably well; it could outperform other, more complex techniques.

3.3.4 (Modified) Kneser-Ney discounting

Kneser-Ney discounting (Kneser & Ney, 1995) is based on the idea of absolute discounting, in that it also uses a constant discount. However it takes advantage of another idea: Words (or lower-order n-grams) that appear in many different contexts (i.e. higher-order n-grams) are more likely to appear in new contexts than those appearing always in the same construction. This applies especially for proper nouns (e.g. *The Hague*, *Leonardo da Vinci*), but also for idiomatic expressions (e.g. *"shoot the breeze"*). *Hague* (which is not so uncommon in newspaper text) should receive a high estimate only if it follows *The*. For all other contexts its probability should be estimated lower; a (similarly frequent) noun like *sparrow*, occurring in many different contexts should however receive a higher estimate. This intuition can be formalized by calculating a *continuation probability*, which is based on the number of distinct contexts h a word w_i appears in ($|\{h : C(hw_i) > 0\}|$).

$$P_{KN}(w_i|w_{i-2}w_{i-1}) = \begin{cases} \frac{C(w_{i-2}w_{i-1}w_i)-D}{C(w_{i-2}w_{i-1})}, & \text{if } C(w_{i-2}w_{i-1}) > 0 \\ \alpha(h)P_{Cont}(w_i|h), & \text{otherwise} \end{cases} \tag{3.19}$$

where α is again a normalizing factor to assure that the resulting estimates are sum up to 1. The continuation probability is then calculated as follows:

$$P_{cont}(w_i|h) = \frac{|\{h : C(hw_i) > 0\}|}{\sum_w |\{h : C(hw) > 0\}|} \tag{3.20}$$

The denominator represents the total number of contexts beginning with h; it is needed to form a probability-like estimate.

In a large-scale evaluation campaign Chen & Goodman (1999) showed Kneser-Ney smoothing to consistently have the best performance over many different techniques. However they also observed that this method does not perform optimal for low frequency n-grams. For this reason they propose a slight modification of the discounting scheme: Instead of using a single discounting factor D for all counts, they apply three different parameters D_1, D_2 and D_{3+} that are applied to n-grams with one, two and three or more counts, respectively. In (Chen & Goodman, 1999) this technique, which is referred to as *modified Kneser-Ney smoothing*, has scored even better than standard *Kneser-Ney*, and we are unaware of any other smoothing technique with higher performance.

3.4 Parameter estimation and practical issues

3.4.1 The EM algorithm

Finding the optimal parameters for a given algorithm is a crucial (and non-trivial) task. Often this is solved by some empirical pre-testing until the results seem somehow optimal. Empirical testing is not the worst approach, and sometimes it is the only possible way, however it is usually tedious, and there is no way to guarantee that the applied parameter values are (even locally) optimal. For this reason a number of algorithms was developed that provide automated solutions. In probabilistic frameworks the most prominent one is the *expectation maximization* (EM) algorithm (cf. Dempster *et al.*, 1977; Jelinek, 1990). It is a greedy algorithm, i.e. it takes the shortest path in the search space. This also means that it cannot rule out the possibility of getting stuck within a local maximum, however it can be shown that the algorithm always converges (under certain assumptions), moreover it is fast and quite easy to apply. But before explaining the algorithm, we have to define the goal of our optimization task[10]. Our goal is to minimize the cross entropy for a probability distribution which is combined from several singular models; our optimization parameters are the coefficients $\lambda_{1..m}$:

$$\underset{\lambda_{1..m}}{\arg\min}\, H = -\frac{1}{N}\log(\sum_{i=1}^{m} \lambda_i P_i(w_1, \ldots, w_N)) \tag{3.21}$$

[10]In what follows, the EM algorithm is not presented in its most general form; it is shaped onto our task of optimizing coefficients for the interpolation of language models. For further background information cf. (Dempster *et al.*, 1977; Berger, 1998)

The algorithm is then constructed as follows:

```
EM algorithm

Initialization: Pick some random starting values for $\lambda_{1..m}$, e.g. $\lambda_i = \frac{1}{m}$
(the values have to be larger than zero)

Repeat
    Expectation step: Compute the expected outcome $e$ for each $\lambda_i$

        $e_k(\lambda_i) = \frac{\lambda_i^k p_i(w_j|h)}{\sum_{j=1}^{m} \lambda_i p_i(w_j|h)}$    (3.22)

    Maximization step: Compute new $\lambda_i$ for the next iteration $k+1$

        $\lambda_i^{k+1} = \frac{e_k(\lambda_i^k)}{\sum_{j=1}^{m} e_j(\lambda_j)}$    (3.23)

until $\lambda^k \approx \lambda^{k+1}$ (exit if the changes fall below a given threshold)
```

The rationale of this algorithm is to measure the average success of each of the single models with respect to the combined model and to adapt the coefficients at each step accordingly. In this light it becomes also clear why the starting values have to be greater than zero, otherwise the corresponding models cannot contribute anything to the interpolation, their share of the success remains zero as well.

3.4.2 Size reduction: Pruning

As already explained in the introduction to section 3.3, statistical language models of higher order necessarily include a sparse data problem. This also implies that they are the more successful, the more data they have been trained on. This assumption is confirmed by the results of (Chen & Goodman, 1999) and (Goodman, 2001), who tested training corpora of different sizes (up to 284 million words). Language models trained on tens or hundreds of millions of words however attain a considerable size themselves, often making them impractical for real applications, which normally have memory constraints. From a practical point of view, the reduction of a model's size is therefore an important issue.

The most straightforward idea to reduce the size of a model is to eliminate (or *prune*) those parts which are not reliable anyway. Reliability in turn is obviously related to frequency: The less times we have seen the instance of an n-gram the worse our probability estimate will be. A natural conclusion would therefore be to exclude low frequency n-grams. Seymore & Rosenfeld (1996) showed for example that the size a trigram model trained on 45 million words could be reduced from 104MB to 29MB (a reduction of 72%) by simply eliminating all n-grams occurring one time only. But it is evident that this is rather brute-force; Seymore & Rosenfeld (1996) therefore propose a more elaborate method: If in a backoff model a higher-order and a lower-order estimate are present, and if the lower-order estimate (to which the model backs off) is very similar to the higher-order estimate, there is no need to store the latter, it can be pruned. This "*weighted difference*" method could be shown to result in a slightly lesser perplexity increase than plain frequency cutoff.

An even more sophisticated pruning method was developed by Stolcke (1998) (hence known as *Stolcke pruning*). It is based on the relative entropy change caused by removing an n-gram. The relative entropy (or *Kullback-Leibler distance*) D between two models P_A and P_B can be calculated as follows:

$$D(P_A||P_B) = -\sum_{w_i,h_j} P_A(w_i,h_j)[\log P_B(w_i|h_j) - \log P_A(w_i|h_j)] \tag{3.24}$$

where the summation is over all words w_i and all contexts h_j. For the task of pruning P_A represents the original model and P_B the reduced one, and the goal is to find a reduced model P_B so that $D(P_A||P_B)$ is minimized.

Stolcke (1998) now proposes the following pruning algorithm:

Stolcke pruning algorithm

1. Select a threshold θ (usually between 10^{-10} to 10^{-6})
2. Compute the relative perplexity increase due to pruning each n-gram individually
3. Remove all n-grams from the model that raise the perplexity by less than θ
4. In case of a backoff model: recompute backoff weights

This kind of pruning has shown remarkable results: Stolcke (1998) reports of a language model whose size was reduced by 74% without any error increase in a speech recognition task.

3.4.3 Data formats and toolkits

With the growing size of language models their representation and storage format has become an important issue. A *de facto* standard for representing LMs is nowadays the *ARPA* backoff format. It simply lists all n-grams from the lowest to the highest order, together with their discounted (logarithmized) probabilities and the corresponding backoff weights in a standard ASCII file (cf. also Figure 3.3):

$$\textbf{ARPA backoff format:} \qquad \log P^*(w|h) \qquad w\ h \qquad [\log \alpha(h)]$$

Storing log probabilities has an important advantage: As n-gram probabilities tend to be multiplied in many applications, they become smaller and smaller and can finally cause a numerical underflow. Since adding in logarithmic space is equivalent to multiplying in linear space, log probabilities can simply be added, which avoids the underflow problem.

```
\data
ngram 1=133530
ngram 2=1094794
ngram 3=6229280

\1-grams
-1.642683        a                -1.022194
-6.866025        aardvark         -0.229095
-6.610753        abacus
   ...
\2-grams
-6.378439        a as
-3.081988        a baby           -0.060502
-4.770818        a babysitter     -0.072190
   ...
\3-grams
-1.303144        a as in
-2.994654        a baby ant
-0.984528        about all other
   ...
```

Figure 3.3: Example of a language model in ARPA backoff format

N-gram language models stored in the ARPA format can apply any kind of smoothing strategy; interpolated models can be expressed in terms of a backoff format as well. In this case the probabilities of higher order n-grams simply take into account the corresponding lower-order n-gram, there is no need to recompute the values on the fly.

The ARPA format is also respected by the two most commonly used language modeling toolkits: The *Carnegie-Mellon* (CMU) toolkit[11] (Clarkson & Rosenfeld, 1997) and the SRI toolkit[12] (Stolcke, 2002). Both are publicly available for non-commercial use[13], and they offer a wide range of smoothing methods and other functionality such as pruning and parameter estimation. They also include a testing mode, in which they calculate the perplexity of a given test corpus on a previously calculated model.

[11]http://www.speech.cs.cmu.edu/SLM/toolkit.html

[12]http://www.speech.sri.com/projects/srilm/

[13]For which we are very grateful to the authors!

3.5 Linguistically motivated models

In the beginning of this chapter (3.2) it was already discussed that language models of the n-gram family (thus representing finite-state Markov processes) are intrinsically unable to describe the full structure of natural language. At the same time it was made clear that the modeling techniques we have looked at so far, do not make use of any explicit linguistic knowledge. A word is regarded as any symbol, its linguistic properties are neither analyzed and nor used for the probability estimation. Moreover, due to the already described combinatory explosion we encounter when estimating probabilities of all possible n-grams, the statistics we rely on are extremely short-sighted. Consider the following example (from Brill *et al.*, 1998): If we want to calculate the probability of *'barked'* in *"The dog on the hill barked"*, a 4-gram model would consider nothing but: $P(barked|on\ the\ hill)$. It is obvious that this results in a wrong estimate, mainly because the subject-verb dependency (*the dog-barked*) is not captured. But even if we could calculate a 6-gram-model, it would not help us, if the dog barked *on the little, grassy hill.* A more adequate analysis (or *parse*) of this sentence might look as displayed in Figure 3.4.

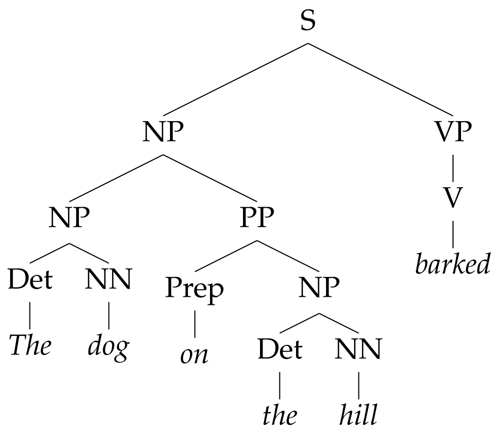


Figure 3.4: A possible parse of the sentence *"The dog on the hill barked"*

Theoretically, a model including such an analysis could estimate the probability of *'barked'* more properly. However, parsers based on the already mentioned grammar formalisms could not (yet?) be successfully applied in task-oriented NLP, mainly due to three major deficiencies:

- *Algorithmic complexity*: A task like speech recognition or word prediction (as well as other AAC tasks) is particularly sensitive to time; the user is not willing to wait even a second for the recognition or prediction to be accomplished. Therefore a complex parsing algorithm (which does usually not perform in linear time) cannot be applied.

- *Parse ambiguity*: The syntactic analysis of many phrases turns out to be highly ambiguous, i.e. often several parses are valid for a given sentence.

This poses a serious problem for further processing the information provided by the analysis: If all parses are considered the complexity problem aggravates and many possible but improbable structures dilute the results; blindly discarding parses is however risky, since the most appropriate analysis may be among them.

- *Robustness*: Current parsing algorithms have severe problems with the treatment of partial structures (sentence beginnings), unknown words or simply ill-formed sentences that occur quite often in normal conversation (possibly including postponed corrections); if an unknown (or wrong) construction is encountered, most algorithms do not return anything, not even a partial parse of the already recognized structures. This is usually fatal in a real-world application, especially in an AAC context, where the user's possibilities to deal with such a "blank screen"-situation are highly reduced.

For these reasons, since the early 1990s many research efforts have focused on the development of *shallow* parsing algorithms that offer an uncomplete but (rather) quick and more robust analysis (cf. for example Abney, 1991; Aït-Mokhtar *et al.*, 2002; Basili & Zanzotto, 2002). Conversely one attempted to enrich statistical language models with linguistic information. One of the first attempts in this perspective was to rely on Part-of-Speech information (n-PoS). A more advanced model is Abney's (1991) *chunk* parsing strategy and Chelba and Jelinek's *Structured Language Model* (1997; 1998; 2000), relying on a partial analysis of the constituent structure. Finally, *probabilistic context-free grammar formalisms* (PCFGs) are able to determine the recursive structure of a sentence, while they are more robust than non-probabilistic parsing algorithms. In the following these models are shortly presented.

3.5.1 Part-Of-Speech and class-based models

A straightforward way to introduce linguistic knowledge is to build a language model on n-grams of parts-of-speech rather than on the words themselves. This reduces the parameter space tremendously, since tens of thousands of singular words are mapped to a rather small number (app. 30-100) of equivalence classes. The same can be achieved by automated clustering techniques, such as *IBM clustering* (cf. Brown *et al.*, 1992), where each word is assigned to a cluster or class, based on its distributional properties. Approaches based on clustering instead of manually coded parts-of-speech are usually referred to as *class-based* models, the formalization however is the same. The probability of a word w_i, given a context h_i is calculated from the probability of w_i belonging to a class c_i and the probability of c_i after having seen the $n-1$ previous classes:

$$P(w_i|h_i) = \sum_{c_i \in B(w_i)} P(w_i|c_i) \times P(c_i|c_{i-n-1}, \ldots, c_{i-1}) \qquad (3.25)$$

The probability of a word w_i belonging to a class c_i can be determined from the maximum likelihood estimate of w_i given c_i; likewise we can use the MLE to calculate the conditional probability of c_i given $c_{i-n}, \ldots, c_{i-1}$:

$$P(w_i|c_i) = \frac{C(w_i, c_i)}{C(c_i)} \tag{3.26}$$

$$P(c_i|c_{i-n}, \ldots, c_{i-1}) = \frac{C(c_{i-n-1}, \ldots, c_{i-1}, c_i)}{C(c_{i-n-1}, \ldots, c_{i-1})} \tag{3.27}$$

Note that the formula in 3.25 applies to the general case, where a word can belong to several classes ($B(w_i)$ being the bin of all classes that w_i belongs to). This is obviously true for parts-of-speech; many words can be assigned to more than one linguistic category; for example a word like *'blind'* can be an adjective, a noun and a verb. In such cases we have to calculate the estimates separately for each possible category. The situation becomes easier when a word is allowed to belong to one class only (so-called *hard clustering*). The already mentioned technique of Brown *et al.* (1992) follows such a hard clustering approach; their clustering algorithms allow for a syntactic as well as a semantic regrouping of words.

The intention of using classes instead of words lies foremost in the reduction of the parameters to be estimated. It is obvious that with a smaller parameter space the length of the history can again be extended; 7-grams or even higher order n-grams over parts-of-speech then become imaginable. An interesting approach in this direction has been presented by Niesler & Woodland (1996) who developed a variable-length n-gram model over parts-of-speech (with n up to 10). The size of the history is increased until the estimates of the models at order n and $n + 1$ converge.

While the length of the history can be extended, a class-based model is surely not a linguistically adequate model in that it still does not explicitly represent constituent structure; moreover such models lose quite some of the semantic information which is inherent to word-based models. Typical collocations or selectional preferences of verbs for example are not modeled anymore: P(*barked*|*the dog*) will certainly give a more accurate estimate than P(*barked*|V) $\times$ P(V|Det NN).

For this reason, most approaches make use of a combination of word- and class-based models. Fazly & Hirst (2003) for example interpolate a tag trigram model with a word bigram model for the task of word prediction. The gains of the combined model were however not very large with respect to the word-based bigram model alone(+0,9% in ksr).

3.5.2 Chunk-based and partial-parse language models

The idea of chunking is related to the works of George A. Miller (1956), who discovered that the short-term memory of people does not depend on the number of raw symbols (e.g. numbers or characters), but on the number of perceptual units. Once such a unit is memorized, it can comprise a (nearly) arbitrary number of symbols, and it can as easily be recalled as a singular symbol. Miller concluded that human perception is largely making use of chunking, i.e. people perceive chunks of features rather than the features themselves. With regard to language perception (and production) this means that not single words are processed but phrasal constituents. A number of psycho-linguistic experiments support this idea, for example it can be shown that people, who mispronounced a word, tend to repeat the whole constituent (e.g. *"I swear to swell, uhm, **to tell** the truth."*).

Whereas the idea of chunking cannot yet account for the full structure of natural language, it is still much more plausible than that of a simple word-based Markov process. Furthermore, chunks are normally far easier to determine than the full parse of a phrase. In (Abney, 1991) a chunk (or partial) parsing strategy was proposed, and many other chunk parsers have been developed since.

Schadle *et al.* (2004) have presented a word prediction model based on the idea of chunk parsing. Instead of considering the last words for prediction they propose a model based on the last *chunk heads* (i.e. the grammatically dominating items), derived from a partial parse of the previous constituents. Returning to our example from above a chunk parse of *"The dog on the hill barked"* would look as follows (the heads are printed in bold face):

$$[\textit{the } \textbf{\textit{dog}}]_{NP}[\textbf{\textit{on}} \textit{ the hill}]_{PP}[\textbf{\textit{barked}}]_{VP}$$

It is obvious that such a structure helps to better predict *'barked'*, since *'dog'* (being the best predictor for *'barked'*) is the head of the first chunk. Their model comprises three major components: a *tagger* assigning parts-of-speech to the previous words, a *chunker* providing a chunk segmentation of the previous constituents (including the chunk heads), and a *predictor*, which uses the categories of the previous $n-1$ chunks as well as their heads to predict the following chunk.

The *Structured Language Model* by Chelba & Jelinek (1998, 2000); Xu *et al.* (2002) goes a step further in that it joins the chunks to larger constituents, i.e. its prediction is based on a (binary) partial parse of the sentence beginning (Charniak's 2001 *immediate head* strategy is very similar). For our example such a partial parse tree before the prediction of *'barked'* would look as depicted in Figure 3.5 (the respective heads are given as indices to the constituent labels).

Just as the model of Schadle *et al.* (2004), the SLM consists of three modules:

- a *predictor* that predicts the following word w_i, given the word-parse prefix.
- a *tagger* that predicts the PoS-tag of the following word, given the current word-parse prefix.

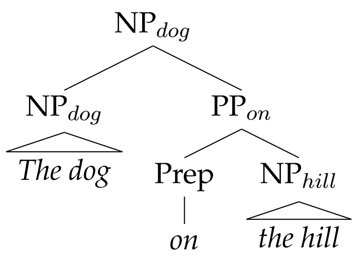


Figure 3.5: A partial parse tree for "The dog on the hill "

- a *parser* that continues the parsing process at each step by either generating a unary non-terminal label or performing a binary (left or right) *join* operation to connect two constituents.

So, at every step the algorithm performs three actions: Estimating the probability of the following word w_i and its tag t_i, and the next parsing operation k_i.

The chunk-based model as well as the SLM showed a slight but significant performance improvement (with respect to a trigram baseline), however some difficulties could not yet be addressed in a satisfying way. Since these models estimate probabilities on complex entities which are harder to observe than plain words, optimal parameter estimation and smoothing strategies for these models have not yet been developed. Further research would be necessary in order to exploit the full potential of these approaches.

3.5.3 Probabilistic context-free grammars

Context-free grammar (CFG) formalisms are a common way to describe linguistic structures. Whereas their expressive power is still not sufficient to describe all linguistic phenomena (this is only achieved by *mildly context-sensitive* formalisms), they cover at least a large part of them. For example the tree structure over our example sentence given above can be described in terms of a context-free grammar. CFGs consist of a (finite) set of *terminal* symbols T, normally representing the words of a given vocabulary, a set of (again finite) *nonterminal* symbols N , corresponding to the higher-order linguistic categories and a set of transformation (or rewrite) rules of the form "$A \rightarrow \gamma$, where $A \in N$ and $\gamma \in (N \cup T)^*$, i.e. whereas the left-hand side of the rule has to be a single non-terminal symbol, the right-hand side can consist of any sequence of terminals and non-terminals.

The probabilistic counterpart of CFGs includes additionally a probability estimate for every rule. These estimates have to sum to 1 at each choice point (the left-hand side of a rule; i.e. $\forall i \sum_j P(N_i \rightarrow T_j) = 1$). The overall probability of a parse is the product of all applied rule probabilities. A small probabilistic context-free grammar being able to generate our example sentence *The dog on the hill barked* could look as displayed in Table 3.5, the tree in Figure 3.6 shows a parse of the example, based on this grammar.

S → NP VP	1,0	Det→ *the*	0,5
NP → Det NN	0,8	Det→ *a*	0,5
NP → NP PP	0,2	NN→ *dog*	0,7
PP → Prep NP	1,0	NN→ *hill*	0,3
VP → V NP	0,5	Prep→ *on*	0,6
VP → V PP	0,2	Prep→ *under*	0,4
VP → V	0,3	V→ *barked*	1,0

Table 3.5: An example for PCFG transformation rules

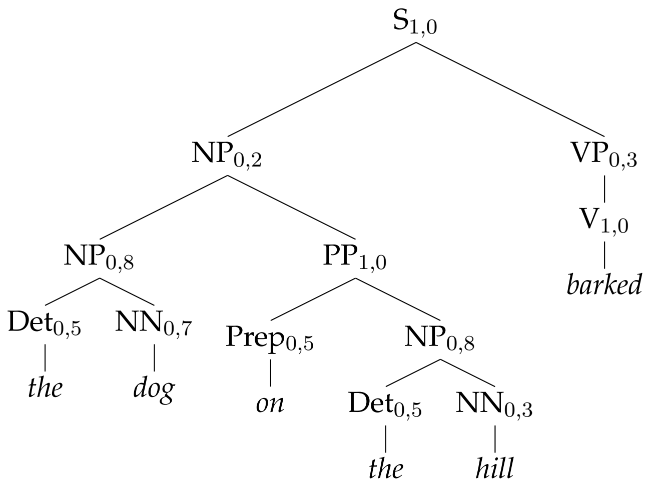


Figure 3.6: A possible PCFG parse of the sentence *"The dog on the hill barked"*

As opposed to CFG parsing formalisms, which also suffer from the already described problem of robustness, a PCFG has an important advantage: A structure which could not be parsed is not simply ruled out or leads to a parse error, it is only assigned a low probability (in practice PCFG probabilities are also *smoothed*). Another advantage is that such a framework can deal with the already mentioned syntactic ambiguity (e.g. *"She saw the rabbit with the binoculars"*), in that all possible parses can be ranked according to their probability.

An important problem however remains in the complexity of parsing: Such grammars have the same complexity as normal CFGs, so the parsing process is still rather slow. Another difficult issue is the estimation of the rule probabilities, which is not as straightforward as for n-grams. This is normally done by applying the *Inside-Outside algorithm*, an EM-style algorithm (3.4.1) that iteratively adapts the rule probabilities according to the number of times they are applied during training. But, as already mentioned, EM algorithms can get stuck within local maxima, and the probability surface of such models is prone to contain many of them, so in many cases a suboptimal estimate is calculated.

PCFGs could not be shown to provide a useful language model alone, but some works have reported interesting gains when a PCFG is interpolated with a word-based n-gram model (cf. Moore *et al.*, 1995; Roark, 2001).

3.5.4 A special problem: Treatment of compounds

After a short overview on general linguistic formalisms we now focus on a particular problem concerning the word level. All models we have considered so far assume (at least implicitly) that a word consists of a sequence of characters between two empty spaces or punctuation signs. At the same time they assume that the amount of different words, i.e. the size of the vocabulary, is more or less finite. Unfortunately this does not hold for some languages (such as Dutch, Swedish, Greek and German): They allow words to agglutinate so that compound words of sometimes impressive length can be formed. While this formation process respects certain rules, it is productive, i.e. valid compounds can be formed instantaneously, and the number of possible compounds is (theoretically) infinite. Table 3.6 shows a few examples from German (since German is one of the languages treated in this work):

German word/compound	Translation	Frequency
Wort	'word'	39.241
Wortvorhersage	'word prediction'	0
Wortvorhersagemodul	'word prediction component'	0
Wortvorhersagemodulentwicklung	'development of a word prediction component'	0

Table 3.6: German compounds and their frequencies in a 120 million word newspaper corpus (from: *Die Tageszeitung* 1989-1998)

When we look at the frequencies of the compound words in Table 3.6, the challenge that these words mean for an NLP application becomes apparent immediately. Since many compounds are created on the fly, they never appear in even large corpora. And this phenomenon is not marginal: Baroni *et al.* (2002) report from an analysis of a large German newswire corpus (APA) in which nearly half (47%) of all the word types were compounds; most of them (83%) had a very low frequency (≤ 5). The analysis of unknown words (out-of-vocabulary words) in the following chapter confirms this observation: More than half of the unknown words (51%) of a German newspaper test corpus were compounds (cf. chapter 4, Table 4.6). So, when we deal with a language that allows compounding, this phenomenon has to be addressed in some way, since a word not occurring in any vocabulary, cannot be translated, recognized or predicted by an NLP application.

There have been a number of works studying the linguistic properties of compounds (cf. Goldsmith & Reutter, 1998; Langer, 1998), and some methods were proposed to treat compounds in NLP applications. However, unlike language modeling in general, compound treatment turned out to be very task-dependent, so that a proposed method cannot easily be transferred to another NLP domain. A number of approaches has been proposed for machine translation (cf. Koehn & Knight, 2003; Popović *et al.*, 2006), for speech recognition (cf. Larson *et al.*, 2000; Ordelman *et al.*, 2003), and for word prediction (cf. Baroni *et al.*, 2002; Trost *et al.*, 2005). Since the latter is the main concern of this work, in the following the focus is placed on compounding in word prediction:

Baroni *et al.* (2002) and Trost *et al.* (2005) propose a compound prediction model, which is based on the morphological structure of many nominal compounds: They consist of a *head* (i.e. grammatically dominating) and a *modifier* part, and are normally *right-headed*, i.e. the head follows the modifier. In between so-called *compounding suffixes*[14] (CS) can occur, which are only functional. To give an example: As Figure 3.7 shows, '*Hundenase*' ("dog's nose") is analyzed into a modifier '*Hund*' ('dog'), '*e*' (compounding suffix) and a head '*Nase*' ('nose').

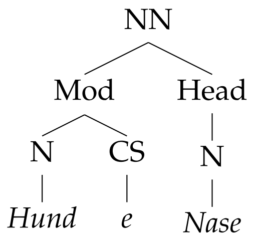


Figure 3.7: Head-modifier analysis of a German compound ('*Hundenase*')

The model of Baroni *et al.* (2002) is based on automatically derived semantic classes, from which the modifier-head pairs can be selected. In addition probabilities for words being either a head or a modifier ($P_{ishead}(w_i), P_{ismod}(w_i)$) have been calculated on previously parsed training corpora. When in use the compound predictor is started as soon as a word which is likely to be a modifier has been entered. The probability of the following word (which has to be a head to the modifier) is then calculated by interpolation of its raw probability $P(w_i)$, the probability of its PoS-tag $P(w_i|c)$ (only nouns can serve as heads), the probability for being a compound head $P_{ishead}(w_i)$ and the probability of being a head to this modifier $P_{class}(w_i|m)$, as derived from the previously determined semantic classes. In addition a set of rules deals with the assignment of the correct suffix.

Unfortunately, even though it is rather elaborate, and even though it offers to create a large range of compound words, which are not part of the vocabulary, this model shows only marginal improvements in ksr for German as for other Germanic languages (app. +0,3%). This is probably due to the presuppo-

[14]Sometimes these elements are also referred to *joint morphemes*, this is however not the appropriate term, since they morphologically belong to the first element.(cf. Langer, 1998).

sitions that are made on the type of compound to be treated (*N+N*). Whereas this type represents certainly the most frequent one, the nature of compounding seems to be more creative than that. In chapter 6 we will reconsider this problem and propose our method to deal with compounds.

3.6 Maximum Entropy models

After having considered Markovian models as well as approaches making use of linguistic insight we now present a model which is – in principle – able to integrate both perspectives. As we have seen especially in the past section, there are many more information sources that can be exploited to estimate the probability of a word than simple n-grams: We could use partial part-of-speech information, chunks or partial parse trees, and we could also imagine to exploit semantic information (which has not been an issue so far), all derived from the already entered context h. However, the combination methods we have discussed up to now, can join these estimates only in a suboptimal way. Whereas a backoff strategy would always choose either one of the given models (which is certainly not a useful solution), interpolation constructs the linear average over all models. This means that for every model a part of the probability mass is reserved, no matter if a model can be applied for a given history or not. The influence of each model is optimized globally, it does not take into account the local constraints given by the current history. Thus, an interpolation makes rather rigid assumptions about the influence of each model given any possible context, which finally leads to wrong estimates.

Maximum Entropy (ME) models resolve this problem by not combining model estimates but by combining the information sources directly into one model. The Maximum Entropy principle can be attributed to the works of Jaynes (1957), and it can be stated as follows (from: Rosenfeld, 1996):

1. Reformulate the different information sources as constraint functions that are to be satisfied by the combined estimate.

2. Among all possible probability distributions that satisfy these constraints, choose the one with the highest entropy.

The rationale behind this principle is to make the fewest assumptions on events for which we have no further evidence; as long as the constraints – imposed by our training data – are satisfied, no further assumptions are made.[15] To build an ME model from our previous models, we first have to reformulate all our single models as constraint functions. For a bigram model, this might look as follows:

[15]Some authors (e.g. Berger *et al.*, 1996) attribute this idea to William of Occam's principle to make the least assumptions possible in order to explain a phenomenon (*Occam's razor*).

$$f(w_{i-1}, w_i) = \begin{cases} 1 & \text{if } (w_{i-2}w_{i-1}) \in S \\ 0 & \text{otherwise} \end{cases} \tag{3.28}$$

where S is the observed subset of our possible event space S, i.e. $f(w_{i-1}, w_i)$ is 1 if we have seen w_{i-1}, w_i and 0 if not. It is clear that this leads to a very large number of constraint functions, one for every possible sequence w_{i-1}, w_i. To form a model out of these constraints, the following formula is applied:

$$P(w_i|h) = \frac{1}{Z(h)} \cdot exp\left(\sum_k \lambda_k \cdot f_k(h, w_i)\right) \tag{3.29}$$

where λ_k are the parameters and $Z(h)$ is a normalizing term, so that the sum of all estimates is 1 for every possible context h. To maximize entropy, we have to set the parameters λ_k so that the resulting distribution is the flattest possible. This can be achieved by a *Generalized Iterative Scaling* (GIS) algorithm (cf. Darroch & Ratcliff, 1972), which operates very similar to the already described EM algorithm (cf. 3.4.1), but is guaranteed to find the global maximum. Unlike the EM algorithm it is however computationally very demanding, which makes training an ME model a problematic procedure. The application of such a model is CPU-intensive as well, because the normalization has to be performed at every update of the context, taking into account every parameter λ_k.

To conclude: ME modeling offers an elegant and general way to combine information from various sources. There is also a growing number of works reporting significant performance gains for ME models with respect to standard approaches (cf. Rosenfeld, 1996; Khudanpur & Wu, 2000). However, up to now its computational complexity has prevented this technique from being widely used. This could change with Moore's law and with the development of more efficient algorithms, in any case ME modeling will remain subject to intensive research.

3.7 Conclusion

We started this chapter by an abstract consideration on communication. We then gave a short introduction to the basics of information theory in order to characterize the notion of redundancy and to show how future symbols can be predicted from the context, and how all this is related to AAC. Afterwards we presented the theory and practice of Markovian (n-gram) language models and explained their intrinsic problems: data sparsity, limited context and inability to incorporate non-linear dependencies. We introduced and discussed a number of smoothing methods, making the estimation of unseen events possible, and we depicted some algorithms which are needed for parameter estimation (EM) or model reduction (*Stolcke* pruning). In the end we presented possible alternatives to n-gram approaches: linguistically motivated models such as PCFG's, and maximum entropy models, which enable to integrate any kind of information within one model.

Chapter 4

User adaptation

An army without flexibility never wins.
An unbending tree is easily
brought down by the wind.
The hard and stiff fall.
The soft and flexible rise.

LAOZI
(*Tao Te King*[1], 76, 6*th* c. BC)

The prediction models presented in the last chapter have one property in common: they are all static in the sense that their predictive capacities do not evolve over time. While the methods presented in the following still rely on such static models, they do however include a dynamic aspect in that they adapt the underlying model to the current input. Such an adaptation can happen temporarily or permanently, and it can focus on different linguistic properties of the input.

In this and the following chapter an introduction to a number of adaptation techniques is given, and results from our work on the application of these techniques are presented and discussed. Whereas the subsequent chapter focuses on semantic adaptation, in the following a range of methods are presented that allow adapting a word predictor to the user's style of communication: recency promotion (or *cache*) models, a user lexicon and a dynamically adapted language model, learning on the user input (*Dynamic User Model*, DUM).

The chapter is structured as follows: While in the first section (4.1) the need for adaptation is motivated, the second section (4.2) approaches the adaptation problem from a theoretic point of view and introduces three frameworks to achieve a dynamic integration of new information. Section 4.3 then presents in detail the above mentioned adaptation techniques. In section 4.4 the evaluation methodology as well as the characteristics of the training and the test data

[1]Transl. by George Cronk, 1999.

are explained. The last section (4.5) then presents and discusses the evaluation results for all three adaptation methods.

4.1 The necessity of adaptation

As argued in the last chapter, statistical language models have become very influential in many NLP techniques and applications. A difficult issue for these models however – as for all data-driven approaches – is their sensitivity to the training material. Such models necessarily adapt to the prevalent syntactic and semantic characteristics of these resources, which means that their performance will always be best on text from the same domain, and it will be the worse the more the statistical properties of the training and the usage (or test) data differ. To give an example for the strength of this dependency: Rosenfeld (1996) has shown that a language model which was trained on *Dow-Jones* newswire corpora has a perplexity twice as high when tested on a (quite similar) *Associated Press* corpus rather than on some other *Dow-Jones* text.

One first trivial conclusion from this observation would be to use training corpora which are as close as possible to the task at hand, e.g. for building a machine translation system for parliament speeches, one should use parliament speeches as training material.

However, when we try to transfer this idea to the construction of an AAC system, we run into two problems: Firstly, for the time being there are no large corpora from AAC users available for training (cf. Trnka & McCoy, 2007). Secondly, most of the time an AAC device is to be considered a *general purpose* tool, i.e. it can neither be shaped to fit a standard user nor on a particular communicative context. From our own experiences with AAC users at the rehabilitation center of *Kerpape* (cf. also section 6.4) we have found the following typical purposes for which an AAC system can be used:

- Colloquial conversation (as usual speech)
- Writing personal letters and e-mails
- School-related dialog (student's replies)
- Specific language training with speech therapists
- Medical dialog with physicians and practitioners

It is already clear that these usage contexts differ quite strongly from one another, and one can imagine many more (for example writing a book or a scientific work). In addition, as discussed in chapter 2, AAC users usually differ quite strongly from one another, depending not only on their personality and specific way of communicating but also on their clinical pattern. A child suffering from cerebral palsy and a 50-year old *Locked-In* patient will probably communicate in a very different manner.

So, if the prospective user of an AAC system and the purpose(s) for which she or he will use it are not known, how can we reduce the deteriorating training effects as much as possible? There are three remedies: The first is to diversify the training corpus, i.e. to include resources from various domains (referred to as *register-diversified corpora*; cf. Biber, 1993). Such corpora are not only valuable training resources, they are also an indispensable means for the study of linguistic phenomena. Whereas they do exist now for the English language (cf. the *British National Corpus*, comprising more than 100 million words from various domains; Burnard & Aston, 1998), it is still rather difficult to find diversified corpora for other languages. Especially obtaining large corpora of transcribed speech, which would be most helpful in the context of AAC, is very difficult for languages such as German or French.

The second remedy is size: it can be assumed that larger corpora provide more solid statistical estimates, therefore it is better to use large amounts training data. In some works it could be shown that training on very large corpora, even from other styles and domains, results in more reliable and robust models than training on small but adapted corpora (cf. Trnka & McCoy, 2007).

Obtaining large corpora is not very difficult anymore. Many important newspapers and press agencies now offer their archives in electronic format, and one single year of a newspaper edition already comprises tens of millions of words. Moreover, newspaper text is a valuable resource for training a language model, since the text is usually rather new, it is written by many authors, and it covers various topics and text styles, such as press statements, essays, interviews and travel reports. But it goes without saying that the main focus of newspaper text is on politics and that most articles are written in an objective and factual "news"-style, so we can certainly not speak of a well-balanced corpus. Whereas it is not unimaginable that AAC users do write newspaper articles, most of the time they will not communicate in the way newspaper editors write.

Another source for obtaining large corpora is the world-wide-web, offering textual documents to an almost inexhaustible extent. The already mentioned *Web 1T 5-Gram* data set, offered by *Google Inc.* for example offers n-gram frequencies that have been estimated on 2 trillion word token from publicly accessible web pages. Data acquired in this way however include a number of problems: They are uncontrolled, i.e. their origin can hardly be verified, they can comprise duplicates, and the number of spelling and grammatical errors is rather high.

The third remedy to reduce the effects of the training data is model adaptation. Rudimentary strategies to adapt to the user's way of communicating have already been applied by the first electronic AAC devices (cf. Hunnicutt, 1986; Swiffin *et al.*, 1987a), and in many other NLP areas such strategies are applied as well (cf. De Mori & Federico, 1999; Rosenfeld, 2000; Bellegarda, 2004). Adaptation strategies can be characterized with respect to their latency: While *short-term* methods have an immediate (and usually ephemeral) effect on the underlying model, *long-term* methods adapt slowly but permanently to the user's input.

Furthermore, we can distinguish adaptation methods with regard to the information sources they exploit: Whereas *user adaptation* methods make use of the available input from the user to adapt to her or his lexical and syntactic preferences (the user's language style), *topical* or *semantic adaptation* refers to methods that try to discover the current semantic context of a document to be written in order to adapt to the given semantic constraints. This concerns in particular the content vocabulary, which can be assumed to be more dependent on the current topic of discourse. Such methods will be the subject of the next chapter.

4.2 Combination schemes

From an abstract perspective the problem of adaptation can be described as follows: We face two information sources, a large text corpus of general language B, which can be used to derive base estimates, and a (usually) small adaptation corpus A, stemming from user input. Two tasks have to be performed in order to obtain an adapted model: (i) we have to extract the relevant information from the adaptation corpus, and (ii) we have to combine this information with the base model. The information to be extracted can concern various aspects of linguistic description, and a large range of methods can be applied here. Such approaches will be described and investigated in the remainder of this and the following chapter. Let us first however look at some fundamental paradigms how the incoming information can be integrated.

Information integration can be performed on several processing levels: One can combine the (real or estimated) frequency counts of the base and the adaptation data and then continue to build an integrated model, one can supplement a base model with additional constraints, derived from the adaptation data, or one can operate on the model level (model merging). In the following a short introduction to the main paradigms in this field is given.

4.2.1 *MAP* adaptation

The *Maximum a Posteriori* (MAP) criterion is closely related to the *Maximum Likelihood Estimate* (MLE). Given two distributions (based on two data sets A and B) it maximizes the combined probability for the observed data (cf. Federico, 1996; De Mori & Federico, 1999; Bacchiani & Roark, 2003).[2] A MAP-based adaptation scheme therefore operates on the count level: To maximize the posterior probability estimate (based on the observations from A and B), the frequency counts of the single models are simply added; this leads to the following posterior estimate $P_{MAP}(w_i|h)$:

[2] In fact, when one of the two data sets gets larger and larger, the MAP estimate approximates the MLE.

$$P_{MAP}(w_i|h) = \frac{\alpha \cdot C_B(hw_i) + C_A(hw_i)}{\alpha \cdot C_B(h) + C_A(h)} \tag{4.1}$$

In its general form, the weighting factor α ought to be 1. In this case however the combined model is strongly biased towards the base data, since adaptation corpora are usually much smaller. To reduce the influence of the base model, an α smaller than 1 is normally applied. It can remain constant over all histories h, or it can be calculated separately for each h. The difficulty is then to find the optimal value (or values) for α. It is usually determined empirically on held out data, but kept constant over the whole integration process. Another difficulty using MAP is that the raw frequency counts of both data sets A and B have to be carried along; the probability calculation (including smoothing and truncation) then has to be performed on-line during prediction, which, depending on the schemes applied, implies more computational effort.

4.2.2 *MDI* adaptation

In section 3.6 the *Maximum Entropy* model was presented. Adaptation by *minimum discrimination information* (MDI) is strongly related to this model (cf. Della Pietra *et al.*, 1992; Chen *et al.*, 2003). It integrates an a priori distribution $P_B(M)$, estimated on a large, general corpus B with a set of constraints, derived from the adaptation corpus A so that the resulting distribution is *closest* to $P_B(M)$. Closeness is here established by measuring the *Kullback-Leibler* distance (cf. equation 3.24) of the prior distribution and the resulting model. MDI adaptation has the following parametric form:

$$P_{MDI}(w_i|h) = \frac{P_B(w_i|h_i)}{Z(h)} \cdot exp\left(\sum_k \lambda_k \cdot f_k(h, w_i)\right) \tag{4.2}$$

where $f_k(h, w_i)$ is a constraint function derived from the adaptation corpus A and λ_k is an associated MDI coefficient. The similarity of this definition with the equation in section 3.6 is obvious. And as for Maximum Entropy the constraint parameters λ are estimated by *Generalized Iterative Scaling* (cf. Darroch & Ratcliff, 1972), which is guaranteed to provide a globally optimal solution. However it was already mentioned that this algorithm is computationally rather demanding. This is especially hindering in the context of dynamic adaptation, since in this case new information continuously arrives and parameter estimations have to be updated on-line.

4.2.3 Model merging

Another way to integrate new information is to operate on the model level, i. e. to estimate two separate models on the data of A and B, respectively, and to combine them afterwards. The most straightforward way of combining models probably is linear interpolation of the two estimates $P_B(w_i|h)$ and

$P_A(w_i|h)$:

$$P_{MM}(w_i|h) = \lambda_B \cdot P_B(w_i|h) + \lambda_A \cdot P_A(w_i|h) \tag{4.3}$$

The coefficient λ can then again be estimated by an *expectation maximization* algorithm (cf. section 3.4.1). In an adaptation setting this algorithm enables to dynamically modify the coefficients, according to the performance of each model (cf. also Choi & Oh, 2006). As the EM algorithm constantly observes the current performance of the participating models, the overall model thus becomes sensitive to dynamic performance changes over time.

However, depending on the model under consideration (e.g. if the two models are very complementary), it might be more sensible to apply a backoff technique:

$$P_{MM}(w_i|h) = \begin{cases} P_A(w_i|h) & \text{if } C(hw_i) > 0 \\ \alpha \cdot P_B(w_i|h) & \text{otherwise} \end{cases} \tag{4.4}$$

Here the base model estimate is only applied if the adaptation model cannot provide any estimate. As before, the constant backoff weight α serves here as a normalizing factor. Depending on the reliability of the adaptation data this scheme can also be applied vice versa, so that we back off to the adaptation model only if the base model does not return an estimate.

4.3 User adaptation techniques

In the following three user-adaptive strategies will be presented: Whereas the cache model represents a short-term method, the adaptive user lexicon as well as the dynamic user model are techniques acting in the long term. Introductory overviews on adaptation methods for language models are given by De Mori & Federico (1999) and Bellegarda (2004).

4.3.1 Recency promotion (cache) model

Cache models take advantage of the phenomenon that people like to repeat themselves. And even if we do not want to accept this (slightly disrespectful) allegation, it is a matter of fact that words are not evenly distributed over a corpus but rather occur in bursts[3]; once they are uttered, they have a higher probability of recurrence in the given context. This has been known for a long time, but the importance of this influence was only recognized in the 1980s, when *Shannon game* experiments (cf. also section 3.1.2) were performed at IBM (cf. Rosenfeld, 1994). The results of humans were compared to a trigram language model, and it was discovered that in 40% of the cases, where the humans

[3]This only applies for content words. Function words have a much more even distribution.

outperformed the LM, the word to predict had already occurred in the context. It therefore seems worthwhile to keep track of the recently entered words and promote their probability estimates.

Cache models were first presented by Kuhn (1988) and Kupiec (1989); Kuhn & De Mori (1990) report significant perplexity reductions with respect to a trigram baseline. In general such models work by interpolating a standard model (e.g. a trigram) with a cache function:

$$P(w_i|h) = \lambda_B P_B(w_i|h) + \lambda_{cache} \cdot P_{cache}(w_i|w_{i-1-L}, ..., w_{i-1}) \tag{4.5}$$

where L represents the maximum length of the cache, and λ_B and λ_{cache} have to sum up to 1 (dynamically determined by EM). Obviously, L is reduced to $\min(i-1, L)$ if the current history is shorter than L. In its simplest form the cache-based probability is calculated as follows:

$$P_{cache}(w_i|w_{i-l-1}, ..., w_{i-1}) = \frac{1}{L} \sum_{k=i-L}^{i-1} I(w_k = w_i) \tag{4.6}$$

where $I(w_k = w_i)$ represents an indicator function returning 1 if the given word w_i occurs among the L most recent words (i.e. is in the cache), and 0 otherwise. In this way every word w_i in the cache receives a constant amount of size $\frac{\lambda_{cache}}{L}$ (according to the number of times w_i occurs). Normal sizes for L are 100 to 1000; function words are normally not added to the cache because their occurrence probability depends much less on previous occurrences.

The elements to be cached need not be words only. There have been promising approaches caching the most recent bi- and trigrams (cf. Jelinek *et al.*, 1991). Smaïli *et al.* (2006) have presented a feature cache model that performs a shallow analysis of the context and keeps the last morpho-syntactic features (such as number and gender) in order to better model agreement of future words.

Moreover, different cache functions can be applied. In the above function the cache factor is constant, i.e. every element of the cache receives the same amount of additional probability. This however probably is not the optimal solution: While a word is unlikely to re-occur immediately after its first occurrence, it can be expected that there is a recurrence peak and that the influence of recurrence diminishes with growing distance; the probability of a word to re-occur will therefore not be constant.

To get a clearer picture of this dependency between distance and recurrence, we determined distance frequencies on large newspaper corpora for French (*LeMonde*; 100 million words), German (*Die Tageszeitung*; 121 million words) and English (*The Guardian*; 118 million words). Function words were excluded from the distance frequencies but not from the distance calculation itself. Figure 4.1 displays the average probabilities of recurrence from 0 to 1000 words for the three languages.

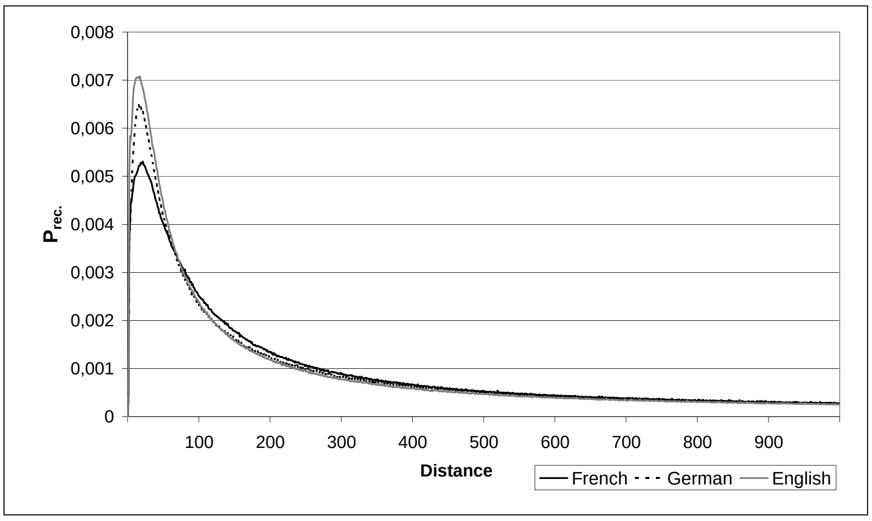


Figure 4.1: Recurrence probability for distances from 0 to 1000 words; measured on large newspaper corpora for French, German and English

The curves in Figure 4.1 suggest a clear and stable dependency between the distance and the probability of recurrence. The probability rises quickly from a distance of 0 to approximatively 20 words and falls exponentially after this peak.

Since the curves in Figure 4.1 show such a characteristic and smooth pattern, it seems worthwhile to adapt the cache function accordingly. This has been done by Clarkson & Robinson (1997) who present a cache model with an exponentially decaying probability function. They calculate the cache probability as follows:

$$P_{cache}(w_i|w_{i-1-l},...,w_{i-1}) = \nu \cdot \sum_{k=i-L}^{i-1} I(w_k = w_i) \cdot \mathrm{e}^{\gamma(i-k)} \tag{4.7}$$

where I is again the indicator function, α is the decay rate and ν a normalizing constant ($= \sum_k^{i-1} P_{cache}(w_k)$). The inclusion of the decay factor $e^{\gamma(i-k)}$ assures that the recurrence probability declines exponentially with a growing distance. However, this function cannot model the low probability for short distances (0-20 words, cf. Figure 4.1), it therefore overestimates recurrence for short distances.

To approximate the probability function even better, we propose a two-fold cache function that distinguishes the rising and the falling part of the probability curve. This is achieved by the following function (distance $d = i - k$):

$$Cf(d) = \begin{cases} \nu \cdot \frac{d^\alpha}{\mu^\alpha} & \text{if } d \leq \mu \\ \nu \cdot \mathrm{e}^{\gamma \cdot d} + \delta & \text{if } d > \mu \end{cases} \tag{4.8}$$

where μ is the distance of the highest recurrence probability (peak), α is the growth rate, δ is an adjustment factor, γ is the decay rate and ν a normalizing constant. This two-fold cache function $Cf(d)$ then replaces the previous decay function:

$$P_{cache}(w_i|w_{i-1-l}, ..., w_{i-1}) = \nu \cdot \sum_{k=i-L}^{i-1} I(w_k = w_i) \cdot Cf(i-k) \tag{4.9}$$

Figure 4.2 shows the probability curves for distances from 0 to 1000 words for the average of the three curves in Figure 4.1, the twofold function as displayed in 4.8, the exponential function used by Clarkson & Robinson (1997) and the constant function as introduced in the beginning of this section (cf. 4.3.1).

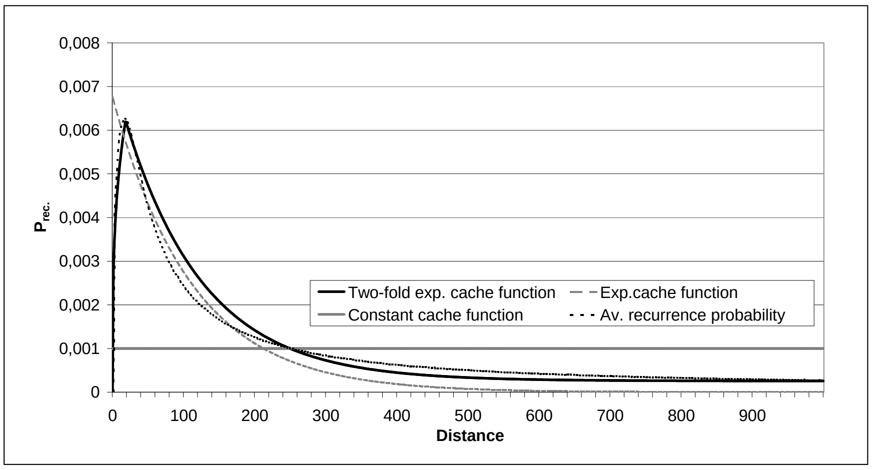


Figure 4.2: Average recurrence probability and probability estimates of the constant, the exponential and the two-fold cache function (with $\alpha = 0,3$, $\beta = 1,1$, $\gamma = -0,009$, $\delta = 0,04$)

It can clearly be seen in Figure 4.2 that the constant cache function largely underestimates the recurrence probability for words of a distance of up to 250 words, and it overestimates afterwards. The plain exponential function estimates the recurrence probability much better, however it strongly overestimates on short distances (from 0 to 20 words), and it disregards the influence of long-distance words (after 500-600 words). The two-fold exponential cache function, as given in equation 4.8, best approximates the average recurrence probability. It is therefore reasonable to apply this one in our cache model.

4.3.2 User lexicon

Updating the lexicon according to the user's input has been one of the earliest adaptation techniques, it can be found already in the first AAC systems (cf. Hunnicutt, 1986; Swiffin *et al.*, 1987a) and in many succeeding systems as well (cf. Le Pévédic, 1997; Carlberger *et al.*, 1997; Trost *et al.*, 2005; Blache & Rauzy, 2007b). The ability to predict previously unknown words significantly increases the user's personal satisfaction with the system. Especially personal names and other proper nouns have a strong emotional impact; if the system is able to integrate them, it shares an important part of the user's knowledge, and in turn it is appreciated much more.

But also from the modeling perspective, vocabulary adaptation is of help. It is a trivial observation that the percentage of *out-of-vocabulary words* (*OOV* henceforth) grows with the difference between the training data and the context of use, and the percentage of unknown words has a direct deteriorating effect on prediction capacity.

Two aspects have to be distinguished with respect to the adaptation of the vocabulary: The first one is the ability extend the vocabulary by previously unknown words, the second is the way to provide statistical information for these words. Many models simply integrate new words by assigning a static (usually very low) probability, other approaches dynamically update the probability estimates for these words by recording their usage frequency. The model presented in the following calculates a dynamic probability estimate that takes into account the user's word choice.

Using a backoff scheme (cf. sections 3.3.2 and 4.2.3) a user lexicon (including its dynamic update) can be integrated to a static language model as follows:

$$P(w_i|h) = \begin{cases} P_B(w_i|h) & \text{if } P_B(w_i|h) > 0 \\ \alpha \cdot P_{UL}(w_i) & \text{if } P_{UL}(w_i) > 0 \\ & (\rightarrow C_{UL}(w_i) = C_{UL}(w_i) + 1) \\ 0 \text{ otherwise} & (\rightarrow C_{UL}(w_i) = 1 \text{ if } SC(w_i)) \end{cases} \tag{4.10}$$

where $SC(w_i)$ is a *sanity check* routine deciding if the unknown lexical item is to be integrated or not (e.g. depending on its length, presence of numeric characters, punctuation signs etc.). $P_{UL}(w_i)$ is based on the maximum likelihood estimate (MLE; cf. section 3.2):

$$P_{UL}(w_i) = \frac{C(w_i)}{\sum_j^s C(w_j)} \tag{4.11}$$

where s is the size of the user vocabulary. Some approaches also use a simple indicator function $I(w_i, UL)$ returning $\frac{1}{s}$ ($s = |UL|$) if w_i is in the user vocabulary and 0 otherwise. The above formula already formalizes the idea of automated integration ($C_{UL}(w_i) = 1$), sanity check ($SC(w_i)$) and frequency update of unknown words ($C_{UL}(w_i) = C_{UL}(w_i) + 1$).

Such an automated update is of course not a necessary part of the user lexicon model. Integration as well as update can also be controlled by external actions such as manual insertion and modification. However, each of these possibilities also includes a number of difficulties, which have to be dealt with:

- If the adaptation process is explicit, every unknown word has to be confirmed or rejected by the user, which can quickly cause dissatisfaction.
- In the case of implicit adaptation (i.e. every new word is integrated), the user lexicon gets polluted by miss-spelled words or words that have been used only once (remember for example the problem of single-word compounds, described in section 3.5.4). Such a strategy has to be equipped with a *garbage collection* method that automatically filters out newly integrated words which are not re-used (e.g. by applying a frequency filter). However, such an automated procedure is not without risks, since it changes the model in an unsupervised way.
- If the system uses morpho-syntactic information for keystroke reduction, new words represent a particular problem, since their linguistic properties are not known. They have to be guessed from the form of the given word, but such guessing procedures are obviously error-prone. Some systems (such as Blache & Rauzy, 2007b) rely on the user to specify the missing linguistic information, but this increases the cognitive demands even more, and the user's linguistic knowledge is often very limited.

4.3.3 Dynamic user model

The idea of the dynamically updated user lexicon can be extended to a full n-gram language model. Such a model uses as training input all incoming data from the user, and its probability estimates are dynamically updated after every insertion of a word. This model is then interpolated with a large base model, providing more general (and reliable) statistical estimates. The approach of merging a large general model with a small user-oriented model has shown to be rather helpful in various NLP tasks, such as speech recognition (cf. also Woodland *et al.*, 1998; De Mori & Federico, 1999; Bellegarda, 2004; Choi & Oh, 2006).

The application of a dynamic user model (DUM) involves three major steps: estimation, integration and update. In the *estimation* step the current probability estimates for the user model are calculated, and a smoothing scheme is applied. We use a trigram model and we apply an absolute discounting scheme (cf. section 3.3.3), which is simple and has been shown to work reliably on smaller training corpora:

$$P_{DUM}(w_i|w_{i-2}w_{i-1}) = \begin{cases} \frac{C(w_{i-2}w_{i-1}w_i)-D}{C(w_{i-2}w_{i-1})}, & \text{if } C(w_{i-2}w_{i-1}w_i) > 0 \\ \alpha(w_{i-2}w_{i-1})P_{DUM}(w_i|w_{i-1}), & \text{otherwise} \end{cases} \tag{4.12}$$

The *integration* step of the DUM with the base model is performed by linear interpolation of the single estimates. As explained in section 4.2.3, an EM algorithm (cf. section 3.4.1) is applied in order to estimate the coefficients. This enables to dynamically control the influence of the DUM on the overall estimates.

The *update* step then cares for either integrating the just inserted n-grams ($w_{i-2}w_{i-1}w_i$, $w_{i-1}w_i$ and w_i), if they do not occur yet in the model, or for augmenting their frequencies:

$$
\begin{aligned}
C^{k+1}_{DUM}(w_{i-2}w_{i-1}w_i) &= \begin{cases} C^{k}_{DUM}(w_{i-2}w_{i-1}w_i)+1, \text{ if } w_{i-2}w_{i-1}w_i \in DUM \\ 1 \text{ if } SC(w_{i-2}w_{i-1}w_i) \end{cases} \\
C^{k+1}_{DUM}(w_{i-1}w_i) &= \begin{cases} C^{k}_{DUM}(w_{i-1}w_i)+1, \text{ if } w_{i-1}w_i \in DUM \\ 1 \text{ if } SC(w_{i-1}w_i) \end{cases} \\
C^{k+1}_{DUM}(w_i) &= \begin{cases} C^{k}_{DUM}(w_i)+1, \text{ if } w_i \in DUM \\ 1 \text{ if } SC(w_i) \end{cases}
\end{aligned}
$$

As for the user lexicon the incorporation of an unknown n-gram is again controlled by a sanity check routine (*SC*) ruling out unwanted words and n-grams. However, such a model still has to address the problem of pollution; it needs to be equipped with either an automated garbage collection or an editing system that allows the user to modify or delete inappropriate elements. Another method to deal with the garbage problem is to start every session with an empty DUM or to flush it every few thousand words. In this case the dynamic user model becomes very similar to a trigram cache model (cf. section 4.3.1 and Jelinek *et al.*, 1991).

It is clear that such an approach does not need to be restricted to one DUM only. A word predictor could store and manage several DUMs in parallel and apply the most appropriate one depending on the task at hand. The text style of e-mails for example is rather different from scientific or literary writing, therefore charging a certain DUM could be based on the given application in use (e.g. word processor or e-mail client). In this way the adaptation capacities of the word predictor could be maximized.

4.4 Experimental paradigm, training- and test corpora

4.4.1 Register-based model evaluation

Empirically evaluating the adaptation capacities of a given model is not straightforward. As already explained in the introduction of this chapter, persons using an AAC system have rather heterogeneous demands, and even a single user will apply a communication system in very different situations. It is therefore almost impossible to model the real behavior of such a user for evaluation, however we can test the adaptive capacities of our models by a paradigm which approaches the diversity of real-use contexts: *cross-register evaluation*.

Unlike *intra-register* evaluation, which uses test data from the same register[4] as the training corpora, and which is by far the most common means of evaluating language models (cf. for example Woodland *et al.*, 1998; Lesher *et al.*, 1999; Goodman, 2000; Schadle *et al.*, 2004), cross-register evaluation uses test corpora from several distinct registers in order to show the performance of a model over a broader spectrum.

This evaluation paradigm is based upon the works of Biber (1988, 1993), who clearly shows the strength of variability in the lexical and grammatical properties of different text types (genres or registers) and who argue that the empirical analysis of general language phenomena has to be based on corpora from multiple registers. Cross-register evaluation is also promoted by Trnka *et al.* (2007), who investigate the dependency between training and test data on a broad range of different corpora.

In the following a detailed description of our training as well as of our test data is given. This section is also relevant for the models and methods presented in the following chapters, because (for the sake of comparability) their evaluation will be based on these data as well.

4.4.2 Training data, baseline model and evaluation method

As discussed before, obtaining large corpora from different registers can be rather complicated for languages other than English. Newspaper corpora normally represent a reasonable trade-off between diversity and size, since they are edited by many authors and comprise various topics and (sub-)registers (newspaper text is often itself regarded as a register). Moreover, their quality (with respect to spelling errors) is usually high (compared to text acquired from the web) and their origin is controlled. We therefore decided to use newspaper corpora instead of web-based data, and since our study also aims to assure cross-language comparability, we also used newspaper text for the English model, instead of a more balanced corpus like the *BNC*, which also includes literary and speech data (cf. Burnard & Aston, 1998). Table 4.1 lists the corpora that were used for training our language models. The corpus sizes were determined by memory constraints.[5]

The vocabularies were determined by an algorithm taking both frequency and orthography into account. The vocabulary size is – as usual – traded off

[4]Note that we use the term *'register'* here in the (loose) sense of *textual category*, such as Biber (1988, 1993) did, and not in its (tight) socio-linguistic definition (e.g. of Halliday, 1978).

[5]The workstation on which the language models were computed had 2GB of RAM.

[6]Available from ELDA: http://www.elda.org/

[7]Available from TAZ: http://www.taz.de/

[8]Available from *ProQuest* (formerly *Chadwyck Media*): http://www.proquest.co.uk/

	Origin (years)	Nb. of words	Vocab. size
French	*Le Monde*[6]1998-1999)	$\sim$44 $\times 10^6$	141.078
German	*Die Tageszeitung*[7]1997-1999)	$\sim$37 $\times 10^6$	141.242
English	*The Guardian*[8]1997-1998)	$\sim$49 $\times 10^6$	133.558

Table 4.1: Training corpora and sizes used for the baseline models

between coverage and the resulting model size; a detailed study of the influence of vocabulary size on prediction performance can be found in section 6.9. Words not present in the vocabulary were mapped onto a `<UNK>` tag. Punctuation characters were treated as normal words, and numbers were replaced by a `<NUMBER>` tag, representing an equivalence class for all numbers. In this way, many n-grams typically containing a numeric expression are mapped onto a single n-gram (e.g. "`<NUMBER>` *years old*"), whose probability can then be estimated more reliably. It is obvious that this measure is a first step towards a class-based model (cf. section 3.5.1), and more classes could easily be defined in this way (e.g. `<MONTH>` or `<DayOfWeek>`). It would certainly be interesting to investigate the effects of forming more such classes; this would however require a thorough evaluation, which could not be performed here. We therefore used only the most obvious class, unifying a very large number of items and thereby reducing the number of candidate types to an important extent.

For calculating the baseline model, we employed the *SRI* language modeling toolkit (cf. also section 3.4.3). A 4-gram[9] LM was calculated using interpolated *Modified Kneser-Ney* discounting (cf. Chen & Goodman, 1999; Goodman, 2001, also explained in 3.3.4), which can currently be regarded as the best performing smoothing technique. We tested other smoothing schemata, but could only confirm Chen and Goodman's results; for this reason we will not report any further results here. To reduce the models to a manageable size (the unpruned models had sizes of 800 to 950 MB), *Stolcke pruning* was applied with a threshold θ of 10^{-7} (cf. Stolcke, 1998, and also section 3.4.2). Table 4.2 displays the numbers of n-grams and model sizes for each model after the application of *Stolcke pruning*.

These models form the baseline for all adaptation methods that will be presented here and in the following chapters. The evaluation measure will be the *keystroke saving rate* as detailed in section 2.5.2 with respect to a list containing 5 words (ksr_5), if not marked otherwise; words that have been shown once in the list, will be filtered until the current word is completed (cf. section 6.3.2).

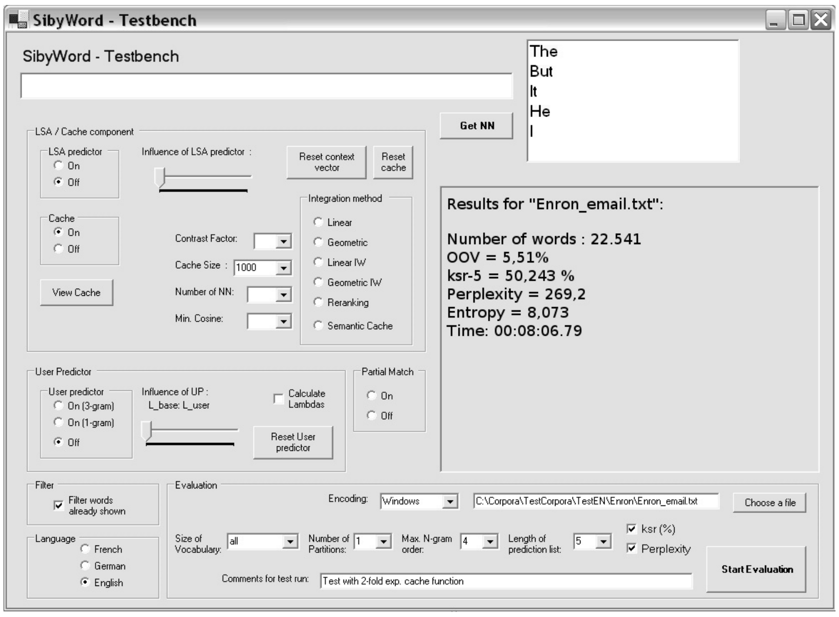
For evaluating a given language model on different test corpora, we developed a *testbench*, which simulates the behavior of a real user applying the predictor to type text. This device reads in a test corpus character by character and compares at every insertion the contents of the prediction list to the origi-

[9]For a comparison of models of different n cf. section 3.4.

	French	German	English
1-grams	141.078	141.242	133.558
2-grams	1.158.832	1.308.970	1.470.967
3-grams	839.049	657.614	999.648
4-grams	348.864	278.617	358.694
Model size	69 MB	67 MB	77 MB

Table 4.2: Numbers of n-grams used and model sizes for the baseline models

nal text; if the intended word is present, it is added to the input, and its current probability estimate (as calculated by the underlying model) is determined as well. In this way keystroke savings, cross entropy or perplexity can be computed. The testbench also allows for parameterizing various factors, such as maximum n-gram order, interpolation coefficients, cache sizes (and functions) etc. All parameters and results are written to log files. Figure 4.3 shows the interface of our testbench.



```
The
But
It
He
I
```

```
Results for "Enron_email.txt":

Number of words : 22.541
OOV = 5,51%
ksr-5 = 50,243 %
Perplexity = 269,2
Entropy = 8,073
Time: 00:08:06.79
```

LSA predictor: On [✓ Off]
Cache: [✓ On] Off
Contrast Factor:
Cache Size : 1000
Number of NN:
Min. Cosine:
User predictor: On (3-gram) On (1-gram) [✓ Off]
Calculate Lambdas ☐
Partial Match: On Off
Filter words already shown ✓
Language: French German [✓ English]
Encoding: Windows
C:\Corpora\TestCorpora\TestEN\Enron\Enron_email.txt
Size of Vocabulary: all
Number of Partitions: 1
Max. N-gram order: 4
Length of prediction list: 5
ksr (%) ✓
Perplexity ✓
Comments for test run: Test with 2-fold exp. cache function

Figure 4.3: User interface of the testbench (v. 1.6)

4.4.3 Test data

As mentioned before, cross-register evaluation assesses a model on corpora from several (sufficiently distinct) language registers. The registers for our test corpora were selected with respect to their distinctness and (to some extent) to their importance for an AAC context. Apart from intra-register data (*newspaper*), text from the following registers was acquired: *literature* (19th century novels), *speech* (dedicated dialogue) and *e-mail* (personal correspondence).

Especially the latter two registers are very interesting for our evaluation, since they represent 'real-use' language, containing lexical and grammatical errors and inserted corrections. It was tried to use test corpora of similar sizes, but – due to availability – this could not be achieved for the speech and the e-mail register. Table 4.3 gives an overview of the test corpora and their sizes.

Register	Identifier	Description	Nb. of words
News	*news-fr*	From *L'Humanité* (2000)	58.457
	news-de	From *Süddeutsche Zeitung* (1999)	56.031
	news-en	From *The Guardian* (2000)	53.070
Literature	*lit-fr*	From *Germinal* by Emile Zola (first published in 1885)	50.251
	lit-de	From *Effi Briest* by Theodor Fontane (1894)	54.844
	lit-en	From *The Picture of Dorian Gray* by Oscar Wilde (1891)	53.640
Transcribed Speech	*speech-fr*	Transcriptions of spontaneous spoken dialogues between French tourist agents and customers; from the *OTG* corpus[10]	15.435
	speech-de	Transcriptions of telephone conversations (making appointments); from the *Verbmobil*[11] project (German part)	20.729
	speech-en	Transcriptions of telephone conversations (making appointments); from the *Verbmobil* project (English part)	20.788
E-mail	*email-fr*	Personal e-mails (from a native speaker, *sent* folder, replies removed)	44.946
	email-de	Personal e-mails (from a native speaker, *sent* folder, replies removed)	15.774
	email-en	From the *Enron* e-mail dataset[12]; *personal sent* folder from one user account, replies removed	22.151

Table 4.3: Overview of the evaluation corpora used

This set of test corpora will be used throughout the evaluation part of this and the following chapter. When subsets hereof are tested, it will be clearly marked. Otherwise the identifiers – as displayed in Table 4.3 – will be used. For demonstrating parameter effects, tests are performed on the intra-register corpora (*news*-(*fr*,*de*,*en*)).

4.5 Results

4.5.1 Baseline model

Table 4.4 shows the baseline evaluation results (ksr_5) for all 12 test corpora. It can be recognized immediately that there is an important difference between the intra-register corpora (*newspaper*) and the corpora from the other registers. Whereas we obtain ksr_5 results of more than 50% for the newspaper corpora, the other registers show significantly lower results (up to -11,8% for *lit-fr*).

Register	French	German	English
Newspaper	57,8%	51,6%	55,5%
Literature	46,0% (*-11,8*)	44,9% (*-6,7*)	49,8% (*-5,7*)
Speech	48,3% (*-9,5*)	49,1% (*-2,5*)	48,5% (*-7,0*)
E-mail	48,6% (*-9,2*)	48,0% (*-3,6*)	49,4% (*-6,1*)

Table 4.4: Results (ksr_5) for the different corpora and languages on the baseline models

A second remarkable aspect are the significantly lower results for German, especially for the newspaper corpus. It can probably be explained by the rather complex noun morphology (3 genders, 4 cases), leading to more inflected forms, but also by the problem of compounding (as discussed in section 3.5.4). This becomes apparent if we regard the proportions of unknown words (OOV) for the different corpora. Table 4.5 displays the percentages of OOVs for all test corpora.

As can be seen from Table 4.5, apart from the e-mail corpus, the OOV proportions of the German corpora are two to three times higher than of the French and English corpora. For *news-de*, almost every fifteenth word was not in the respective vocabulary, which has of course a strong impact on the keystroke savings, since such words cannot be predicted. For better understanding the nature of OOV words, these words were collected during evaluation, and samples were manually categorized for the *news* and the *speech*

[10](cf. Nicolas *et al.*, 2002)

[11]http://www.phonetik. uni- muenchen.de/Forschung/Verbmobil/VerbTRL.html

[12]This corpus was made public during the legal investigation concerning the *Enron* corporation after its bankruptcy in 2001. Cf. http://www.cs.cmu.edu/∼enron/

	French	German	English
Newspaper	2,2%	6,4%	2,4%
Literature	2,4%	4,5%	1,5%
Speech	1,3%	2,5%	0,5%
E-mail	5,1%	4,9%	5,5%

Table 4.5: Percentage of out-of-vocabulary words (OOV) for all test corpora

register. Table 4.6 gives an overview of the results.

	news-fr	*speech-fr*	*news-de*	*speech-de*	*news-en*	*speech-en*
Numeric exp.	36%	12%	12%	24%	30%	26%
Proper names	45%	47%	13%	32%	25%	39%
Compounds	3%	0%	51%	26%	16%	10%
Other nouns	4%	14%	4%	1%	6%	5%
Verbal exp.	4%	13%	4%	6%	7%	6%
Adjectives	3%	4%	5%	2%	10%	3%
Misspellings	3%	4%	5%	3%	4%	3%
Other	2%	6%	6%	6%	2%	8%
Total number:	1273	196	3589	516	1291	97
Sample size:	100	100	100	100	100	97

Table 4.6: Percentage of out-of-vocabulary words (OOV) for all test corpora

The first observation is expected: The largest proportions of OOV words are either numeric expressions or named entities; in the speech corpora we find more proper names than in newspaper. However, for the German corpora we find very high rates of compound words (51%; 26%) among the OOVs[13]. Interestingly, the number of compound words in English is higher than expected (this is mostly due to hyphenated compounds (e.g. *Euro-obstructionist*, *tea-plantation*). For French however the number of compounds (e.g. *psychosociologique* 'psycho-sociological', *hyperconcurrence* 'hyper-competition') is almost negligible ($< 3\%$).

4.5.2 Cache model

The cache model comprises three critical parameters: (i) the interpolation coefficient λ_{cache} integrating the cache estimates to the combined function, (ii) the

[13]Some elucidating examples for non-German speakers: Giant words like *'Institutseinweihungsfeier'* (25 characters) 'inauguration ceremony of the institute' , *'Filmproduktionsgesellschaften'* (29 chrs.) 'film production companies', *'Pflanzenschutzmittelverordnung'* (30 chrs.) 'ordinance on plant protectants' did occur in our corpora!

cache function and (iii) the cache size. The optimal coefficient is not very hard to find, here the EM algorithm (as introduced in section 3.4.1) can be applied.

While it certainly is more appropriate to apply a cache function which follows the recurrence probability (as displayed in Figure 4.2), it is however interesting to see how strongly such a function can improve the results with respect to a constant function. Table 4.7 shows the results of the constant and the two-fold exponential cache function, using the parameters specified in section 4.3.1 for the newspaper corpora.

Cache function	λ_{cache}	*news-fr*	*news-de*	*news-en*
Baseline	0	57,78%	51,56%	55,49%
Constant	0,09 - 0,11	58,05% (+*0,27*)	52,01% (+*0,45*)	55,87% (+*0,38*)
Two-fold exp.	0,09 - 0,12	58,20% (+*0,4*)	52,18% (+*0,62*)	56,03% (+*0,54*)

Table 4.7: Results (ksr_5) for the constant and the two-fold exponential cache function (cache size = 1000)

The results in Table 4.7 show that the exponential cache function yields better results than the constant function; however the gain is not very high (+0,14% - +0,17% in ksr_5). The coefficients – as determined by the EM algorithm – are also comparable, librating around 0,1. It therefore can be concluded that whereas an important part of the potential of a cache is already exploited by a constant function, an exponential function models the recurrence probability of terms even better.

The third parameter to be considered is the size of the cache. To find out about its influence, several cache sizes ranging from 50 to 1.500 were tested. Table 4.8 shows the results for all sizes tested on the three newspaper corpora.

Cache size	λ_{cache}	*news-fr*	*news-de*	*news-en*
0 (Baseline)	0	57,78%	51,56%	55,49%
50	0,06 - 0,07	57,96%	51,84%	55,69%
100	0,08 - 0,09	58,11%	52,01%	55,93%
300	0,09 - 0,10	58,18%	52,10%	55,98%
500	0,09 - 0,12	58,18%	52,19%	56,02%
1000	0,10 - 0,12	58,20%	52,18%	56,04%
1500	0,10 - 0,12	58,20%	52,16%	56,03%

Table 4.8: Results (ksr_5) for different cache sizes, using the two-fold exp. decaying cache function

The development of the results is very similar for all three corpora: The advantage of the cache model grows quickly from already 50 elements to 300, then the additional gains become marginal. After a size of 1000 the gains become stable or decline slightly. Setting the size to 1000 elements therefore seems to be a well-balanced solution. Note that "flushing" the cache (i.e. emptying after an article boundary) was not applied, because - due to missing cues - such a behavior cannot be achieved in a real-use situation.

Table 4.9 gives an overview on the results of the cache model for the corpora from all registers (using the two-fold exponential cache function with the parameters given in 4.2, a cache size of 1000 and a λ_{cache} of 0,11).

	French	German	English
Newspaper	58,2% (+*0,4*)	52,2% (+*0,6*)	56,0% (+*0,5*)
Literature	46,4% (+*0,4*)	45,2% (+*0,3*)	50,0% (+*0,3*)
Speech	51,0% (+*2,7*)	50,7% (+*1,6*)	51,1% (+*2,6*)
E-mail	49,0% (+*0,4*)	48,1% (+*0,2*)	50,1% (+*0,7*)

Table 4.9: Results of the (two-fold exp. decaying) cache model for all test corpora

The gains of the cache model for the written language corpora (*news*, *literature* and *e-mail*) are surprisingly close, ranging from +0,2% to +0,6%; differences between the languages cannot be determined. For the speech corpora however we obtain a completely different picture: Here, the gains of the cache model are much stronger (up to +2,7% with respect to the baseline). This is probably due to the high number of repetitive words and phrases in dedicated speech (e.g. *"I want to"*, *"do you have"*); moreover the vocabulary in speech is usually much more restrained than in written text. Once a word has been added to the cache, it is therefore more probable to re-occur.

In general it can be stated that the cache model yields small but reliable gains over all registers and languages. While remarkable differences between languages cannot be recognized, we observe an important performance discrepancy for oral and written corpora.

4.5.3 User lexicon

For the evaluation of the user lexicon two aspects have to be considered: The plain reduction of OOV words and its overall impact on keystroke savings.

Table 4.10 shows the OOV proportions (and the reduction from the baseline) for all test corpora. All tests were performed on an initially empty UL. Elements of length smaller than 2 or containing numbers or punctuation signs were not added to the lexicon.

The OOV reduction resulting from the user lexicon is quite substantial: For almost all corpora the proportion of OOVs could be reduced by 30%, for some (e.g. *lit-fr*) even over 60%. This shows that an automatically updated user

	French	German	English
Newspaper	1,2% (*-1,0*)	4,3% (*-2,2*)	1,5% (*-1,0*)
Literature	0,9% (*-1,5*)	2,9% (*-1,6*)	1,1% (*-0,4*)
Speech	0,6% (*-0,7*)	0,9% (*-1,6*)	0,2% (*-0,3*)
E-mail	2,4% (*-2,7*)	3,8% (*-1,1*)	3,6% (*-1,9*)

Table 4.10: Reduction of OOV by the user lexicon model for all corpora

lexicon is indeed effective for reducing unknown words. But how does this reduction affect keystroke savings? Table 4.11 gives an overview on the results and deviations from the baseline. Frequencies (and likewise the probability estimates) of OOVs were dynamically updated after every word.

	French	German	English
Newspaper	58,2% (*+0,4*)	52,6% (*+1,0*)	55,9% (*+0,4*)
Literature	47,5% (*+1,5*)	45,9% (*+0,9*)	50,3% (*+0,5*)
Speech	48,5% (*+0,2*)	50,0% (*+0,9*)	48,7% (*+0,2*)
E-mail	49,4% (*+0,8*)	48,7% (*+0,7*)	50,1% (*+0,7*)

Table 4.11: Results for the *user lexicon* (ksr_5) for all test corpora

The effect of the user lexicon on keystroke savings is small but beneficial for all corpora. It seems a little higher for the German corpora, which is obviously due to the higher OOV rates and the stronger OOV reductions achieved (cf. Table 4.10). A significant outlier is the result for *lit-fr* (+1,5%), but, as can be seen from Table 4.10, the OOV reduction achieved by the UL for this corpus is also quite strong (over 60%).

We did not further evaluate the evolutionary properties of the user lexicon; this was however done for the dynamic user model (cf. Figure 4.4). Since that model basically extends the idea of a dynamically updated user lexicon, it can be assumed that its performance evolution will not be too different.

4.5.4 Dynamic user model

As for the user lexicon it is difficult to assess the long-term effect of the dynamic user model, since it evolves with the user input, and it can be expected that the more input the model receives the better it will perform. The test corpora considered here are however of limited size, so the theoretical maximum of performance improvement cannot be found in this way. The results nevertheless give an important indication of the potential of such an approach. As before all tests were performed on an initially empty user model, the DUM was updated after every completed word (marked by either an empty space or a punctuation sign), and unknown strings were automatically integrated, if they were more than 2 characters long and did not contain any number or punctuation characters. Table 4.12 gives an overview on the ksr_5 results for all test corpora and deviations from the baseline.

	λ_{DUM}	French	German	English
Newspaper	0,21 - 0,26	58,5% (+*0,7*)	54,6% (+*3,0*)	56,3% (+*0,8*)
Literature	0,27 - 0,31	50,6% (+*4,6*)	50,0% (+*5,1*)	53,0% (+*3,2*)
Speech	0,42 - 0,53	57,7% (+*9,4*)	57,5% (+*8,4*)	56,9% (+*8,4*)
E-mail	0,28 - 0,38	53,0% (+*4,4*)	51,6% (+*3,6*)	54,1% (+*4,6*)

Table 4.12: Results of the *Dynamic User Model* for all test corpora

As Table 4.12 shows, we get an important increase of keystroke savings (up to 9,4%) for all extra-register corpora. But even for the newspaper corpora we observe some improvement, which is surprising since we would assume the newspaper corpora to be already well adapted to the language model. Whichever test corpus considered, keystroke savings remain higher than 50%, which is a considerable achievement.

Interestingly, as with the cache model we find the highest gains in all languages again for the speech corpora. And again this is probably due to the important stylistic differences, to the rather limited vocabulary of spoken language and also to the high number of repetitive words and phrases. Such phrases can be predicted very easily by the DUM once they have been integrated. The important performance gain can also be recognized by the coefficients controlling the influence of the DUM on the combined model, as calculated by the EM algorithm. Even for the intra-register corpora the final λ_{DUM} scores give more than 20% of the probability mass to the user model. But the extra-register corpora receive even significantly higher scores; for the speech corpora we observe λ_{DUM}-values of up to $0,53$, which means that here the influence of the DUM (trained on as little as 15.000 to 20.000 words!) is larger than the baseline model.

To assess the learning speed of the DUM, we also observed the ksr evolution during the prediction of the test data. Keystroke savings with and without DUM were measured every 2.000 words, and their difference was recorded. The curves in Figure 4.4 show the ksr advantage over the baseline for two test corpora (*lit-fr* and *email-fr*).

As the learning curves in Figure 4.4 show, the DUM-based model already performs 2% better than the baseline after only 2.000 words, and it reaches a plateau of 5% to 6% after approximately 20.000 words. This implies that the training time needed in order to take advantage of the DUM-based model is not very long.

The evolution of the DUM performance can also be estimated by observing the λ_{DUM} coefficients, as estimated by the EM algorithm. Figure 4.5 displays

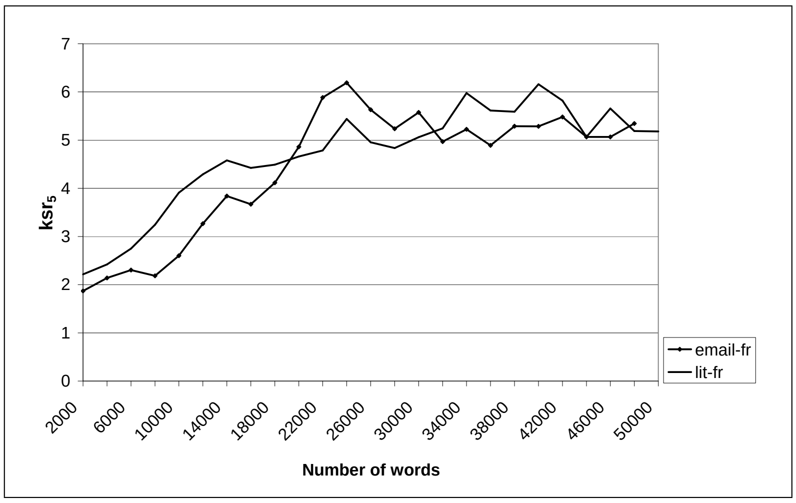


Figure 4.4: Evolution of ksr advantage (+%) of the *Dynamic User Model* over the baseline for two test corpora (*email-fr* and *lit-fr*)

the development of the coefficients for the same test corpora as above (*email-fr* and *lit-fr*).

The curves in Figure 4.5 look very similar: After a quick rise at the first few thousand words the curves flatten and finally seem to approach a level after app. 50.000 words. This is in accordance with the development of the performance gain, and it also shows that learning on a few thousand words is already sufficient in order to improve the overall performance.

To estimate the theoretical limits of this model, another test was performed which is obviously considered illicit under standard experimental conditions, namely the evaluation of the DUM after it has already integrated the test corpus (2^{nd} pass). Therefore the results, as displayed in 4.13, are to be taken with a pinch of salt, however they show how far keystroke savings can be pushed when some text is to be predicted once again.

	news-fr	*lit-fr*	*news-de*	*lit-de*	*news-en*	*lit-en*
ksr_5 1^{st} pass:	58,5%	50,6%	54,6%	50,0%	56,3%	53,0%
ksr_5 2^{nd} pass:	72,9%	70,0%	77,6%	71,7%	74,7%	70,2%

Table 4.13: Results (ksr_5) for the newspaper and the literature corpora, before (1^{st} pass) and after (2^{nd} pass) they have been learned by the DUM

The results of the second pass (line 2 in Table 4.13) are quite remarkable. All keystroke savings are over 70%, the German newspaper corpus even attains 77,6%, which means that nearly every word was displayed immediately in the list and only had to be selected (this result is not far from the theoretical

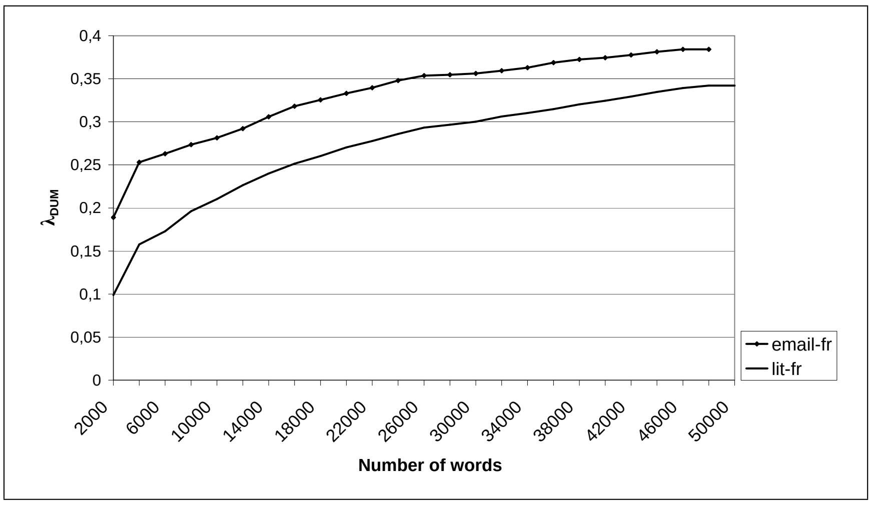


Figure 4.5: Evolution of λ_{DUM} (as calculated by the EM algorithm) for two test corpora (*email-fr* and *lit-fr*)

maximum of 84%, cf. section 2.5.2). The coefficients λ_{DUM} vary from 0,46 to 0,58. This means that the DUM has been able to learn the statistical characteristics of the test corpora very well; moreover, the combined model seems to be react in a rather flexible way to messages it has already learnt. It is obvious that keystroke savings of over 70% will not be achieved in practice, it can however be supposed that the production of messages that have already been integrated will be significantly facilitated by this model. And this property is rather important for an AAC context, where a considerable amount of messages (e.g. functional phrases such as *"I am hungry"*, *"I feel cold"*) are repeated quite often. The dynamic user model learns such messages without any additional effort, and – as can be seen from the results presented above – it is able to retrieve already learned phrases in a rather efficient manner.

4.5.5 Comparison with related work

Considering the results reported above the question arises how they can be interpreted with respect to results from similar works. It is obvious that a straight comparison of our results to others is cannot be achieved, since they were obtained on varying training and test data; moreover many works do not describe their prediction methods in full detail. Still, we deem it important to assemble results of other works, not in order to compare them directly but to enable a more intuitive understanding for their interpretation.

A first work being relevant in this context has already been mentioned in the introductory part: Lesher *et al.* (2002) have investigated the performance of human subjects in a word prediction task. For their experiment they presented a few sentences as initial context to the subjects and then asked them to generate a list of the six words that they deemed most likely to follow. If the correct word occurred in the list, it was added, if not, the next character was provided (similar to the *Shannon game* (cf. Shannon, 1951, and also section 3.1.2)). In this way they could calculate the keystroke saving capacities of their subjects. Three conditions were tested: In the 'no help' condition, the subject had to perform the task without any further information; in the 'simple help' condition a frequency-ordered word list could be consulted, and the 'advanced help' setting provided in addition more advanced statistical information for the given context. The subjects of the 'no help' group arrived at an average ksr_6 of 49%; the 'simple help' subjects obtained 54%, and in the 'advanced help' condition a ksr_6 of 59% was measured (cf. Table 4.14). Another finding of this study was that the amount of context provided seemed to play a major role for the performance: Humans seem to rely much more on contextual clues than on local frequency information. This explains also the significant improvements of the 'simple' and 'advanced help' conditions, where such information was provided in addition.

One of the first systems whose prediction performance was systematically evaluated, was *Profet* (cf. Carlberger *et al.*, 1997, and also section 2.4.1). The system has been developed for several languages (Swedish, English, Danish and Norwegian, among others). Its word predictor uses unigram and bigram information as well as a cache component and a user lexicon. Keystroke savings were calculated on rather small texts (4000 to 6000 characters), the results vary between 33% (Danish) and 37% (English). This is a stark contrast to our results, dropping in no condition under 44% even for the baseline model. Unfortunately, the authors do not further analyze their results, nor they present details on the test data used, so it is difficult to find reasons for such an important difference.

In the *HandiAS* project (Le Pévédic, 1997; Maurel & Le Pévédic, 2001, cf. also section 2.4.1) probabilistic finite state automata (PFST) are applied for prediction. These automata analyze shallow syntactic features (such as number, gender and tense) of the left context and constrain thereby the predicted terms to the ones which are syntactically possible. The system also comprises a user lexicon to which unknown words are added, and the probabilities of the state transitions evolve with the user input. Using these techniques, the system arrives at a ksr_5 of 43,5%, however no further information on the kind of test corpus is given.

SibyMot (Schadle *et al.*, 2004) the word predictor developed for the 2003 version of our AAC system SIBYLLE (Schadle, 2003, cf. also section 2.4.1), makes use of a chunk-based language model for prediction (cf. section 3.5.2). Unlike the two approaches described above it does not integrate any adaptive features, however significantly higher keystroke savings are reported, reaching up to 57,1%. But even the baseline model (trigram) is reported to achieve a ksr_5 of 55,8%, meaning an advantage of +1,3% for the chunk-based model. The rather high results of *SibyMot* are however also due to the nature of the test data, stemming from the same register as the training corpus (newspaper). The size of the training data was however rather small (2 million words), which explains the difference to our model, being trained on much more data (44 million words). As shown in Table 3.4, our trigram model achieved a ksr_5 of 57,6% in a comparable condition.

The *Fasty* language component (cf. section 2.4.1) has been evaluated by Trost *et al.* (2005). The predictor is based on a combination of several strategies such as word bigrams, PoS trigrams, and a FST-based grammar component, a prediction method for German compounds and a user lexicon. The advantages of each of these methods have not been evaluated in full detail, however all languages were evaluated. The ksr_5 results vary from 48% for French and Swedish to 51% for Dutch and 52% for German. Interestingly, in contrast to our results, the French version of *Fasty* scores worse than the German one.

The French project *PCA* is described by Blache & Rauzy (2007b). Its word predictor combines a large lexicon (320.000 words), a morphosyntactic based on *property grammars* (cf. Blache & Rauzy, 2006) and a user model, saving each phrase typed in a tree structure. This predictor arrives at a ksr_5 of nearly 50%, which is an advantage of approximately 4% over the unigram baseline ($ksr_5 \approx$ 46%). Moreover it was observed that the user model learns very quickly; after 2000 words the ksr showed an advantage of more than 2% over the baseline. This is in straight accordance with our results (cf. section 4.5.4, Table 4.4). Table 4.14 gives an overview on the different results reported above.

System/Reference	Prediction method	Language	ksr_5
Human (Lesher *et al.*, 2002)	No help condition	English	49%[14]
	Simple help (word list)		54%
	Adv. help (predictor)		59%
Profet (Carlberger *et al.*, 1997)	Bigram + cache + user lexicon	Swedish	33,1%
		English	37,3%
		Danish	32,9%
		Norwegian	35,6%
HandiAS (Maurel & Le Pévédic, 2001)	PFST	French	39,2%
	Adapt. PFST + user lexicon	French	43,5%
SibyMot (Schadle *et al.*, 2004)	Trigram	French	55,8%
	Trigram + Chunk model		57,1%
FASTY (Trost *et al.*, 2005)	Bigram + PoS-Trigram + grammar model + compound prediction + user lexicon	German	52,02%
		French	$\approx$ 48%
		Swedish	$\approx$ 48%
		Dutch	$\approx$ 51%
PCA (Blache & Rauzy, 2007b)	Unigram + user model + morphosyntactic pred.	French	50%

Table 4.14: Keystroke savings (ksr_5) reported by different works

[14]The results reported by Lesher *et al.* (2002) are based on a prediction list of 6 items (ksr_6).

4.6 Conclusion

This chapter started with the observation that the performance of language models depends to a considerable extent on the similarity of the training data to the actual task or register. Our experiments showed a performance loss of up to 20% on corpora from registers other than the training register; a deterioration, which has to be expected under real-world conditions as well.

In order to overcome the effects of training dependency, we presented three techniques to adapt an LM to the user's style of communication: a cache model, a user lexicon and a dynamic user model, interpolating a large static language model with a small LM trained on user input.

For the cache model, evidence has been provided that the effect of word recurrence is best modeled by a twofold, exponentially decaying distance function. As for the user lexicon, we presented a model which dynamically integrates new words and their usage frequencies with no need for user interaction, which is an important feature for an AAC system. Finally, the dynamic user model is also dynamically updated so that no further intervention is needed.

We then gave a detailed description of the data involved in training and testing as well as a presentation of the evaluation paradigm applied (*cross-register evaluation*), and in the last part of the chapter we evaluated each of these models in detail:

Whereas all three approaches proved to be beneficial in terms of keystroke savings, the advantages of the cache model and the user lexicon are moderate (from +0,2% to +2,7% gain of ksr_5). The dynamic user model however, showed important gains (up to +9,4%) for all corpora. From the superiority of the latter model we conclude that local syntactic information, as provided by the DUM, is of much more importance for prediction than simple lexical knowledge. Moreover, it could be shown that this model is very flexible: After being trained on as little as 2.000 words it performs considerably better than the baseline.

This chapter presented adaptation techniques, focusing on lexical and local syntactic information. A path that has yet to be pursued is the use of semantic information, acting on a more global level. This will be the subject of the following chapter.

Chapter 5

Semantic adaptation

J'ai cependant compris la poésie
de ces jeux de l'esprit le jour où,
comme j'entreprenais
de réclamer mes lunettes,
on m'a élégamment demandé
ce que je voulais faire avec la lune ...

JEAN-DOMINIQUE BAUBY
(*Le scaphandre et le papillon*, 1997)

While we have previously presented a number of techniques enabling a word predictor to adapt to the user, the focus of this chapter will be models and methods that exploit long-distance semantic dependencies and thereby adapt the predictor to the current semantic context. As in the previous chapter, we first motivate why this kind of adaptation is beneficial for a prediction task (5.1) and then give an overview on the different models making such an adaptation possible (5.2). In section 5.3 we present in detail the approach that we have chosen for our predictor. It makes use of Latent Semantic Analysis, a data-driven technique, based on the distributional properties of words in text, which is able to model such large-span semantic relationships as are missing in conventional n-gram models. After a description of different techniques of integration (5.4) in section 5.5 we present the results of an extended evaluation of our semantically adaptive prediction model.

5.1 Semantic information helps predicting words

In chapter 3 it was discussed that statistical language models normally exploit an extremely short context. This means that syntactic and semantic dependencies of longer distance cannot be described by such models. Especially semantic constraints however play an important role for the anchoring or interpretation of a message within a meaningful situational context. Humans make use of world knowledge for this semantic anchoring, they rapidly adapt their expectation to the current semantic preconditions.

In cognitive psychology this phenomenon has been studied extensively: In *semantic priming* experiments it was clearly shown that the recognition of a target word in a lexical decision task can be strongly influenced by the presentation of a semantically related prime; e.g. a word like *cat* is recognized much quicker, when a semantically close term like *dog* has been presented before. Psycholinguistic theories explain this effect by an activation spread of the semantic features of a recognized concept towards other concepts sharing similar features (cf. Plaut, 1995). The *spreading activation network* model by Collins & Loftus (1975) describes the mental lexicon as a semantic network consisting of concepts as nodes and semantic (or associative) relations as edges. When a concept is invoked, its node is activated and this activation spreads over the edges to related concepts that can then quicker be accessed.

This important property of human language processing is of course also highly relevant for prediction. When predicting a word, humans exploit semantic information to a large extent. It was found by Lesher *et al.* (2002) that the performance of humans in a word prediction task deteriorated strongly when they were provided with insufficient context information. This can be illustrated by a short example. Consider the following context:

Ex. 1 *in the first*

Here, it is nearly impossible to guess the following word. However, if more semantic information is provided, it can become much easier. Consider for example the following beginning of a phrase:

Ex. 2 *Whereas the game was quite aggressive in the first*

Here, humans would almost instantly predict a temporal quantity like *quarter* or *halftime* to follow. The first part of the sentence opens a semantic context in which words like *quarter* or *halftime* become much more likely than they would be otherwise, in other words, the context *primes* the occurrence of these terms. If the context is limited to the last two or three words (as in the first example), this can be achieved to a minimal extent only.

5.2 Using semantic information: Previous work

It was recognized very soon that the exploitation of semantic knowledge can help in a prediction task. For this reason we find semantically oriented approaches already in the first large-scale AAC projects. They were mostly based on structural or logical paradigms. Later, *trigger*-based approaches were presented that make use of word contiguities, determined on text corpora. Topic models represent a more document centered way to access semantics. In the following, these approaches will be presented and discussed.

5.2.1 Structural approaches

The first prediction methods that tried to incorporate semantic knowledge into an AAC system were based on structural paradigms. A straight-forward approach was presented in the work of Hunnicutt (1989). Here, a part of the lexicon was manually labeled with semantic categories such as *Living*, *Moving*, *Color* etc. The predictor was then adapted according to the semantic categories of the words appearing in the context; words belonging to the same category were given higher priority.

The KOMBE project (Guenthner *et al.*, 1992, 1993, cf. also section 2.4) integrated semantic knowledge in form of rules and a conceptual model. The rules link each lexical item to its semantic interpretation, which is based on the lambda calculus. For example a verb like *'chanter'* (to sing) is represented as:

$$\lambda x\ \texttt{chante}(x)$$

The semantic interpretation of a phrase like *Max chante* ('Max sings') is obtained by applying the semantic representation of the subject to the above formula:

$$\begin{aligned}&\lambda x\ \texttt{chante}(x)\text{: } [(\lambda p[p\ Max])\ (\lambda x\ \texttt{chante}(x)]\\ &= [(\lambda x\ \texttt{chante}(x))\ Max]\\ &= \texttt{chante}(Max)\end{aligned}$$

The conceptual model then assures the conceptual validity of the interpretations by imposing further constraints. It consists of a set of domains, a set of individuals and a set of relational symbols, from which formulas in first order logic can be built.

Since this model performs a rather deep semantic interpretation of the inserted message, it is able to exclude semantically implausible phrases such as "*La table chante*" ("The table sings"). However, a detailed evaluation considering keystroke reduction or communication speed was not published.

The semantic parsing module of the *Compansion* system (McCoy & Demasco, 1995, , cf. also section 2.4) is based on frame semantics (cf. Fillmore, 1968). Here, each verb is represented as a frame including one or several roles or cases to be filled, each of which has its own semantic constraints; the nouns of a phrase are then considered as case fillers with respect to the given verb frame. Typical cases are `AGEXP` (Agent/Experiencer), `GOAL` or `THEME`. For each case filler preferences can be specified, so that e.g. humans or animate objects are preferred over inanimate objects.

During composition the semantic parser starts with an empty frame, where all roles are represented but unfilled. After the verb was parsed all roles that cannot be filled are removed from the frame and the remaining roles are successively filled with the noun representations, taking into account the specified case filler preferences. For each possible filler configuration a heuristic value is calculated from the preferences, and in the end the interpretation with the highest value is retained. For example, the preferred frame semantic interpretation of the sentence *"The apple was eaten by John"* would look as follows (from McCoy & Demasco, 1995):

```
(25 DECL
    (VERB (LEX Eat))
    (AGEXP (LEX John))
    (THEME (LEX Apple))
)
```

The number in the first line corresponds to the computed overall preference value for this interpretation.

As before, a profound evaluation of this approach has not been published, therefore no conclusions about the applicability of this approach can be made. It is however obvious that a parser relying on frame semantics requires a large amount of hand-coded linguistic knowledge (each verb frame has to be defined), which is hard to achieve on a large scale. Projects like the Berkeley *FrameNet* (Ruppenhofer *et al.*, 2006), containing roughly 800 frames and 10.000 lexical units, could however provide a valuable resource here.

Likewise, other large-scale lexical or ontological databases, such as *WordNet* (Fellbaum, 1998) or *OpenCyc*[1], providing hand-coded semantic information on words, concepts and their relations, could be employed. It would be interesting to see to what extent this kind of semantic information can help in prediction, especially in comparison with the distributional approaches that will be presented in the following.

5.2.2 Trigger models

A rather different approach to exploiting semantic information for prediction is based on the distributional properties of words. In the last chapter (section 4.3.1) it was shown that words are not evenly distributed in a corpus but tend to occur in bursts; after a word has occurred one time it is more likely to reoccur than expected. This is however not only true for the word itself but for other, associated words; the occurrence of a word w_i can make the occurrence of a word w_j more likely. This distributional form of association can be calculated by an information-theoretic measure: *mutual information* (MI). This measure calculates the amount of common information in any two variables, but it can also be interpreted as a measure of dependence between these variables. It is calculated as follows:

[1]http://www.opencyc.org

$$MI(X,Y) = \sum_{x \in X} \sum_{y \in Y} P(x,y) \cdot \log_2 \frac{P(x,y)}{P(x) \cdot P(y)} \tag{5.1}$$

where X and Y are two random variables comprising $\{x, \neg x\}$ and $\{y, \neg y\}$ (i.e. presence and absence of the event x, y), respectively. Regarding the co-occurrence of two words in text, the joint probability $P(w_i, w_j)$ is calculated with respect to the frequency of co-occurrence of the events w_i, $\neg w_i$ and w_j, $\neg w_j$ in a pre-defined text window.

A closely related measure is *pointwise mutual information* (PMI), which is based on positive evidence only. Regarding the occurrence of two words w_i and w_j it is defined as follows:

$$PMI(w_i, w_j) = \log_2 \frac{P(w_i, w_j)}{P(w_i) \cdot P(w_j)} \tag{5.2}$$

Pointwise mutual information was introduced as a lexical association norm by (Church & Hanks, 1989). They showed that when the context window is sufficiently large (≥ 5 words), word pairs with a high PMI (which is measured in bits) are very often semantically or associatively related. This finding was confirmed in many other works since (cf. Brown *et al.*, 1992; Rosenfeld, 1996; Tillmann & Ney, 1997; Rapp, 2002). Table 5.1 shows some of the strongest PMI associations including *'doctor'* from (Church & Hanks, 1989), calculated on an *Associated Press* corpus of 15 million words; the size of the co-occurrence window was ± 5 words.

PMI	w_i	w_j
11,3	*honorary*	doctor
11,3	doctor	*dentist*
10,7	doctor	*nurse*
9,0	*examined*	doctor
8,9	doctor	*treat*
8,7	doctor	*bill*
8,7	doctor	*visit*
8,6	doctor	*hospital*
8,4	*nurse*	doctor

Table 5.1: Word pairs with strongest PMI values including *'doctor'*, from (Church & Hanks, 1989)

Regarding the terms occurring with *'doctor'* it becomes obvious that PMI is indeed capable of identifying semantically related terms (e.g. *'dentist'* or *'nurse'*), however these relations do not seem systematically related (e.g. by synonymy or hyponymy).

Many other distributional relatedness measures were developed since (e.g. the *log-likelihood ratio* (Dunning, 1993), χ^2 test or the *Jensen-Shannon divergence* (Dagan *et al.*, 1999)[2]). However, in a large-scale evaluation Terra & Clarke (2003) could show that MI and PMI perform best in identifying semantic relations in a range of different tests; interestingly, MI and PMI had very similar performance in all settings.

This kind of long-distance information can be beneficially exploited in many NLP tasks like speech recognition (cf. for example Brown *et al.*, 1992; Rosenfeld, 1996) or word prediction (cf. Matiasek & Baroni, 2003; Li & Hirst, 2005). Therefore, two words showing a high relatedness score (as MI) are considered as a *trigger-target* pair. Whenever the trigger word occurs in the context, the probability of the target word is enhanced. Tillmann & Ney (1997) have presented different methods to integrate trigger information to a stochastic language model. Here, only the single-trigger, linear interpolation method will be explained, basically following the idea of the cache model. For a given base probability $P_{base}(w|h)$ and a set of trigger-target pairs (a, b) the adapted probability is calculated as follows:

$$P^*(w|h) = \begin{cases} [1 - q(b|a)] \cdot P_{base}(w|h) + q(b|a) & \text{if } a \in h, w = b \\ [1 - q(b|a)] \cdot P_{base}(w|h) & \text{otherwise} \end{cases} \tag{5.3}$$

$q(b|a)$ is the trigger parameter controlling the influence of the trigger component. Depending on the model under consideration, it takes into account the association strength, the distance from the trigger, and, as Tillmann & Ney (1997) show, it can be optimized by an EM algorithm (cf. 3.4.1).

It has been made obvious that this model is very similar to the cache model, presented in the previous chapter (4.3.1). And indeed, as Rosenfeld (1996) shows (using mutual information), *self-triggers* (i.e. trigger pairs of the form (a, a)) are usually the most powerful triggers. He found for over 90% of the target words the self-trigger among the 6 strongest triggers. Thus, most of the time a trigger model will also incorporate cache information.

5.2.3 Topic-oriented models

While the trigger model introduces semantics at the word level, another family of approaches aims at integrating global, document-level semantics. Topic-oriented models assume that a document or message to be written has a certain theme or topic. When this topic can be identified, the underlying model can be adapted accordingly so that the topic-specific vocabulary and/or preferred phrase structure can be favored. While the basic idea of these approaches is not necessarily probabilistic, most works are based on stochastic language models.

[2]A good overview on different measures can be found on http://www.collocations.de/

The standard approach is to train a separate language model for each given topic and then to refer to the model (or models), which is (or are) most appropriate for the given topic at hand. In general, three major tasks have to be addressed by topic-based approaches:

1. determining a reasonable number and partitioning of topics
2. assigning each document of a given corpus to one (or more) of these topics
3. identifying for a given context the most appropriate topic(s)

Each of these tasks includes some difficulties. Considering that the definition of topic is a rather fuzzy one and cannot be based upon independent criteria, the identification of an optimal set of topics has to be a trade-off between adaptation capacity and the difficulty of assignment. The assignment itself is very hard to achieve for a larger set of documents when done manually. A number of automated topic-assignment approaches exists, but – as usual – they introduce error. For this reason many approaches refer to large corpora which are already topically assigned (this also replies to the first task). Such corpora exist now for English (e.g. the *Switchboard* corpus[3]), but they are (still) hard to find for other languages.

The third task is also not trivial, moreover it has to be performed on-line (i.e. during recognition or prediction). The current topic has to be identified from the context, and, since the topic can change, this procedure has to be updated regularly. A standard approach for topic identification is a *tfidf classifier* (cf. Seymore & Rosenfeld, 1997); here each topic is represented as a high-dimensional vector. The similarity between a topic and the current context is then determined by building a vector of *tfidf* values from the word (tf) and document frequencies (*df*) of the context and by calculating the distance between these two vectors. The most similar topic is then considered the topic of the context. Some approaches also use the k nearest topics to describe the current topic of a document (cf. Mahajan *et al.*, 1999). For an overview of different topic identification methods see the works of Bigi *et al.* (2001b,a).

Two works apply topic modeling for word prediction: While the work of Lesher & Rinkus (2001) focuses more on studying the theoretical benefit that can be achieved by topic models, the approach of Trnka *et al.* (2006) comprises a full model including identification and adaptation. Both works make use of the just mentioned *Switchboard* corpus, in which each document (telephone conversations) is assigned to one of 70 topics[4].

Trnka *et al.* (2006) present two methods to integrate topic information. For the first method (A) they compute a separate language model (bigram) for each of the 70 topics. The combined probability is then calculated as follows:

[3]http://www.ldc.upenn.edu/Catalog/readme_files/switchboard.readme.html

[4]Lesher & Rinkus (2001) constrain their study to the 20 most frequently occurring topics.

$$P(w|h) = \sum_{t \in T} P(t|h) \cdot P(w|t,h) \tag{5.4}$$

where t is one of the topics in the topic set T, and $P(w|t,h)$ is the probability assigned by the language model to t. The topic probability $P(t|h)$, serving as an interpolation coefficient, is calculated from the similarity value (cosine) of the given context h with t. In this way, each topic model contributes to the overall probability according to its similarity with the context. This means that the difficult problem of topic identification for a given context is reduced to a mere similarity problem; a given context is not forced to belong to one topic, the probability estimates result from a similarity-weighted combination of all topics.

The second method (B) presented by Trnka *et al.* (2006) uses topic-dependent unigram probabilities, which are again derived from the *Switchboard* corpus. These topic probabilities are then combined by multiplying them with the base model (also referred to as *geometric* interpolation, cf. section 5.4.3):

$$P(w_i|h) = \nu \cdot P_{base}(w_i|h) \cdot P_t(w_i)^{\alpha} \tag{5.5}$$

where ν is a normalization factor, $P_t(w)$ is the probability estimate of w_i belonging to topic t, and α is a tuning parameter. For optimization reasons however they omit the normalization.

5.3 Semantic adaptation with Latent Semantic Analysis

Each of the methods incorporating semantic information presented so far included some model-specific difficulties, making them suboptimal or inappropriate to apply for our purposes. The structural as well as the topic-oriented approaches rely on resources which are hard to obtain for languages other than English; the trigger model integrates semantic information only at a very local, context-independent level. Moreover, none of the approaches showed convincing and reliable gains for prediction performance. For these reasons, we want to present and employ in the following a rather different paradigm, which has shown remarkable success in a wide-spread range of tasks and problems: Latent Semantic Analysis (LSA).

5.3.1 Latent Semantic Analysis in NLP

Since the early 1990s, LSA has become a well-known technique in NLP and neighboring areas. When it was first presented by Deerwester *et al.* (1990), it aimed mainly at improving the vector space model in information retrieval (cf. Salton & McGill, 1983); for this reason it is also known as *Latent Semantic Index-*

ing. Its abilities to enhance retrieval performance are remarkable; results could be improved by up to 30%, compared to a standard vector space technique (cf. Dumais, 1995, TREC *routing* task).

This finding was the headstone for many subsequent researches. Moreover, it was soon recognized that LSA offers a rather universal paradigm to deal with semantics in NLP but also in psycholinguistics. The central concept of LSA is the vector space, in which every meaning-bearing element is (or can be) placed. Any two elements (be it words, sentences or full documents) can then be compared to one another; semantic similarity is represented by the distance of the corresponding vectors. An easy-to-read introduction to the geometric perspective of semantics is given by Widdows (2004).

And yet another aspect makes LSA very appealing for real-life NLP tasks: While structural approaches mostly rely on handcrafted semantic information, represented in logical formulae or frame-like structures, LSA derives semantic information from nothing but large amounts of text (i.e. no further manual knowledge engineering is involved).

Due to these two factors, the geometric perspective on semantics and the almost effortless construction of a knowledge base, LSA was applied in many different domains, as diverse as speech recognition, *e*-learning, and cognitive modeling. In the following a few examples of these works and their results are given:

Landauer *et al.* (1997) present an LSA-based scoring technique of student essays: They represent each of the essays (on heart anatomy) as a document vector in an LSA-based vector space. By measuring their distance to sample essay vectors (whose essays have already been scored by humans), they assign a score to each of the essays. The correlation between the LSA-based scores and those by human experts was 0,77, which was the same as the correlation between any two human experts. Based on these techniques a number of intelligent tutoring systems were developed, e.g. *AutoTutor* (Wiemer-Hastings *et al.*, 1999), or *SummaryStreet*, (Wade-Stein & Kintsch, 2003) for summarization training.

In (Landauer & Dumais, 1997) a theory of knowledge acquisition and representation is presented, assuming that the meaning induction mechanisms performed by LSA are very similar to those of humans. As an example task, LSA is applied to solve the *TOEFL*[5] synonym test, and it could be shown that the results of LSA are the same as those of the average foreign student passing the *TOEFL* (LSA: 64,4%; human participants: 64,5%). In (Rapp, 2003), a slightly different LSA model, trained on much more data, was able to solve even 92,5% of the synonym questions.

But LSA was also applied in more language-centered domains: Bellegarda (1997, 2000) as well as Coccaro & Jurafsky (1998) combine LSA information

[5]Test of English as a Foreign Language

with statistical language modeling; they report significant reductions in perplexity (up to 32% with respect to a bigram baseline). In (McCauley, 2004; Deng & Khudanpur, 2003) and (Pucher, 2007) LSA-enhanced language models are applied in several speech recognition tasks, their results were however not as clear. The major difficulty here seems to lie with the combination of a language model and information from LSA. This will be subject of section 5.4.

5.3.2 LSA: Method and formal background

The analysis is divided into two major steps, each of which will be explained in detail below:

1. Construction of a co-occurrence matrix (term$\times$document or term$\times$term)
2. Application of *singular value decomposition* and reduction of the term matrix to a lower dimensionality

The concept of a co-occurrence matrix has been developed in information retrieval. It displays the frequency of each term of the vocabulary in each document of the document set; therefore its size is determined by the number of terms in the vocabulary and the number of documents. It is obvious that such a matrix can become extremely large, and it can be assumed that it is very sparse, since usually only a small number of distinct terms (types) occur in a given document.[6] Still, such a matrix enables to quickly search and compare documents, only depending on the frequency counts of the terms they contain (*vector space model*, cf. Salton & McGill, 1983).

In its original form, as presented in (Deerwester *et al.*, 1990), LSA takes as input such a matrix, based on frequencies of terms in documents. However, due to the complex matrix computations that are performed subsequently, the amount of corpus data is very limited in such a setting. In addition the notion of document varies strongly over different corpora, no general criteria can be defined: a document can be only a small text section, a newspaper article, a telephone conversation or a book, which implies very different lengths and consistencies. For these reasons Schütze (1998) as well as Widdows (2004) propose the use of a different kind of matrix: a term$\times$term matrix stores the co-occurrence frequencies of terms within a predefined context window (similar to the co-occurrence window discussed in section 5.2.2). The two sets of terms need not be identical, one can also define a set of *index terms* $I = (i_1, \ldots, i_m)$, usually consisting of more specific content words. It is clear that such a matrix is much denser than a matrix based on documents; moreover its size is independent of the size of the training data, therefore much larger corpora can be used for training.

[6] In Wandmacher (2005) a term$\times$document-matrix was used that had less than 0,08% non-zero elements.

After the matrix is filled with the co-occurrence frequencies, in principle no further processing is needed in order to compare terms to one another; each term is represented by a vector (of size $|D|$ or $|I|$). The distance of these term vectors can then be calculated by one of the usual vector distance measures (e.g. Euclidean distance, city-block metric, scalar product etc.[7]). Most works on LSA however apply the cosine measure (cf. for example Deerwester *et al.*, 1990; Landauer & Dumais, 1997; Coccaro & Jurafsky, 1998; Schütze, 1998). The cosine of the angle between any two vectors $\vec{w_i}$ and $\vec{w_j}$ of dimensionality m with components w_{ik} und w_{jk}, $k \leq m$ is defined as follows:

$$\cos(\vec{w_i}, \vec{w_j}) = \frac{\sum_{k=1}^{m} w_{ik} w_{jk}}{\sqrt{\sum_{k=1}^{m} w_{ik}^2 \sum_{k=1}^{m} w_{jk}^2}} \tag{5.6}$$

The cosine has the important advantage that it normalizes for length (by the denominator); in this way frequency influences are leveled out. In addition the result becomes standardized ($[-1;1]$), which facilitates further comparisons.

The comparison of co-occurrence vectors of terms is also referred to as *second order* co-occurrence (cf. Grefenstette, 1994; Rapp, 2002); here, similarity is not established because two terms occur with one another, but because they occur with similar terms, they have similar environments. This is an important difference, because two terms can show a very high second-order similarity, even though they do not co-occur. Moreover, while a high first-order similarity mostly describes syntagmatic relations, second-order similarity seems to display more semantic relatedness between terms (cf. Rapp, 2002; Wandmacher, 2005); this is however only a tendency, apart from the one mentioned above no clear-cut criteria can be determined.

Second-order similarity, determined on such a raw matrix, can already yield interesting results. The *Hyperspace Analogue to Language* (HAL) model by Lund & Burgess (1996) for example is based on this kind of matrix. However, since such a matrix is very large, vector comparisons can take a considerable amount of time. Another weakness of this model is that such co-occurrence frequencies are rather noisy. The proponents of LSA assume that the important semantic structures are hidden (or *'latent'*) behind many arbitrary co-occurrences. To reveal these structures the weak relations have to be eliminated.

To achieve this, *Singular Value Decomposition* (SVD) is applied to the input matrix. This procedure is similar to *principal component analysis*, but, unlike PCA, it can be performed on any type of matrix (cf. Berry *et al.*, 1999). The

[7]A good overview on different distance measures and their effects in information retrieval is given by Salton & McGill (1983).

main achievement of an SVD is to enhance the contrast between weak and strong components in a matrix; for this reason SVD is also applied in signal processing, image compression and numeric meteorology. In this way noisy elements are filtered out and important connections are strengthened. Furthermore, the matrix size is reduced from several thousand to a few hundred dimensions, making vector comparisons much quicker. In the following a formal definition of *singular value decomposition* is given:

Let A be a matrix $\in R_{m\times n}$, of rank r[8]. The *singular value decomposition (SVD)* of A is then defined as:

$$SVD(A) = U\Sigma V^T \tag{5.7}$$

where $U^TU = V^TV = I_n$ (Orthogonality condition) and $\Sigma = diag(\sigma_1, ..., \sigma_r)$, $\sigma_i > 0$ for $1 \leq i \leq r$, $\sigma_j = 0$ for $j \geq r+1$ and $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r \geq 0$.

The matrices $U_{m\times r}$ and $V_{n\times r}$ consist of the eigenvectors of the columns (in U) and rows (in V) of A. They are orthogonal, i.e. their vectors are linearly independent. Σ is a diagonal matrix containing the singular values σ_i of A in descending order. The product of these three matrices is again A.

The decisive step now consists of mapping the original matrix into a subspace, i.e. a space of lower dimensionality. This is done by eliminating the $r - k + 1$ lowest singular values and by remultiplication of the cropped matrix of singular values σ_k with U and V. The resulting matrix A_k is the best *least squares* approximation of A in a k-dimensional space, i.e. the euclidean distances between all vectors of A were preserved as much as possible. Figure 5.1 gives a schematic overview of an SVD.

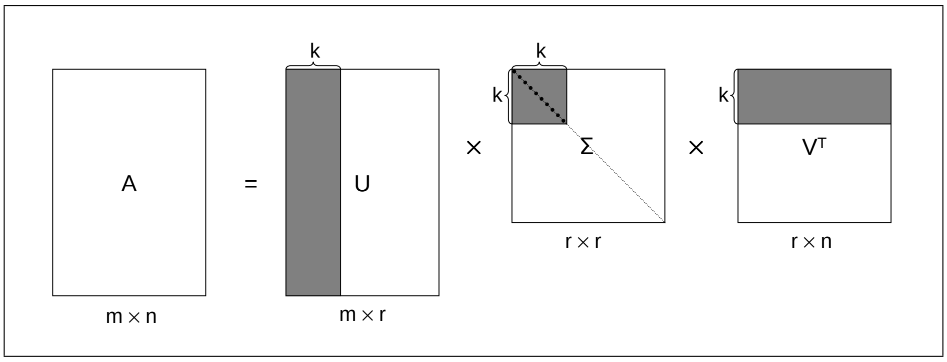


Figure 5.1: Singular value decomposition (SVD) of a matrix A and reduction to k dimensions

[8]The $rank$ of a matrix equals the number of its linearly independent columns or the number of non-zero eigenvalues of $A^T \times A$. Often r is simply $\min(m, n)$.

If only the term vectors are needed (e.g. for determining the semantic relatedness of terms), it is not necessary to rebuild the matrix A_k. It is then sufficient to crop the vectors of the row matrix U to k dimensions and use these k-dimensional vectors for comparison.[9]

A difficult aspect however is to determine the optimal number of dimensions k to which the matrix will be reduced: If the reduction is too high, the vectors cannot anymore be properly placed in the resulting space, and important relatedness information gets lost. If the reduction is too low, spurious relations are not sufficiently filtered, the desired contrast enhancement does not arise. In most works the factor k is heuristically set to 100 – 400 dimensions (cf. Deerwester *et al.*, 1990; Dumais, 1995; Kintsch, 2000). Landauer *et al.* (1998) have investigated this question on the *TOEFL* synonym test (mentioned above): Their performance curve shows a quick and straight rise from 2 to 100 dimensions; it then oscillates around a peak of 300 and falls slowly afterwards (cf. Figure 5.2). Interestingly, the percentage of correct answers for the unreduced model (full dimensionality) was approximately the same as with a reduction to 2 dimensions.

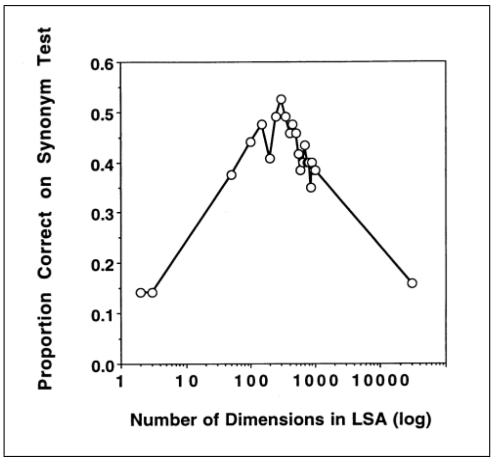


Figure 5.2: Performance curve on the TOEFL synonym test with different dimensions ($2 \leq k \leq 30.743$) (from: Landauer *et al.*, 1998, p. 23)

These results suggest that the application of the SVD indeed plays an important role for the uncovering of relations within the matrix. The effects of the SVD on term (and document) similarity has been studied in the works of Story (1996) and Wiemer-Hastings (1999) as well as by Kontostathis & Pottenger (2003, 2005). Their overall conclusion is that the SVD-based dimensionality reduction indeed contributes in an important (and beneficial) way to the

[9]When *Lanczos*-type algorithms (cf. Lanczos, 1950) are used for calculating the SVD, not more than k dimensions will be calculated anyway. These algorithms represent a very efficient way to calculate an SVD on large, sparse matrices. An overview on different public-domain implementations of *Lanczos*-algorithms can be found in (Berry, 1997).

quality of the analysis.

The vector space resulting from a dimension-reduced co-occurrence matrix is often referred to as a *semantic* space (cf. for example Landauer *et al.*, 1998; Kintsch, 2000); Schütze (1998) as well as Widdows (2004) prefer the term *word space*. Figure 5.3 summarizes the different steps involved in its construction (here for term×term-matrices).

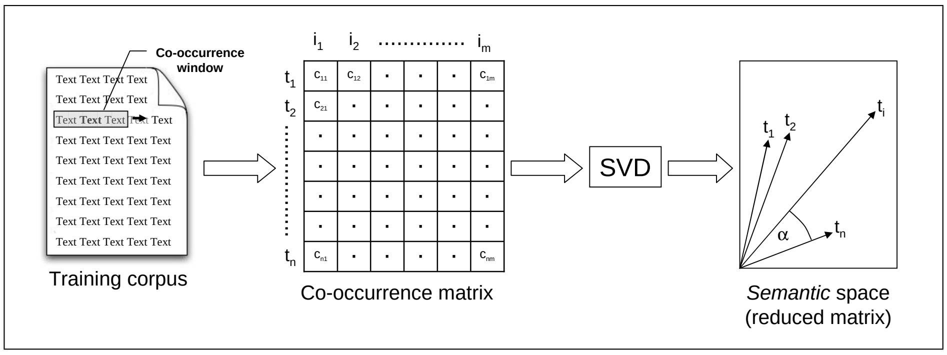


Figure 5.3: Schematic overview of an LSA transformation

5.3.3 *Excursus*: Measuring semantic relatedness

It has become obvious in the previous sections that capturing semantic relatedness is crucial for the success of a semantic adaptation method. In the past years a considerable number of different semantic relatedness measures have been developed, many of which are based on hand-crafted lexical-semantic nets like the Princeton *WordNet* (cf. Fellbaum, 1998). Most of these methods make use of the path distance that is measured on the hyponym tree of such a word net, some in combination with frequency information. Typical representatives of these measures are the ones proposed by Wu & Palmer (1994), by Hirst & St-Onge (1998), Lin (1998), Resnik (1995) or by Leacock & Chodorow (1998). In the following only the *Leacock-Chodorow* measure will be exemplarily presented; a more detailed overview of all measures can be found in (Cramer & Finthammer, 2008).

The *Leacock-Chodorow* measure is based solely on the structure of the hyponym tree of a lexical-semantic net; it measures the length of the shortest path (in intervening nodes) between two terms (or their respective synonym sets) and scales this value by the maximum depth of the complete tree. Equation 5.8 presents the details of this measure:

$$Sim_{LC}(w_1, w_2) = -\log \frac{2 \cdot sp(w_1, w_2)}{2 \cdot D_{max}} \tag{5.8}$$

w_1 and w_2 are any two terms described in the lexical-semantic net,

$sp(w_1, w_2)$ is the length of the *shortest path* between w_1 and w_2 on the hyponym structure, and D_{max} represents the maximum depth of the net (= length of the longest hyponym path).

Considering the multitude of these measures the question arises how well they are able to capture semantic relatedness, and how LSA or other corpus-based measures (like *pointwise mutual information*, cf. equation 5.2) perform in comparison.

In collaboration with Irene Cramer and Marc Finthammer (Universität Dortmund, Germany)[10], we have conducted an extended evaluation of many relatedness measures, including PMI and LSA.

We created two test sets: Set A comprised 320 manually assembled German word pairs, out of which 100 were randomly selected. Set B consisted of 500 word pairs, which were automatically compiled from several resources (10% collocations, 10% associations from a thesaurus, 80% randomized). The pairs of these test sets were rated by human annotators (German native speakers) on a 5-level scale with respect to their semantic relatedness (Set A: 35 participants, Set B: 75 p.). The subjects were asked to base the rating on their intuition about any kind of conceivable semantic relation between the two terms.

In addition semantic relatedness of each word pair was determined using each of the following methods: *Wu-Palmer*, *Hirst-St-Onge*, *Lin*, *Resnik*, *Jiang-Conrath*, point-wise mutual information (PMI, cf. equation 5.2), based on hit counts from *Google*, as well as LSA (cosine)[11]. For the application of the tree-based measures *GermaNet* (cf. Hamp & Feldweg, 1997), the German version of *WordNet* was utilized. After calculating the semantic relatedness for all measures, the correlation (*Pearson*) of each of them with the average human ratings was determined. Table 5.2 shows the correlation coefficients for all measures applied. All correlations were significant at a level of $< 0,01$.

Test set	Wu & Palmer	Leacock & Chodorow	Lin	Hirst & St-Onge	Resnik	*Google* PMI	LSA cosine
A	0,36	0,48	0,48	0,47	0,44	0,37	**0,62**
B	0,21	0,17	0,27	0,32	0,24	0,35	**0,59**

Table 5.2: Correlations (*Pearson* coeff.) of several semantic relatedness measures and human estimations

Regarding the results in Table 5.2 we can see that for both test sets the LSA measure shows the highest correlation with the human ratings. Not only the net-based measures, but also PMI were outperformed in a significant way.

It is clear that the results reported depend on many factors, for example

[10] Most of the work reported here is to be attributed to Irene and Marc. More on the objectives and methods of this evaluation can be found in (Cramer & Finthammer, 2008).

[11] The semantic space was calculated as described in section 5.5.1

the net-based measures are strongly dependent on the quality of the lexical-semantic net applied. We therefore do not want to enter into a deeper discussion about the differences between the measures evaluated or to draw conclusions about their general capacities. It can however be concluded that LSA is able to determine semantic relatedness, and it seems to perform more human-like than other measures, which is a crucial precondition for its employment in a prediction task.

5.3.4 Representing the context in a vector space

In section 5.3.2 it was mentioned that LSA enables to compare any two textual elements with one another via vector distance measurement in the semantic space. So far however we have only a vectorial representation of the vocabulary. How can larger units be represented in such a space?

The mostly applied method is to construct a context vector by summing up the term vectors of all the terms it comprises (i.e. forming the centroid of these term vectors). This vector will have the same dimensionality as all term vectors, so that the usual distance measurements can be used:

$$\vec{h}_1^m = \sum_{i=1}^{m} \vec{w}_i \tag{5.9}$$

It is of course an highly simplifying assumption that the semantics of larger constituents is additive; logical or syntactic constraints like quantification and negation cannot be represented in such a way.[12] However, vector addition is by far the most common method to represent the meaning of larger constituents in a semantic space (cf. Landauer *et al.*, 1998; Foltz *et al.*, 1998; Kintsch, 2000), and its application is straightforward and quick. It will therefore be applied henceforth in order to build up a context vector.

Let us look at two examples (in Table 5.3) illustrating how illustrate how LSA based information can contribute to predictive purposes; they were calculated using our own semantic spaces (to be specified in section 5.5.1). Here, the context vector was built by adding the respective term vectors, and the nearest neighbors (term vectors) were determined.

Considering the nearest LSA-neighbors for the two examples, it can clearly be seen that these terms are all semantically related to the respective context. Moreover, it becomes obvious that the terms with the highest similarity values (cosine with the context vector) are those that have actually occurred in the context (cf. *'game'* or *'mathematics'*). This is expected; as the context vector contains each of the corresponding term vectors, it will therefore be quite similar to them. And with regard to prediction this is not an unwanted effect, since, as

[12] Widdows (2004, ch. 7) however proposes an interesting idea to represent logical constraints in a vector space by means of quantum logic.

rank	NN	cos-val	rank	NN	cos-val
1.	*game*	0,541	1.	*professor*	0,352
2.	*kick*	0,534	2.	*mathematics*	0,324
3.	*offside*	0,374	3.	*dad*	0,302
4.	*pass*	0,373	4.	*taught*	0,257
5.	*tackles*	0,276	5.	*mathematician*	0,166
6.	*upfield*	0,263	6.	*father*	0,159
7.	*volley*	0,250	7.	*mathematic*	0,143
8.	*touchline*	0,153	8.	*grand-father*	0,142
9.	*referee*	0,123	9.	*sciences*	0,111
10.	*pitch*	0,115	10.	*teacher*	0,097
h =	*"The game was nearly over when the ball"*		*h* =	*"My dad was a professor in mathematics and I think that"*	

Table 5.3: 10 nearest LSA-neighbors for two example contexts

we have discussed above, *self-triggers* are usually the best semantic predictors (cf. again Rosenfeld, 1996).

However, these examples also show the drawbacks of a (purely) LSA-based prediction model: While it seems that content-related words can indeed well be predicted by LSA, the presence of function words as well as local morpho-syntactic constraints and word order are totally ignored. This means that we have to combine information from a standard prediction approach (e.g. a statistical language model) and LSA in order to take advantage of both models. However, due to the strong variability of the LSA performance on prediction, this turns out to be a non-trivial problem, as will be seen in the evaluation part (section 5.5).

5.4 Integrating LSA information with a language model

When we started to work exploiting LSA-based similarities for word prediction, we quickly realized that the application of brute-force methods would not lead to positive results. We thus developed and implemented a number of different approaches in order to arrive at a better understanding of the capacities (and incapacities) of the model itself (while of course we do not want to claim that the optimal solution has been found). The integration approaches presented in the following are: an LSA-based *trigger* model, a *n-best reranking* approach, and also different kinds of interpolation.

While these approaches are inherently rather different, they all operate on the model level (cf. section 4.2). Béchet *et al.* (2004) present an LSA-based *data augmentation* method which aims at augmenting frequency counts, based on semantic similarity with the current context. However, as explained in the section on MAP adaptation (4.2.1), this kind of approach requires that the raw counts be stored and all probability calculations (including smoothing and truncation) be performed on-line. In order not to further increase the computational load of our model, we did not try out this kind of method nor did we test MDI techniques (cf. section 4.2.2). Such approaches should however be subject to further investigation.

5.4.1 LSA-based triggers

As presented in section 5.2.2, the trigger model incorporates a set of semantically related trigger-target pairs; as soon as a triggering word is seen in the context, the occurrence probability of the respective target word is raised. The classical trigger models (cf. Brown *et al.*, 1992; Rosenfeld, 1996) make use of a first-order co-occurrence measure like pointwise mutual information to calculate the association strength, but, due to the easy comparison of any two term vectors in an LSA-based space, this idea can be transferred without any further effort: For a given trigger word a set of targets is calculated by determining the nearest LSA neighbors having a similarity value of more than a given threshold θ (θ usually $= 0,2 - 0,4$). Using these pairs the base model can then be adapted according to the schema given in 5.3.

Since the trigger model is based on the cache model, it is reasonable to weight the influence of the target according to the distance of the trigger and the word to be predicted: If the recurrence probability of a word w_i depends on its position in the cache, it can be assumed that the words triggered by w_i also depend on it. The schema presented in equation 5.3 can then easily be adapted by defining the trigger parameter $q(b|a)$ as follows:

$$q(b|a) = \begin{cases} \beta \cdot Sim_{LSA}(a,b) \cdot Cf(d_a) & \text{if } Sim_{LSA}(a,b) > \theta \\ 0 & \text{otherwise} \end{cases} \tag{5.10}$$

The LSA similarity is calculated from the cosine of the corresponding term vectors $\vec{a}$ and $\vec{b}$, with negative cosine values being set to 0. β is a constant scaling factor controlling the influence of the trigger component. The *cache factor* $Cf(d_a)$ is calculated as explained in section 4.3.1; the decay function is repeated here for convenience:

$$Cf(d_a) = \begin{cases} \nu \cdot \frac{d_a^\alpha}{\mu^\alpha} & \text{if } d_a \leq \mu \\ \nu \cdot \mathrm{e}^{\gamma \cdot d_a} + \delta & \text{if } d_a > \mu \end{cases} \tag{5.11}$$

with d_a being the distance of the trigger word a to the current position in the text and α, γ, δ and μ being constants scaling the optimal recurrence function.

While the trigger model is probably the most straightforward idea to incorporate LSA information for prediction, it only makes use of term-term similarities; the particular property of LSA, making similarity measurements between terms and contexts possible, is not exploited. This however is performed by the following methods.

5.4.2 Partial reranking

In section 5.3.4 it was argued that while LSA can serve as a predictor for content words being semantically related to the context, it is unable to predict function words, and it is also insensitive to the frequency of word usage. The underlying idea of partial reranking is therefore to constrain the effect of the LSA prediction by considering only the best m candidates (i.e. the terms with the m highest probabilities) from the base model for LSA: The m best terms are reranked according to their LSA similarity with the current context. In this way the model is prevented from making totally implausible predictions, and the probability of function words (which are not represented in the vector space) is preserved.

The method can be formalized as follows: For a given context h we calculate from our base model the ordered set $Best_m(h) =< w_1, \ldots, w_m >$, so that $P(w_1|h) \geq P(w_2|h) \geq \ldots \geq P(w_m|h)$. We then calculate for each member in $Best_m(h)$ its *reranking probability* P_{RR} by determining its LSA similarity with the context:

$$P_{RR}(w_i|h) = \begin{cases} \nu \cdot \beta \cdot P_{base}(w_i|h) \cdot Sim_{LSA}(w_i, h) & \text{if } w_i \in Best_m(h) \\ \nu \cdot P_{base}(w_i|h) & \text{otherwise} \end{cases} \tag{5.12}$$

where $Sim_{LSA}(w_i, h)$ determines the cosine similarity of $\vec{w_i}$ and the context vector $\vec{h}$.[13] β is again a scaling constant controlling the influence of the LSA component, and ν represents a normalizing factor to assure that all values sum up to 1. Optimal values for m obviously depend on the overall size of the vocabulary; it can be assumed that a value of 5 to 10% of the total vocabulary represents a reasonable size.

5.4.3 Linear and geometric interpolation

Interpolation is a very important tool for integrating information from different resources; many models presented in the last chapters rely on it. In a probabilistic framework however, interpolation requires that the resources to be combined provide probability-like estimates. While our base model fulfills this requirement, we have to transform the LSA estimates.

[13]As for the trigger model, Sim_{LSA} crops negative cosine values at 0 to avoid negative probabilities.

This can be achieved in a straight-forward manner by standardizing and normalizing the cosine similarities between each word vector $\vec{w_i}$ and the vector of the current context $\vec{h_i}$ so that the estimates for all words again sum up to 1. This *pseudo*-probabilistic distribution is however rather flat; its dynamic range is very low. For this reason a contrasting (or *temperature*) factor γ is normally applied which raises the cosine value to some power (normally $\gamma = 3 - 8$) and thereby stretches the distribution. The probability-like LSA estimate for a word w_i given a context h is then calculated as:

$$P_{LSA}(w_i|h) = \frac{\left[(\cos(\vec{w}_i, \vec{h}) - \cos_{min}(\vec{h})\right]^{\gamma}}{\sum\limits_{k=1}^{v} \left[\cos(\vec{w}_k, \vec{h}) - \cos_{min}(\vec{h})\right]^{\gamma}} \tag{5.13}$$

where $\cos_{min}(\vec{h})$ returns the lowest cosine measured for h; it is needed to make all values non-negative.

Pseudo-probabilities calculated in this way can then be interpolated with estimates from a base model. However, as mentioned above, the estimates for the two models are very heterogeneous; the LSA model neglects the presence of function words, and it often overestimates very rare words. A linear combination therefore tends to level out the single models, which means that the deficiencies of the LSA estimation are partly transferred to the combined model. For example, when we consider the trigrams "*the student sleeps*" and "*the student university*", due to the strong semantic relatedness between '*student*' and '*university*' a linear combination with an LSA model would lower the probability of the first and raise the probability of the second, which is however ungrammatical.

To reduce this unwanted effect, Coccaro & Jurafsky (1998) propose to perform the integration by means of *geometric interpolation* (cf. also Klakow, 1998). While linear interpolation simply adds the (weighted) probabilities, geometric interpolation multiplies them, weighting is performed by an exponential coefficient:

$$P_{GI}(w_i|h) = \frac{P_a(w_i|h)^{\lambda_a} \cdot P_b(w_i|h)^{\lambda_b}}{\sum\limits_{k=1}^{v} P_a(w_k|h)^{\lambda_a} \cdot P_b(w_k|h)^{\lambda_b}} \tag{5.14}$$

Because the sum of all geometrically combined estimates deviates from 1, the values have to be renormalized (in the denominator), which is computationally more demanding. However, the agreement of the single models is taken into account; the combined estimate is only high if the singular estimates have similar values.

5.4.4 Confidence weighting

In the interpolation variants presented so far, the coefficients controlling the influence of each component were kept static at a given point of time. This is however not a necessary condition of the model; the coefficients can be adapted according to the expected success of each model for each word (requiring of course a renormalization afterwards).

It was already mentioned that the prediction capacities of LSA depend strongly on the semantic properties of a term. While it does well at predicting specific content words, it is unable to predict more general terms, having a rather uniform distribution over a context[14]. For this reason is is reasonable to modify the interpolation parameters with respect to the term under consideration; yet how can the predictive success of the LSA component be estimated?

Coccaro & Jurafsky (1998) propose the use of entropy as a confidence measure, which is also often applied as a term weighting scheme in information retrieval. It can indeed be argued that low-entropy words are more specific than words occurring in many different contexts.

However, as shown in (Wandmacher, 2005), entropy does not correlate significantly with the relation quality of a word in an LSA space; other measures of specificity like the (already mentioned) *tfidf* measure do not correlate very strongly ($r = 0,32$). A higher correlation however was found for the mean distance of the nearest LSA neighbors of a word, i.e. the *density* of a word cluster ($r = 0,56$). For this reason we here propose a density-based confidence measure. We define the cluster density D_m for a term w_i with respect to its m nearest neighbors as follows:

$$D_m(w_i) = \frac{1}{m} \sum_{j=1}^{m} Sim_{LSA}(w_i, NN_j^i) \tag{5.15}$$

where NN_j^i represents the j'th nearest neighbor to word w_i and Sim_{LSA} again reflects the (standardized) cosine between the vectors of w_i and NN_j^i. An example motivating the idea behind this confidence measure is given in Figure 5.4.

A rather specific term like '*plane*' has strongly related nearest neighbors (like '*jet*' or '*flight*'), the average distance is rather high (0,57). The nearest neighbors of a more unspecific term like '*introduce*' however are less related, the density of its 10-word cluster is lower (0,26).

While cluster density can serve as an estimate for the prediction quality of the LSA component, it still has to be scaled in order to serve as a coefficient for interpolation, otherwise the influence of the LSA component can get too large (e.g. for our vocabulary D_{max} was 0,95). By empirical testing we found

[14]Coccaro & Jurafsky (1998) call them *promiscuous* words

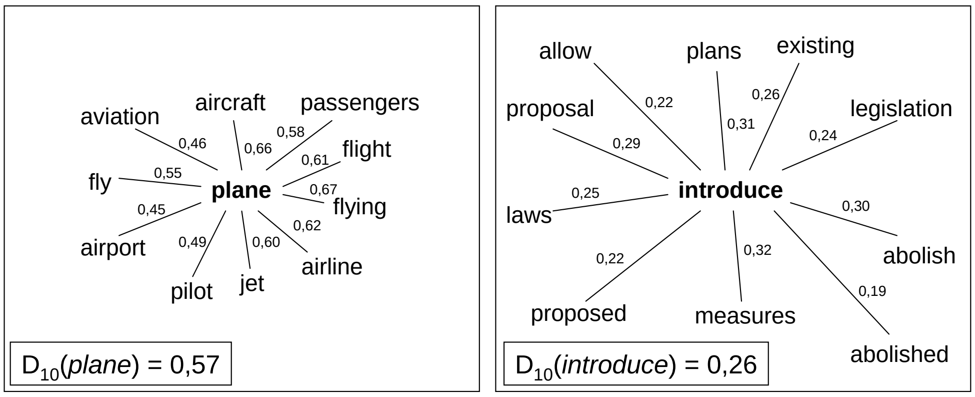


Figure 5.4: Examples of terms having high (*'plane'*) and low (*'introduce'*) cluster density and their 10 nearest LSA neighbors

reasonable sizes for the scaling factor β of 0,2 – 0,4. All function words which are not represented in the LSA space will receive a default value of $\beta = 0$, so that they are estimated by the base model only.

5.5 Evaluation: Method and results

For the evaluation of the semantic adaptation component we make use of basically the same evaluation method as for the user adaptation models (cf. section 4.4). We first introduce the training and test data and then take a closer look at some of the parameters that have to be optimized in order to construct an optimally performing LSA model. We then compare exhaustively all of the integration methods that were just presented and finally discuss the overall results.

5.5.1 Training and test corpora

It was already mentioned that LSA requires large training corpora in order to reliably estimate its co-occurrence statistics. And as discussed in the previous chapter (4.4), an easy and reliable way to acquire large amounts of general language text is to use newspaper corpora, which – despite their well-known deficiencies – provide sufficiently general and modern language samples, covering a multitude of different semantic contexts. For reasons of quality and missing control we were again reluctant to use web-based corpora.

The resources that we use for training are from the same origins as the data described in section 4.4, however we used significantly larger amounts of text (more than 100 million token per language). The vocabulary is limited to

80.000 words for all models; words having a corpus frequency of less than 20 as well as closed-class (function) words, bearing only little semantic information, are excluded. Table 5.4 gives an overview of the different corpora used for training:

	Origin (years)	Nb. of token	Vocab. size
French	*Le Monde* (1996-1999)	$\sim 100 \times 10^6$	80.000
German	*Die Tageszeitung* (1997-1999)	$\sim 101 \times 10^6$	80.000
English	*The Guardian* (1997-1998), *The Times* (1993, 1995)	$\sim 108 \times 10^6$	80.000

Table 5.4: Training corpora and sizes used for the construction of the co-occurrence matrix

For further processing we employed the *Infomap* toolkit[15]. Using this toolkit we could calculate co-occurrence matrices on our given training corpora, and we could also reduce them by singular value decomposition, based on a *Lanczos* algorithm (cf. Berry *et al.*, 1999).

Concerning evaluation we again followed the *cross-register* paradigm, as explained in the previous chapter (section 4.4), in order to get a realistic picture of the performance of our models. And to assure comparability with the results presented in the previous chapter, we used the same test corpora as before: text from newspaper (*news*), literature (19th century novels; *lit*), transcribed speech (dedicated dialogue; *speech*), and personal e-mail correspondence (*e-mail*); these corpora were described in more detail in the previous chapter (Table 4.4.3). For purposes of parameter optimization however we only used the newspaper test corpora (*news*).

In order to have a common reference point to compare our results to, we also made use of the same baseline (4-gram model, trained on newspaper data) as before. It has been described in section 4.5.

5.5.2 Parameter setting

The parameter space of an LSA-based model is quite large; there are at least a dozen of parameters influencing the performance (e.g. corpus size, matrix size, similarity measure etc.). Since a profound examination of each of them would surely merit a thesis of its own, we had to rely on results from other works and on our own experience with LSA-based models. We did however take a closer look at three of the most important parameters: the size of the co-occurrence window, the degree of dimensionality reduction, as well as the amount of context to take into account.

[15]Version 0.8.6, cf. http://infomap-nlp.sourceforge.net. Many thanks to the authors for making their tool available!

Co-occurrence window

The size of the co-occurrence window is a crucial factor for establishing semantic relatedness, since it exerts a direct influence on the input matrix. Previous works employed rather small, roughly sentence-length windows around the target word; Lund & Burgess (1996) used windows of ± 8, Cederberg & Widdows (2003) used ± 15 words. Such values however seemed to perform sub-optimally in our case. We therefore calculated spaces for varying window sizes from ± 10 to ± 120 words. To better investigate small effects, we excluded closed-class words from the ksr calculation (such words were however still present in the context), which obviously stretches the ksr advantages. As integration method (static) geometric interpolation ($\lambda_{LSA} = 0,08$) was applied. Figure 5.5 shows the ksr_5 results for different sizes of co-occurrence windows; the test was performed on the English newspaper corpus (*news-en*).

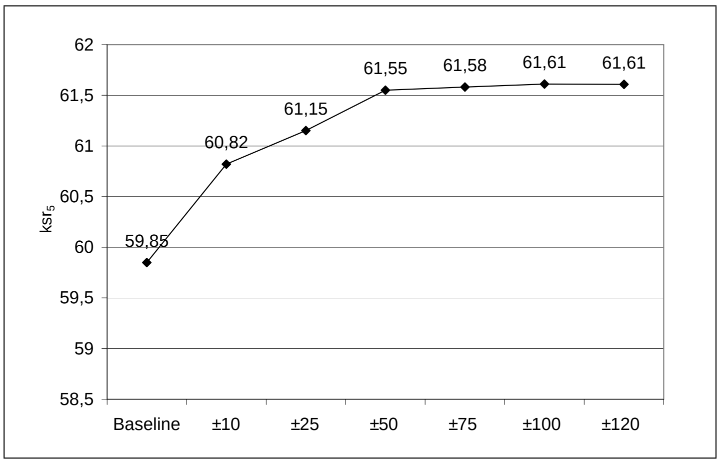


Figure 5.5: Results (ksr_5) for different co-occurrence windows (± 10 - ± 120); tested on *news-en*

The trend of the curve in Figure 5.5 is quite clear: The performance gain grows with the context size, but it seems to reach a level at ± 100 words[16].

Interestingly, this result is in accordance with results that we obtained on a very different task: In (Wandmacher *et al.*, 2008) we applied LSA for predicting human associations. Here as well we could see that the largest co-occurrence windows performed best.[17]

[16]Due to computational constraints we were not able to compute models for window sizes larger than ± 120 words.

[17]Since we used in this work a higher dimensionality, we were not able to extend the co-occurrence window over more than ± 75 words.

Dimensionality

The other important factor in the construction of a semantic space is the SVD-based dimensionality reduction; as already mentioned in section 5.3.2, the reduction is beneficial for the geometric representation of the vectors, since it eliminates spurious influences, it can however also harm if too many dimensions are removed. In the case of word prediction another factor comes into play: the processing time of vector comparisons grows linearly with the number of dimensions. We therefore have to trade off prediction success against processing time.

Figure 5.6 shows the performance as well as the time curve (runtime per prediction)[18] for dimensionalities from 10 to 300[19]; the evaluation was performed on *news-en*, the LSA model was integrated by geometric interpolation ($\lambda_{LSA} = 0,08$).

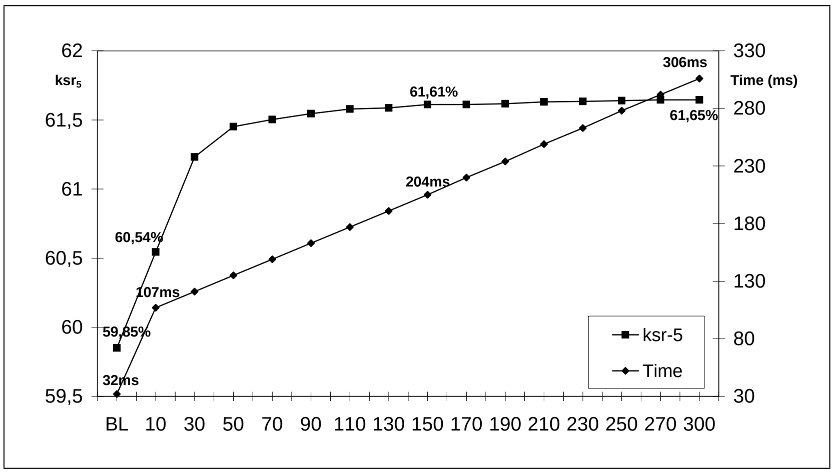


Figure 5.6: Results (ksr_5 and runtime per prediction) for different dimensionalities k ($k = 10 - 300$); tested on *news-en*

The performance curve (ksr_5) in Figure 5.6 basically follows the first part of the (logarithmized) curve in Figure 5.2: The performance grows rapidly in the beginning and then flattens, but we cannot observe a peak before 300 dimensions. The processing time however grows linearly from 107 (10 dimensions) to 306 milliseconds per prediction at 300 dimensions. A delay of more than 300ms has already a distracting effect on the user, the prediction process is not anymore perceived as smooth. In this respect a reasonable trade-off would be a dimensionality of 150 dimensions, saving roughly a third of the processing time (204ms) while loosing only minimally in keystroke savings.

[18]The tests were performed on an Intel *Dual Core* processor with 1,83Ghz, 2GB RAM.

[19]Again, due to computational restrictions, we could not calculate spaces of dimensionalities higher than $k = 300$.

Context size

A last parameter that we have investigated concerns the maximum amount of context from which the context vector will be built. It can be assumed that if the context size is too short, long-distance dependencies (at which our model actually aims) are not considered anymore. If the size is set too large, semantically irrelevant terms may be included; moreover, the influence of each term decreases with a growing size. In order to find a near-optimal value, we tested on context sizes from 1 to 120 content words. This time however the performance (in terms of ksr_5) was measured on all words in the test corpus (*news-en*), since it can be assumed that this parameter also effects closed-class words. Figure 5.7 shows the performance curve with varying context size (the integration was performed using CWGI).

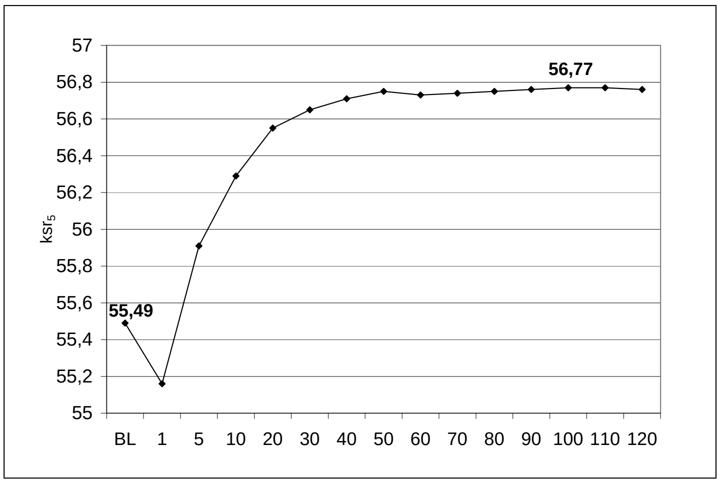


Figure 5.7: Results (ksr_5) for different context sizes ($|h| = 1 - 120$); tested on *news-en*

The curve appears to be quite smooth: It quickly rises up to a size of 50 words and then remains rather flat, reaching its maximum at a context size of 100 words. Interestingly however we measured a degradation in performance for a size of 1 content word only. This can probably be explained by an overestimation of the combined model. Local semantic dependencies are already well estimated by our base model. The information provided by an LSA model which also captures local dependencies only then has a deteriorating effect.

To conclude this section: The LSA models applied henceforth are reduced to 150 dimensions, and they are calculated on a co-occurrence matrix using a window of ± 100 words. A maximum context of approximately 100 words seems to yield stable results, therefore we set the context size in the following to this value.

5.5.3 Integration methods

In section 5.4 we discussed the difficulties that arise when LSA is used for prediction, and we presented a number of different methods how information coming from an LSA model can be integrated with a classical n-gram approach. Here the performance of all of these methods is evaluated. Again, a considerable number of parameters had to be optimized; this (rather tedious) work was performed on held-out test data and will not be reported in detail here, however all parameters specified below were optimal with respect to our test corpora. The following 6 methods were applied with the parameters as specified:

1. *LSA-trigger*: LSA-based trigger model, as explained in section 5.4.1. Parameters: Cache size = 1000; threshold θ = 0,4; scaling factor β = 0,0001.
2. *Rerank*: Partial reranking, as presented in section 5.4.2. Parameters: Number of reranked items m: 1000; scaling factor β = 0,001.
3. *SLI*: Static linear interpolation; Contrasting factor γ = 4; Coefficient λ_{LSA} = 0,096 (Set by EM algorithm)
4. *SGI*: Static geometric interpolation; Contrasting factor γ = 4; Coefficient λ_{LSA} = 0,073 (Set by EM algorithm)
5. *CWLI*: Confidence-weighted linear interpolation; Contrasting factor γ = 4; Coefficients λ_{LSA} dynamically set according to the density-based confidence measure (D_{100}, cf. section 5.4.4). $\beta = 0,4$.
6. *CWGI*: Confidence-weighted geometric interpolation; Contrasting factor γ = 4; Coefficients λ_{LSA} dynamically set according to the density-based confidence measure (D_{100}, cf. section 5.4.4). $\beta = 0,4$.

These methods were evaluated on the three newspaper corpora (*news*), ksr_5 was determined on all words (i.e. closed-class words included). The LSA spaces applied (one per language) were trained on the corpora mentioned above (cf. 5.4), based on a 80.000$\times$3.000 term-term matrix and a co-occurrence window of $\pm$100 words. The matrices have been reduced by SVD to 150 dimensions each.

Table 5.5 gives an overview on the results for all methods tested as well as for the 4-gram baseline and the cache model, as presented in section 4.5 (Tables 4.4 and 4.9).

Considering the results in Table 5.5 three observations can be made: First, the results for the trigger approach, partial reranking and for static linear interpolation are very close, their ksr advantages range from +0,61 to +0,85, which is only slightly superior to the standard cache model (+0,42 – +0,62). Taking into account their rather different functioning, this is an unexpected result.

Method	*news-fr*	*news-de*	*news-en*
Baseline	57,78	51,56	55,49
Cache	58,20% (+*0,42*)	52,18% (+*0,62*)	56,03% (+*0,54*)
LSA-trigger	58,42% (+*0,61*)	52,21% (+*0,65*)	56,25% (+*0,76*)
Rerank	58,53% (+*0,72*)	52,26% (+*0,70*)	56,31% (+*0,82*)
SLI	58,49% (+*0,68*)	52,22% (+*0,66*)	56,34% (+*0,85*)
CWLI	58,30% (+*0,49*)	52,16% (+*0,60*)	56,21% (+*0,72*)
SGI	58,61% (+*0,80*)	52,32% (+*0,76*)	56,59% (+*1,10*)
CWGI	58,92% (+*1,14*)	52,59% (+*1,03*)	56,77% (+*1,28*)

Table 5.5: Results (ksr_5) for all integration methods tested

Second, geometric interpolation, particularly when it uses confidence weighting, performs significantly better than the other methods, here advantages of more than +1% can be measured.

Third, as a surprising matter of fact, the worst performing method is confidence-weighted linear interpolation, performing hardly better than the standard cache. This result is in clear contrast to static LI as well as to CWGI. It can probably be explained by the the already discussed property of linear interpolation, which does not take into account whether two models agree or not. The confidence-weighting scheme gives high LSA weight to rather specific terms, disregarding their appropriateness in the given context. When these terms then show a high semantic similarity with the context, they are overestimated by the model, local morpho-syntactic constraints are ignored[20]. CWGI is able to deal with this unwanted effect, therefore we find the best results here.

While it has been shown that confidence weighting in combination with geometric interpolation is the best performing integration method, the confidence metric itself has not yet been evaluated. Coccaro & Jurafsky (1998) have presented an entropy-based metric, but, drawing on previous results, we have argued that a confidence metric based on cluster density might be more appropriate. To find out whether or not this is the case we have compared the two confidence metrics to one another. Table 5.6 shows the results (ksr_5) for the entropy- as well as the density-based metric, using geometric interpolation; all parameters and methods remained unchanged.

It seems that the density metric has slight advantages over the entropy-based measure. We do not want to argue that the differences are significant, however the density measure scores better for all three corpora. It would now be interesting to try out other measures of specificity as well as combinations of these; moreover, one could also imagine to incorporate part-of-speech information here.

[20]Remember the example of *the student sleeps* vs. *the student university*

Confidence metric	*news-fr*	*news-de*	*news-en*
Entropy	58,83% (+*1,05*)	52,42% (+*0,86*)	56,60% (+*1,11*)
Density (D_{100})	58,92% (+*1,14*)	52,59% (+*1,03*)	56,77% (+*1,28*)

Table 5.6: Results (ksr_5) for the entropy- and the density-based confidence measures; test performed using geometric interpolation

5.5.4 Cross-register evaluation

The previous two sections have shown that the optimization of certain parameters as well as the way of integration is crucial for the success of a model taking semantic information into account. However the optimization of our model was performed on intra-register data only. In order to get an impression of the real-life performance, it has to be tested on corpora from other registers as well.

In Table 5.7 we show the keystroke savings (ksr_5) for all test corpora from 4 different registers and 3 languages (test on all words). As above, the optimal LSA models that we have finally applied have been trained on the corpora specified in 5.4, use a 80.000×3.000 term-term matrix, based on a co-occurrence window of ±100 words, and are reduced to 150 dimensions. The integration of the LSA information was performed by CWGI.

	French	German	English
Newspaper	58,9% (+*1,1*)	52,6% (+*1,0*)	56,8% (+*1,3*)
Literature	47,7% (+*1,7*)	46,1% (+*1,2*)	51,0% (+*1,2*)
Speech	49,9% (+*1,6*)	50,4% (+*1,3*)	50,1% (+*1,6*)
E-mail	50,2% (+*1,6*)	49,1% (+*1,1*)	50,7% (+*1,3*)

Table 5.7: Results (ksr_5) of the LSA-based model (CWGI) and advantages over the baseline for all test corpora

As the numbers in Table 5.7 show, the gains of the LSA-enhanced model are quite stable. Over all languages and registers we find advantages (with respect to the 4-gram baseline) of +1,0% to +1,7%; in a paired t-test all gains turned out to be highly significant (sig. level ¡ 0,001 at a 99% conf. interval). Important differences between the three languages however cannot be determined, and in the inter-register comparison the speech corpora seem to obtain slightly better results than the other registers (for the German and English corpora). This is probably due to the type of dialogues (tourist information/business appointments), making the text semantically quite distinct. But also another factor plays a role here: As shown before, the LSA model gives higher weight to *self-triggers* (i.e. words that have occurred before), it therefore has a similar effect as the cache model. And, as could be seen in the previous chapter (cf. section 4.5, Table 4.5.2), the speech corpora were particularly sensitive to cache information, since they use a highly constrained and repetitive vocabulary.

In general it has to be stated that the gains achieved by our semantic model do not seem overwhelming, especially when compared to the user-oriented models in the previous chapter. Still, it can be noticed that these results range – to our knowledge – among the highest gains for an approach exploiting semantic information for word prediction (cf. section 5.5.5).

In addition, the 4-gram baseline to which we compare our model already performs rather well, and – as discussed in chapter 2 – at a baseline of more than 55% it seems increasingly difficult to achieve additional keystroke savings. Moreover, it has not yet been investigated to what extent semantic information can theoretically contribute to prediction. In this light it is impossible to estimate how much of this information has been exploited by our approach.

Another aspect that could not be evaluated by the quantitative measures applied here is the qualitative effect of such a model: An AAC system cannot only help speeding up text insertion, it can also provide cognitive support for its user, whose communicative abilities might be totally depending on it. Therefore, she or he might feel a strong improvement using a word predictor that is able to propose semantically related terms, even though the actual gain in terms of keystroke savings might be modest. This effect is however quite hard to investigate; only long-term work with several users can reveal to what extent they feel supported by such a predictor.

5.5.5 Comparison with related work

In section 5.2 a number of approaches were presented that aimed at exploiting semantic information for word prediction. This section aims now to give an overview on the results achieved with these approaches, as reported by the respective works. As mentioned before, it is futile to directly compare these results, since they were obtained under highly varying conditions.

In particular we presented structural approaches, trigger and topic-oriented models. For the structural models however detailed quantitative evaluations were not published (cf. Guenthner *et al.*, 1992; McCoy & Demasco, 1995), or an enhancement in terms of keystroke savings could not be found (Hunnicutt, 1989).

Concerning the trigger approach we find an evaluation in (Matiasek & Baroni, 2003). Here, a base model consisting of a word bigram and a PoS trigram model was combined with long-distance co-occurrences (± 50 words), calculated using *pointwise mutal information* (*PMI*, cf. section 5.2.2). The evaluation was performed on a journalistic test corpus; the base model achieved a ksr_5 of 41,475%, the combined model arrived at 41,653%, which represents a ksr advantage of 0,178. Although the authors showed that this gain is significant (applying a two-sided Wilcoxon signed rank test), it is surprisingly low, compared even to our results with the cache model (cf. section 4.3.1).

Word triggers, based on PMI, are also used by Li & Hirst (2005). They apply however a second processing step: the trigger pairs are filtered according to their semantic relatedness, calculated from *WordNet* glosses. The results reported by Li & Hirst (2005) seem at first very promising (+6%), however they define ksr in a very different manner (distinguishing several word classes), they do not mention the size of their prediction list and they test on one small corpus only (3700 nouns); it is therefore difficult to interpret their results.[21]

A topic model was presented and evaluated by Trnka *et al.* (2006). Here, keystroke savings were calculated for integration methods A and B (cf. section 5.2.3); the test data were taken from the *Switchboard* corpus (221 conversations). Using a prediction list of 5 words, both methods achieved the same advantage over a trigram baseline of +0,2%. Interestingly however, method A achieved larger gains with a growing list size, whereas the advantages of method B were constant.

Table 5.8 summarizes the results achieved by the works mentioned above.

Reference	Prediction method	Language	ksr_5
(Matiasek & Baroni, 2003)	Bigram + PoS-Trigram	German	41,475%
	Bigram + PoS-Trigram + triggers (PMI)		41,653%
(Li & Hirst, 2005)	Bigram + PoS-Trigram	English	59%
	Bigram + PoS-Trigram + filtered triggers (PMI)		65%[22]
(Trnka *et al.*, 2006)	Trigram	English	57,7%
	Trigram + topic model		57,9%

Table 5.8: Keystroke savings (ksr_5) reported by different works using semantic adaptation

5.6 Conclusion

The principal objective of this chapter was to discuss in what way semantic information can be exploited in order to improve the prediction of words. We presented a range of approaches, from structural paradigms, over methods using trigger and topical information to Latent Semantic Analysis, a vectorial approach based on distributional properties of words, which already has

[21]We tried to contact the authors. Graeme Hirst replied very kindly but could not provide comparable ksr results, since he lost contact with Jianhua Li, a former Master student of his.

[22]As stated above, Li & Hirst (2005) define ksr differently (based on distinct word classes); moreover the size of the prediction list is not mentioned.

shown convincing performance in determining semantic relatedness. As this method allows for the geometric representation of any kind of linguistic unit, it makes similarity measurements between words and contexts possible. It is however not straightforward how the semantic information provided by LSA can optimally be integrated with local syntactic information coming from a classical language model. We therefore presented several integration methods: an LSA-based trigger approach, partial reranking, and different forms of interpolation. We also proposed a new kind of confidence metric for estimating LSA performance, based on cluster density; it is very easy to compute and correlates better with the quality of LSA relations around a term vector than an entropy-based metric.

Due to their inherent complexity and the data required, not all presented approaches could be implemented and evaluated here. We still performed an extended quantitative evaluation on the LSA model itself (including a number of important parameters) and on the different integration methods presented.

The best performing LSA model (in terms of keystroke savings) used a co-occurrence window of ± 100 words and its input matrix was reduced to 300 dimensions; this model however had an already critical runtime. We therefore opted for using a 150-dimensional model, performing 30% faster despite a slight performance loss. As for the integration method we found a clear advantage for geometric interpolation, dynamically weighted by the proposed density-based confidence metric.

Considering the results in a cross-register evaluation, we observed over all registers rather stable ksr advantages for the LSA based model of +1,0% to +1,7%; clear language-specific differences could not be determined. These gains do not seem very high, compared to the success of the user-oriented approaches in the previous chapter; yet it has to be acknowledged that they already represent a significant improvement with respect to other approaches (e.g. trigger or topic models). Moreover, while we have investigated several influential parameters in detail, we do not want to claim that we have applied the optimal model here; the field of distributional semantics is still rather young, and more work has to be done in order to better understand the capacities of corpus-based co-occurrence models such as LSA as well as the relation between the distributional properties of words and their semantics.

Chapter 6

SIBYLLE - an adaptive AAC system

Beginning with the generation first
Of mortal men down to the very last
I'll prophesy each thing: what erst has been,
And what is now, and what shall yet befall
The world through the impiety of men.

THE SIBYLLINE ORACLES[1]
(Transl. by Milton Terry, 1899)

In the following we describe the latest version of SIBYLLE, the AAC system that incorporates the adaptive word predictor, presented in the previous chapters. We first introduce the user interface, the dynamic key selection (*SibyLetter*) and the word predictor (*SibyWord*), as it has been integrated to the system (6.1). In section 6.2, some implementation issues and design decisions are disclosed, and in part 6.3 we present and discuss results of a quantitative evaluation of the AAC components, in particular the key selection procedures and the word predictor. In the last section (6.4) we report findings from the application of SIBYLLE in a rehabilitation center, where the system has been in use since 2001.

6.1 Components

Among the AAC systems introduced in chapter 2 we have already presented a former version of the SIBYLLE system (cf. 2.4). We have also mentioned that

[1]As the translator notes, these texts should more properly titled 'the Pseudo-Sibylline Oracles'. The original Sibylline Books were oracular scrolls written by prophetic priestesses (the Sibylls) in the Etruscan and early Roman Era as far back as the 6th century B.C., but these books were destroyed. The texts of which the first lines are presented here have been composed between the 2nd and the 6th century A.D.

it has been developed from 2001 onwards in the context of the PhD project of Igor Schadle at the *Université Bretagne-Sud* (cf. Schadle, 2003). At that time it already comprised the main components: a user interface, a letter predictor for dynamic keyboard rearrangement (*SibyLettre*) and a word predictor (*SibyMot*).

It is difficult to enumerate every detail that has been modified since, it can however be stated that the major changes concern the word predictor, the disabbreviation facility, the interfacing functionality (with external applications) as well as the languages included: While the former system was monolingual (French), we have made the architecture language-independent and added two more languages: German and English. In the following we will describe the components mentioned above in their present state of development.

6.1.1 User interface

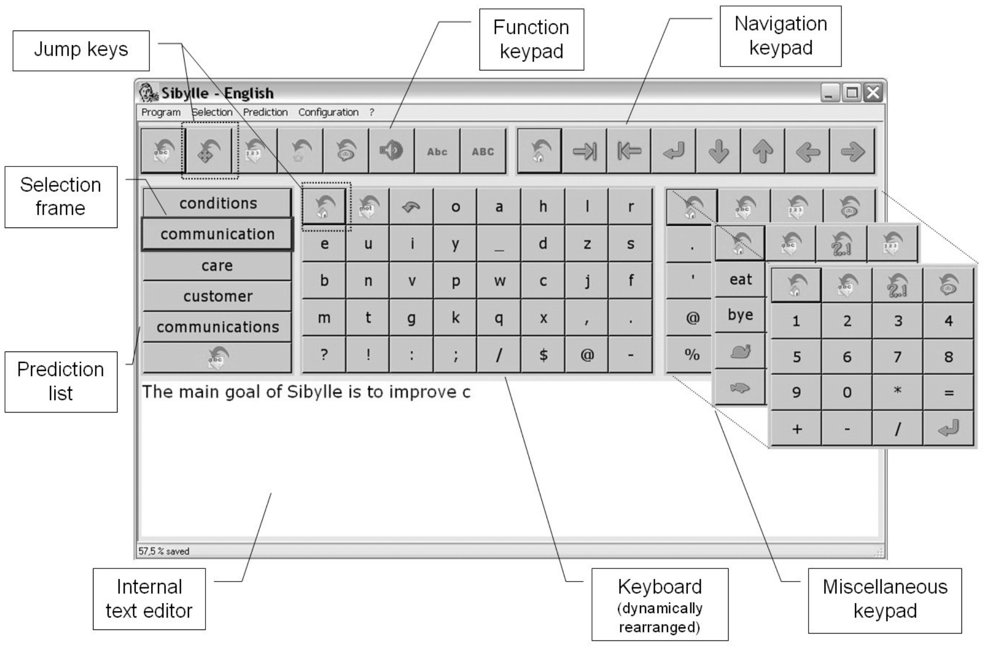
To get an idea of the overall functionality of SIBYLLE, let us first regard its user interface. Figure 6.1 shows its latest version (v. 3.7)[2]. The virtual keyboard combines a set of sub-keypads offering to insert letters, numbers, words and also predefined sentences for *emergency* uses. Jump keys provide fast moves between these sub-keypads: they are usually the first keys on each keypad. The different keypads of the interface are displayed in Figure 6.1 and they will be detailed in the following:

1. *Letter keypad*: it comprises all alphabetic keys and is used to compose messages, character after character. When the user activates the letter prediction component of SIBYLLE (s. section 6.1.2) with linear scanning, the keys are dynamically rearranged in order to present the most probable letters first. Since punctuation signs and numbers are hardly predictable, they are displayed in a separate keypad. The current 5×8 layout of the keypad however offers more keys than there are characters; these keys have been filled with the most frequently used punctuation signs (the layout can be modified).

2. *Prediction list*: when the user selects one of these predicted words, it is automatically inserted into the message. The contents of this list is updated after every change of the given context (left to the current cursor position). The user can choose between a horizontal and a vertical layout. Results from previous works however suggest that a vertical arrangement of the word list is better accepted than a horizontal one (cf. Garay-Vitoria & Abascal, 2006).

3. *Function keypad*: this keypad comprises a number of function keys (like printing, activating the speech synthesis etc.), jump keys (to reach other

[2]Some features of the interface were developed by Nicolas Béchet (Université de Tours) and Zaara Barhoumi (Université Bretagne-Sud) during their research labs. Cf. (Béchet, 2006) and (Wandmacher *et al.*, 2007).



The main goal of Sibylle is to improve c

Figure 6.1: User interface of SIBYLLE, v. 3.7

keypads) and keys to handle external windows and applications. In former versions, SIBYLLE only comprised an integrated text editor connected with a text-to-speech synthesizing application. But since users also want to compose e-mails, use a real word processor or a search engine on the web, we decided to make SIBYLLE more flexible. By interfacing the *Microsoft Windows* API, our system is now able to enter text in any kind of text-processing *Windows* application. Furthermore, configurable function keys enable direct actions such as *Save*, *New file* and *Open file*.

4. *Navigation keypad*: this keypad enables the user to move the text cursor without operating a mouse. It can be used during message composition (for displacing the caret within the text), but it also enables to select menu items of external applications.

5. *Miscellaneous keypad*: this keypad can be used in several modes. One can use it to select numbers, but also punctuation marks, and finally to select predefined sentences or messages. This property is very important for the users, e.g. in emergency situations. Pre-recorded messages can be individually defined, and they are represented by an icon or by the beginning of the respective phrase.

When the user is not able to control a mouse, key selection is performed by scanning: a selection frame successively highlights each key, which can then be selected. Experiments with our system have shown that users are often disturbed by the abrupt shifts of the selection frame. When the cursor approaches the desired key, they have difficulties to temporally prepare their action. As a result, we observed a significant rate of selection errors. For this reason we have added a timing line, which glides from the top to the bottom of the frame (Figure 6.2) and shows the time remaining until the next shift by its position. This temporal feedback has proven useful to many users.

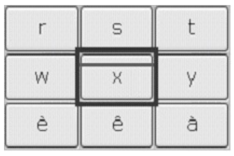
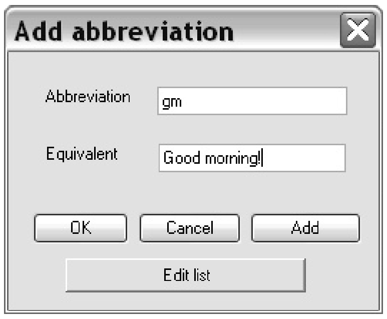


Figure 6.2: Timing line giving temporal feedback on scanning delays (SIBYLLE, v. 3.7)

For users who are still able to control the keystroke duration, we have implemented a click timer to which specific functions (such as erasing, capitalizing or jumping to other windows) can be assigned. This timer distinguishes up to three durations, and when applied, it can save a lot of additional scanning steps.

SIBYLLE also includes a module for abbreviation expansion. This component enables the user to define his/her own abbreviations that are directly expanded during the composition of a message (Figure 6.3). If the user prediction facility is activated, abbreviations will automatically be added to the vocabulary (as soon as they have been used for the first time), so they can be predicted as well.


Add abbreviation

Abbreviation: gm

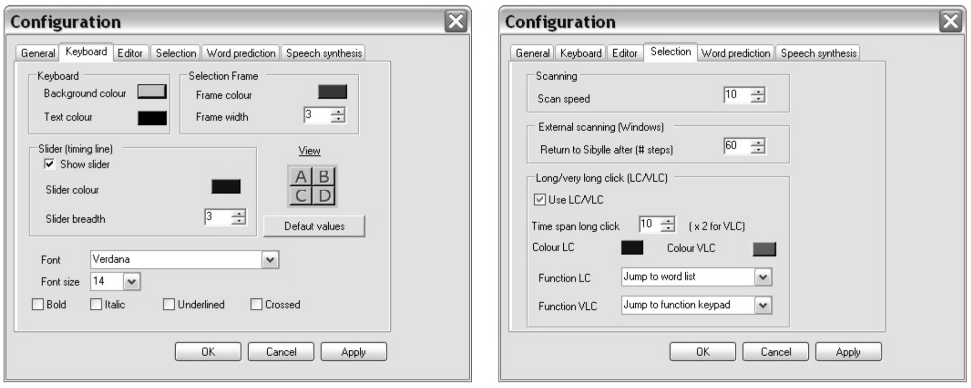
Equivalent: Good morning!

Figure 6.3: Mask for entering abbreviation/expansion pairs (SIBYLLE, v. 3.7)

One of the most distinguishing features of SIBYLLE is the high adaptability of its user interface: In order to best meet the individual requirements of every user, we designed all interactive elements as modifiable as possible. This extends to the whole layout of the interface, for example:

- *Keyboard rendering*: Colors, fonts and font size of all keyboards as well as the keys themselves can be modified and rearranged.
- *Selection parameters*: Scanning mode, scanning delays, time spans and long-click functionality (such as direct jumps to other windows or capitalizing) are adjustable.
- *Interface layout*: The size and position of every sub-keypad within the application window can be set individually.
- *Key assignment*: Each key can be assigned to different actions, sound events or background images.

Figure 6.4 shows some of the configuration panels of SIBYLLE.


Configuration
General | Keyboard | Editor | Selection | Word prediction | Speech synthesis

Keyboard
Background colour
Text colour

Selection Frame
Frame colour
Frame width: 3

Slider (timing line)
☑ Show slider
Slider colour
Slider breadth: 3

View
Defaut values

Font: Verdana
Font size: 14
☐ Bold ☐ Italic ☐ Underlined ☐ Crossed

OK | Cancel | Apply

Configuration
General | Keyboard | Editor | Selection | Word prediction | Speech synthesis

Scanning
Scan speed: 10

External scanning (Windows)
Return to Sibylle after (# steps): 60

Long/very long click (LC/VLC)
☑ Use LC/VLC
Time span long click: 10 (x 2 for VLC)
Colour LC
Colour VLC
Function LC: Jump to word list
Function VLC: Jump to function keypad

OK | Cancel | Apply

Figure 6.4: Configuration panels of SIBYLLE, v. 3.7

Adapting colors and fonts may seem a trivial property at first sight. However in the context of AAC this obtains a different importance, since some users have varied and complex forms of visual impairment. Here, a very particular configuration may become necessary. For the same reason the size and the placement of the keypads can be adapted to the user's needs. Moreover, the configuration of each keypad is stored in a text file with a simple, intuitive format (Figure 6.5 shows an example for the definition of the key '*a*'). By editing these files the rendering, the position as well as the action performed by each key can be modified.

An optimal configuration of the interface however can of course only be achieved by close interaction between the user and the medical or caring staff (cf. also section 6.4).

6.1.2 *SibyLetter*

As discussed in the chapter on AAC systems (ch. 2), key selection is often achieved via a line/column scan which significantly reduces the average num-

```
(1,5)
Texte=a
Image=letter_a.png
Commande=31,a
Special=1
```

Figure 6.5: Definition of the key *'a'* in the configuration file of the letter keypad

ber of scanning steps needed to reach the intended key. However, this selection mode requires at least two keystrokes per item selection, and we learned from user feedback that this kind of selection is rapidly tiring. For this reason static linear scanning is often preferred, despite its significantly higher average scanning distance (ASD, cf. section 2.5.1).

In order to speed up communication using the preferred linear scan, SIBYLLE comprises the possibility to dynamically rearrange the letter keypad according to the given context. The dynamic reordering is directed by *SibyLetter*, a letter prediction module based on a 5-gram letter model. This model estimates at every change of the context the conditional probability of each character given the four previously typed symbols (cf. Schadle *et al.*, 2001):

$$P(c_i) = P(c_i|c_{i-4}, \ldots, c_{i-1}) \tag{6.1}$$

Spaces between words and all punctuation signs are also taken into account for prediction. Three models were trained on large corpora for French, German and English, respectively[3]. Data sparseness is managed in *SibyLetter* with a simple back-off technique (cf. section 3.3.2): if a specific letter n-gram is not observed in the training corpus, its probability is estimated from the (n-1)-gram. As an illustration, Figure 6.6 shows the dynamic reorganization of the letter keypad, when the user composes the first letters of the word *'three'* on the English version of SIBYLLE (at each step only the first line of the keyboard is displayed).

To reduce the cognitive load, this dynamic behavior should only be used in combination with linear scanning. In this mode, the user's attention is focused to the selection frame and its immediate environment, and she or he is not particularly disturbed by the keypad's reorganization. In any case, this selection mode seems to be well accepted, compared to a static line/column mode, as long as the user is not visually impaired (cf. also section 6.4).

6.1.3 *SibyWord*

In the previous three chapters, many word prediction techniques were presented and evaluated. The word predictor applied in SIBYLLE is of course

[3]We used newspaper corpora, as presented in section 4.4

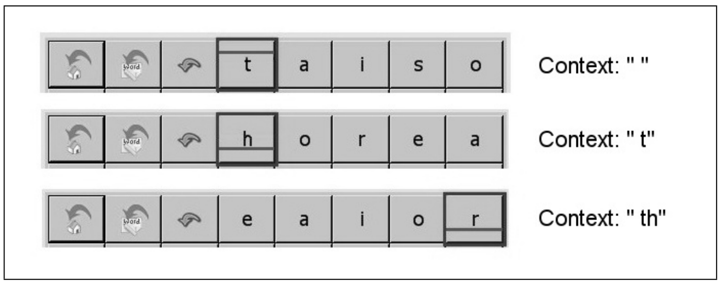


Figure 6.6: Reorganization of the dynamic letter keypad during the composition of the first two letters of *'three'*.

based on the findings presented in these chapters. The prediction engine of *SibyWord* integrates: (i) a base predictor, using word probabilities from an n-gram-based language model, (ii) a user predictor, based on the *dynamic user model*, as presented in chapter 4 (4.3.3), and (iii) a semantic predictor, making use of *Latent Semantic Analysis* in order to adapt the prediction output to the current semantic context (cf. chapter 5).

High prediction performance is however not the only requirement for a word predictor in a real-life system. Here again, configurability plays an important role. For this reason, several parameters can be set for the use of *SibyWord*:

1. *Shown words filtering*: The strategy of filtering words that have been displayed one time has already been mentioned in chapter 4. It is based on the assumption that a word appearing in the list and not being selected right away is not intended; even though it may still match the given onset after insertion of another character, it can be filtered out for the current prediction to leave place for other words. This strategy obviously enhances keystroke savings (s. section 6.3.2), but it includes the risk of missing the intended word in the list (which implies that the rest of its characters have to be inserted). The degree of helpfulness of this strategy depends on the user's cognitive and visual abilities. For this reason the filtering of already shown words can be switched off.

2. *User predictor*: The behavior of the user predictor (UP) can be controlled in several ways: It can be completely disactivated, it can be reset, the automatically acquired user vocabulary (unknown words) can be modified and deleted. Moreover the training process of the UP can be individually controlled, so that after every session it can be decided whether the inserted text is to be added to the prediction model or not. Offering such possibilities of control is very helpful for example in a teaching context: a typical exercise in speech therapy lessons is to repeatedly insert a phrase until the learner is able to type it correctly. Yet such phrases are rather artificial and probably do not help in adapting to the user; they should

therefore not be added to the predictor.

3. *Semantic predictor*: The calculations performed by the semantic prediction module are rather complex and require a considerable amount of data to be kept in working memory (approx. 180 MB). While this is unproblematic for modern machines, it can lead to delays in the prediction process or to execution problems on older computers. In this case the semantic predictor can be switched off.

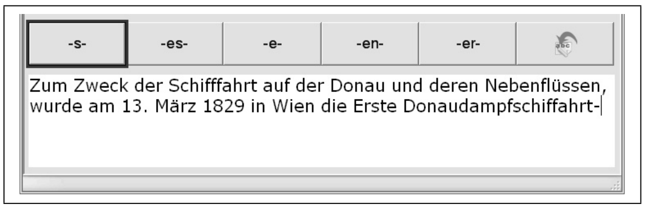
4. *Partial selection*: For languages such as German, it is important to have a strategy for inserting compound words (see section 3.5.4 and below for further details). Partial selection (PS) makes this possible, but it alters the function of the backspace key, which triggers the PS mode. Since compounding is very rare in other languages, and also occurs rarely in oral communication, PS can be disabled.

5. *Vocabulary limitation:* Limiting the vocabulary seems at first counter-intuitive, since it can be imagined that a larger vocabulary is always beneficial for prediction. While this shows to be true from a theoretical point of view (cf. also section 6.3.2, Table 6.9), younger children (language learners) become quickly distracted by a large vocabulary, including many words they have not yet learned. In such cases it is reasonable to use only a subset of the most frequent words in for prediction. Therefore, if the vocabulary limit is set, *SibyWord* calculates a frequency ordering of the complete vocabulary and adds all terms below the limit to the word filter.

Compounding revisited: Partial selection

In chapter 3 we have already discussed the phenomenon of productive compounding in German and the problems it presents for NLP applications (cf. section 3.5.4). In chapter 4 we have analyzed the words in some of our test corpora which could not be predicted (*out-of-vocabulary* words), and we have observed that more than half of the unknown words in the German newspaper corpus (51%) were compounds, a finding which underlines the demand for a solution to this problem (cf. section 4.4).

The *split-compound* model by Baroni *et al.* (2002) – as presented in section 3.5.4 – does not offer a general solution to compounding, it only addresses a frequent type of compounds (noun+noun). Furthermore, its evaluation showed only marginal gains in keystroke savings of up to +0,3% (cf. Trost *et al.*, 2005). We have therefore opted for a different strategy (also applied by Trost *et al.* (2005)) which is straightforward: *partial selection* (PS) allows for selecting each part of a compound one after the other and agglutinating it to the former part by entering a backspace after selection. This alone however would not be sufficient, because in some cases a *compounding suffix* has to be attached to a modifying term (e.g. '*Verein***s***sitzung*' club-**s**-reunion). Therefore, our method allows a person to enter one of these morphemes (-s-, -e-, -en-, -es-, -er-)[4] after a compound part has been selected, and all nouns (starting with a capital letter in German) become lowercased. Figure 6.7 gives an example for the partial selection of a compound.



Zum Zweck der Schifffahrt auf der Donau und deren Nebenflüssen, wurde am 13. März 1829 in Wien die Erste Donaudampfschiffahrt-

Figure 6.7: Partial selection for typing a compound (here: *Donaudampfschiff-fahrtsgesellschaft*, 'Danube steamboat corporation')

As will be shown in section 6.3.2, partial selection leads to a significant gain in keystroke savings for German, and it is straightforward to apply (by typing a backspace). However it can also be distracting for persons who often mistype words (e.g. due to visual impairment), since the backspace has to be typed twice then. In such cases it is better to disable this functionality.

6.2 Implementation issues

This section is not supposed to provide an exhaustive description of the system's implementation, it is rather meant to give a concise overview of the fundamental design decisions and the different modules of the system.

The major design principle for the implementation was to keep the architecture as modular as possible. It is an old truth in software engineering that the encapsulation of different components is crucial for the maintenance as well as for the extension of a system; for our project however modularity also enables to integrate different parts of the system (such as the letter or word predictor) to other projects[5].

Another important aspect of modularity concerns language; all language-specific parameters, menu labels and data files were kept separate from the executing modules in order to keep the architecture generic. This makes an extension of the system to a new language very easy. Moreover, since all prediction methods applied are data-driven, and computing the data files requires nothing but sufficient amounts of text, adapting SIBYLLE to another language

[4]In an extensive lexicographic analysis, Langer (1998) could determine 68 different compounding suffixes. We integrated here only the five most common elements.

[5]In the framework of the French *VOLTAIRE* project, financed by the *Association Française contre les myopathies* (AFM), it is planned to integrate *SibyWord* to the *Custom Virtual Keyboard* (CVK); cf. also http://www.cvk.fr/.

is now only a matter of days. In this way, the effort for developing a well-performing system in even less common languages (like Dutch or Breton) is reduced to a minimum; the problem of AAC for small user bases, as mentioned by Newell *et al.* (1998), can in principle be met.[6]

SIBYLLE has been implemented in *Visual Basic .NET*, a fully object-oriented language, based on the *Microsoft .NET* framework. This platform offers a large range of APIs and a runtime environment (*Common Language Runtime*, CLR), a virtual machine enabling to execute intermediate code written in different languages (such as *C#*) on different system architectures. The idea of a virtual machine has already been introduced with *JAVA*, however it could be shown that the *Microsoft* CLR performs significantly faster in a number of benchmark tests, its runtime performance is comparable to fully compiled code (cf. Rottmann, 2004). Another important advantage of the *.NET* environment is the instant availability of interfacing functions with the *Windows* architecture (e.g. for accessing other applications, printing or control of pointing devices) which is crucial for a system like SIBYLLE.

In the following an overview of the different modules and their interaction shall be given. Due to the large number of classes and modules (exceeding 60), showing a class diagram for the whole system would not be reasonable. We therefore show in Figure 6.8 only the major components, in order to give an understanding of the overall architecture.

The executing class of the system is `Sibylle Application`. It loads the different windows and handles all events generated by them. Window management is performed by MDI (*Multiple Document Interface*), enabling to work with several windows or documents at the same time. The principal window (`MainWindow`) contains the editor window as well as the keyboard, which itself consists of several keypad windows (comprising the letter-, word- and function keys etc.). The configuration of each keypad is stored in a separate text file, which can easily be edited (cf. section 6.1.1, esp. Figure 6.5).

The `SettingsLib` class manages the configuration facilities and comprises a bundle of controls (`SettingsControl`) for parameter setting. All selected parameters are stored in a text file (*'Sibylle.ini'*), and the language labels are stored in another (*'SibyLang.ini'*). Both files are loaded on startup in order to set the system parameters and the language according to the user's preferences; they are read from the user's personal application data folder, so that several users can work with SIBYLLE on the same machine.

The `KeySelection` class takes care of all selection facilities and events; it controls the different scanning (linear or line/column) and pointing modes (performed by `LinearScan`, `LCScan` and `Pointing`, respectively). `DisAbbreviation` provides the facilities for abbreviation expansion, man-

[6]This claim is subject to two conditions: (i) the language to be integrated has to be alphabetic, and (ii) must not be *agglutinative*. In languages such as Turkish, Finnish or Basque words are formed by joining morphemes together. Here, prediction has to be based on the morpheme rather than on the word level.

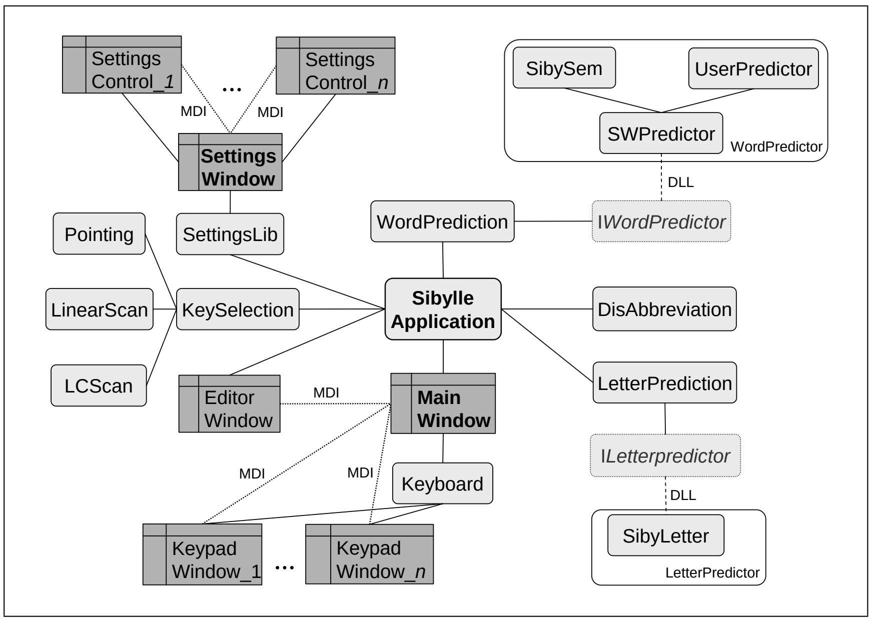


Figure 6.8: Overview of the major components of SIBYLLE v. 3.7

aging a hashtable of abbreviation/expansion pairs which is looked up every time a word has been terminated in the editor.

The classes `LetterPrediction` and `WordPrediction` mainly serve as template classes, they however also comprise important preprocessing functions (e.g. for *partial selection*). The actual prediction is however performed by separate components, *SibyLetter* and *SibyWord*. These components are interfaced as DLLs (*Dynamic Link Libraries*), they implement a small number of public interfacing functions (e.g. *loadData*(), *Predict*()), which are specified by the interface classes `ILetterPredictor` and `IWordPredictor`. The interfacing via DLL makes the integration of the module to external projects particularly easy, since, after declaration of the respective DLL, the interfacing functions can directly be applied within the foreign code.

SibyWord itself then comprises three major components: `SWpredictor` provides the actual word prediction functions, including context parsing, probability computation and candidate ranking, and it manages the user prediction facilities (`UserPredictor`) as well as the semantic predictor (`SibySem`). `UserPredictor` implements the *Dynamic User Model*, as presented in chapter 4 (4.3.3), and `SibySem` the LSA-based prediction method, as described in the previous chapter (5.4). The exact parameters of the models applied can be found in the respective evaluation sections (cf. 4.5.4 and 5.5).

6.3 Performance

The evaluation sections of the previous chapters concentrated on the particular effects of distinct model parameters. In the following we want to consider the performance of the system as it is applied. We first present results from a quantitative evaluation of the static and the dynamic key selection methods incorporated and of the word predictor (*SibyWord*), including the adaptation strategies presented previously. Furthermore we analyze the effects of some user-controlled parameters, such as vocabulary limitation or partial selection.

6.3.1 Key selection

As explained in the second chapter (2.5.1), the performance of a key selection method can be quantitatively evaluated by measuring the *Average Scanning Distance* (ASD), which can be determined for static and for dynamic key arrangements as well as for different kinds of scanning techniques.

To assess the performance of *SibyLetter*, we have again made use of the test corpora from four different registers, as detailed in section 4.4. The probabilities for the static as well as for the dynamic arrangements (using the letter predictor) have been calculated on the same corpora, described in section 4.4. The keys for the static arrangements are ordered by probability[7] in a 5×8 keyboard layout (as can be seen in Figure 6.1). The scanning distance of upper case letters, which are accessed in SIBYLLE via a shift-key in the function keypad was calculated as the distance of the respective lower case letter plus one additional scan step. For determining the ASD only alphabetic characters and the empty space were considered (summing up to 65 in French, 60 in German and 53 in English).

It is obvious that register influences do not play such an important role on the character level, however it is still important to consider the performance for different corpora in order to get an idea of the flexibility of the approach. Table 6.1 shows the ASD results for the three key selection methods incorporated in SIBYLLE on all test corpora considered.

As Table 6.1 shows, the results of the static scanning methods are quite stable with respect to the register as well as to the language. It is interesting to see that the larger number of characters in French, compared to English does not seem to influence the scanning distance (for the static linear method the results are even identical). However the ASD of the linear method is considerably higher than the one of the line-column scan. This was expected, but it has to be remembered that this method implies more cognitive effort from the user.

In dynamic mode the wanted character appears on the average at the 3rd to

[7]While the character probabilities were determined for all three languages on large newspaper corpora (cf. section 4.4), the ordering of the French keyboard has been slightly modified, according to the preferences of the users at the Kerpape rehabilitation center (cf. section 6.4).

Corpus	# chrs.	static linear	static line/col.	dynamic *SibyLetter*
news-fr	65	7,1	5,0	3,1
lit-fr		6,9	4,9	3,5
speech-fr		6,8	5,1	3,9
email-fr		7,2	5,1	3,4
news-de	60	7,2	5,1	3,0
lit-de		7,2	5,1	3,1
speech-de		7,2	5,0	3,2
email-de		7,1	5,0	3,2
news-en	53	7,1	5,1	2,9
lit-en		6,9	4,9	3,1
speech-en		6,8	4,8	3,3
email-en		7,2	5,0	3,2

Table 6.1: Performance (ASD) of the the static (linear and line-column) and the dynamic (linear) key arrangements (*SibyLetter*) on all test corpora

4th position of the keyboard. This is in accordance with the results of Schadle *et al.* (2001). Here the ASD for French is slightly higher than for German or English, which can be probably explained by the higher number of characters.

The differences between the registers are mostly small, however the intra-register test corpora still show the shortest ASD in all three languages. The French speech corpus however shows a significantly worse result (ASD = 3,9). It is hard to find definite explanations here, but it could be due to the high number of interjections and abrupt shifts in the conversation, making prediction more difficult.

In general it can be stated that the performance of *SibyLetter* means an important quantitative improvement, compared to static selection techniques. Apart from the results for *speech-fr* the scanning distances could be reduced by more than 50%, with respect to the static linear scan, and by more than 30%, compared to the line-column method.

6.3.2 Word prediction

In the last two chapters the performance of the baseline word predictor as well as of the user and the semantic adaptation models have been extensively evaluated. Now, we focus our evaluation on some user-related parameters such as the filtering of shown words, vocabulary reduction or partial selection, and afterwards we will present the results for all methods combined in order to investigate the interaction between the different adaptation methods. As before, we will base the quantitative evaluation on the test corpora that have been presented and applied in the previous chapters.

Filtering of already shown words

It is obvious that removing already shown words (ASW) from the list enhances keystroke savings, but, as discussed above, some users have difficulties to select words immediately and therefore prefer to leave them. In this regard it is interesting to know, what difference the ASW filtering makes. We therefore calculated keystroke savings for the newspaper corpora (using the baseline 4-gram predictor, cf. section 4.5.1) with and without filtering. Table 6.2 displays the results.

	news-fr	*news-de*	*news-en*
ASW unfiltered	56,90%	50,26%	54,66%
ASW filtered	57,78%	51,56%	55,49%
Difference	+0,88%	+1,30%	+0,83%

Table 6.2: Filtering of already shown words (ASW); baseline results (ksr_5) for the newspaper corpora

The advantages of ASW filtering (ksr_5) range from +0,8% to +1,3%, which represents a moderate but significant gain. The German corpus shows the highest gains, which can be explained by the higher average word length in German (5,3 characters vs. 4,5 characters for French and English, cf. section 2.5.1): The longer words are (on average) the longer they can remain in the list and occupy space.

Considering these rather small differences, it cannot be clearly decided whether filtering should be applied or not. A confident user will surely prefer not to see a proposed word twice, however she or he has to spend more attention, which increases the cognitive load in the long term.

Vocabulary limitation

Taking into account Zipf's law on the frequential distribution of words, the effect of frequency-based vocabulary limitation can be predicted: The higher the frequency rank of the words which are excluded, the stronger the effect becomes. Figure 6.9 displays the performance curves (ksr_5) for two corpora with increasing limitation (from 141.078 down to 5.000 words), and it confirms our expectation.

The curves look very much the same, and they seem to follow a logarithmic trend: Excluding the 10.000 words with the lowest frequencies has a very small effect on ksr (app. -0,08%); however we observe ksr_5 loss of nearly 3% when the vocabulary is reduced from 20.000 to 10.000 words. The progression of these curves has to be taken into account when the vocabulary for a user is to be limited. It has however also theoretical implications: Regarding the slope in the first part of the curves, one can imagine the effect that vocabulary

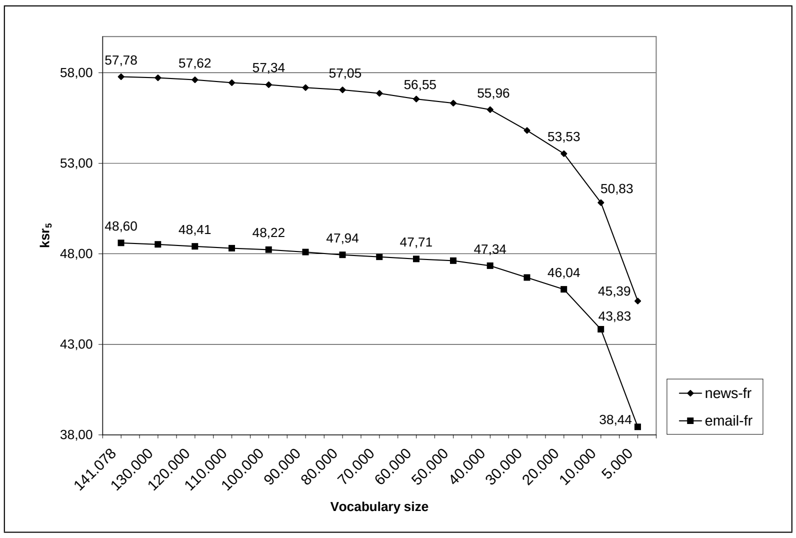


Figure 6.9: Effects of vocabulary limitation on keystroke savings (ksr_5); baseline results (ksr_5) for *news-fr* and *email-fr*

extension might have. So, given the vocabulary were to be doubled, we can expect keystroke savings to increase by less than 1%. This expectation underlines once again the importance of a user-oriented vocabulary extension: As could be seen in chapter 4, the user lexicon, integrating all unknown words could achieve similar gains (up to +1,5%), however – considering the OOV reduction – only a few hundred words were added. In this light an auto-adaptive (or adaptable) lexicon shows to be a more efficient strategy than a general, blind augmentation of the vocabulary.

Partial selection

As mentioned above, partial selection offers a simple way to deal with compound words. Evaluating partial selection by emulation is however not as easy to achieve as for word-based methods. It presumes that the user applies an optimal selection strategy which in practice is more difficult than simply scanning a prediction list to see if a word matches. Saving rates can even decrease when simply every word onset is matched, because it then takes two more selection steps to choose the following element (1 back step + 1 selection). The results in Table 6.3 display the optimal gains, i.e. PS was only applied when it could decrease the number of keystrokes to be typed. Since partial selection is mostly useful in handling the insertion of German compound words, we only display the results for all German corpora here; to get an idea of the effect of partial selection in other languages, we still included the French and English newspaper corpora in the evaluation. The results are again based on the 4-gram predictor (baseline, cf. section 4.5.1).

	news-de	*lit-de*	*speech-de*	*email-de*	*news-fr*	*news-en*
PS off	51,56%	44,94%	49,10%	47,98%	57,78%	55,49%
PS on	53,05%	46,08%	49,99%	48,80%	57,94%	55,82%
Diff.	+1,49%	+1,14%	+0,89%	+0,82%	+0,16%	+0,33%

Table 6.3: Results (ksr_5) using partial selection

For the partial selection method we can observe stable gains of +0,8 to +1,5% for all German corpora. This is somewhat less than the results of Trost *et al.* (2005) who report higher gains of app. +3% for a similar strategy, but it still represents a significant improvement. Interestingly, partial selection also seems to have a slightly beneficial effect for French (+0,16%) and English (+0,33%), but as we have seen in chapter 4, compounding sometimes occurs in these languages as well (3% of the OOV words in *news-fr* and 16% in *news-en* were compounds).

We can therefore conclude that, even though that partial selection is a rather simple solution to the problem of compounding, it has a beneficial effect, and it performs significantly better than the *split compound* model proposed by Baroni *et al.* (2002) who report a ksr improvement of up to 0,3% only.

Combining strategies

In NLP research it has become habitual to evaluate several techniques separately against a common baseline. This is reasonable practice, since it assures reproducibility and it excludes the possibility of interactions. We thus followed this paradigm in the past chapters.

However, as we consider in this chapter the system as a whole, it is important to see to what extent the implemented adaptation techniques complement each other, i.e. how much the information captured by the different methods overlaps. Therefore, we also evaluated the word predictor, including user and semantic adaptation[8] as well as partial selection. Table 6.4 shows the overall results with all strategies combined.

Considering the results, we can observe that keystroke savings for *SibyWord* (using all strategies) remain well over 50% in all test corpora and for all languages. The gains, compared to the (already well performing) 4-gram baseline are quite remarkable, especially in the registers not being used for training. The speech corpora always show the highest gains (over +9%), which is mostly due to the repetitive structures, frequently found in oral communication, which are well captured by the dynamic user model. Smaller gains are observed for the newspaper corpora; this was expected due to the high simi-

[8]The adaptation methods applied are based on the *Dynamic User Model* (cf. section 4.3.3) and LSA (cf. section 5.3), as described before.

Corpus	Baseline	*SibyWord* (All On)	Diff.
news-fr	57,8%	59,4%	+1,5%
lit-fr	46,0%	52,2%	+6,2%
speech-fr	48,3%	57,9%	+9,6%
email-fr	48,6%	53,8%	+5,2%
news-de	51,6%	56,9%	+5,3%
lit-de	44,9%	51,8%	+6,9%
speech-de	49,1%	58,4%	+9,3%
email-de	48,0%	53,1%	+5,1%
news-en	55,5%	57,6%	+2,1%
lit-en	49,8%	54,4%	+4,6%
speech-en	48,5%	57,7%	+9,2%
email-en	49,4%	54,8%	+5,4%

Table 6.4: Performance (ksr_5) of *SibyWord* (+DUM, +LSA, +PS) on all test corpora and differences with the baseline (4-gram)

larity with the training data.

With respect to language, the overall highest gains can be observed for German, here even the newspaper corpora receive an advantage of more than +5%. On the one hand this underlines the effectiveness of partial selection, enabling to treat compound words, but on the other hand this shows in particular, how well the different strategies work together. Here the gains are nearly additive, but also for the other languages we can observe that the methods complement each other (although to a smaller extent). In no case we can recognize any negative interaction between the different techniques.

Altogether it can be stated that the application of the different adaptation techniques reduces the variability of performance in an important way; the problem of training dependency (as discussed in chapter 4) can be alleviated. Our word predictor has theoretically proven high performance for a variety of rather different communication situations and language styles. The practical perspective should now be looked into.

6.4 User assessment

6.4.1 Application in a rehabilitation center

SIBYLLE benefits from the experience of seven years of daily use in the rehabilitation center of Kerpape (Brittany, France). This center receives adult patients and children requiring reeducation or rehabilitation care within the framework of a full-time hospital, a day hospital or an outpatient service. The multi-disciplinary team of professionals (physio- and ergotherapists, speech therapists, orthoptists, teachers and technicians) aims to optimize the independence as well as the social and professional reinsertion of its patients.

When a communication-impaired patient arrives at Kerpape, she or he meets all of the interacting staff, who try to determine her or his specific needs by carrying out a number of experiments. The speech therapists analyze the patient's linguistic abilities and thereby find out which kind of AAC will be most suitable (e.g. use of an iconic, phonetic or alphabetic keyboard). The ergotherapists determine the functional and motor capacities of the patient in order to define the most appropriate input device as well as the selection modes of the AAC system. The orthoptists then analyze the patient's visual abilities to ensure that all elements of the interface are clearly perceptible. When the basic parameter settings are found, the technical staff then configures the AAC system accordingly; this step is of course performed iteratively and in close collaboration with the patient.

Such an adaptation process can take a considerable amount of time; especially in the case of visual disability it involves several months of intense work with the patient until the optimal configuration can be found; yet according to the practitioners at Kerpape, it will eventually be found with SIBYLLE, due to its far-reaching configurability (cf. 6.1.1).

Moreover, due to the intense cooperation with the speech- and ergotherapists it was possible to take the needs of the users and their attendants into consideration for the development of our system; in this way a number of (rather inconspicuous) features were developed that proved to be very useful for the interaction with the system, such as the miscellaneous keypad including *emergency* phrases (cf. 6.1.1), the timing line or the possibility of vocabulary reduction (cf. 6.1.3).

Successive versions SIBYLLE have been used by more than twenty patients, most of which have cerebral palsy or suffered from encephalitis in early childhood. Their clinical patterns include quadriplegia, anarthria (inability of phonation) and sometimes amblyopia (visual impairment). Some of them are adults, but the majority are children and adolescents from the school integrated in the center.

The system was highly appreciated by most users; only two of them, who are visually strongly impaired, felt uncomfortable with the dynamic rearrangement of the keyboard. But even in these severe cases the practitioners could configure the system in a way (i.e. by selecting a static keyboard layout, appropriate colors and font size, and by optimizing the placement of the keypads) that would benefit the users.

The high configurability seems to be the particular strength of SIBYLLE (from a usability point of view). Moreover, the linguistic facilities of the system are able to evolve with the user's capacities and needs: A user can start with a very simple static configuration and then successively use more advanced features in order to speed up his/her communication rate without changing the interface. And indeed the teachers of the Kerpape school could observe a significant acceleration of the text insertion process after their students had started to use SIBYLLE. They also observed that the children accept longer working sessions. This indicates that the use of SIBYLLE implies less physical fatigue, compared to the AAC systems that were previously used in the center. The reduction of the physical fatigue of the users is certainly as important as the improvement of the communication speed (cf. Berard & Neimeijer, 2004).

Finally, we have also noticed a significant decrease of orthographic and grammatical errors when the patients are using the system. A comparable result has already been observed with users of other word-based AAC systems (cf. Morris *et al.*, 1992; Carlberger *et al.*, 1997). This observation applies in particular when the user has additional language impairments.

Despite these encouraging user experiences, a disturbing observation is that some users do not select the intended word even though it is clearly present in the prediction list; similar observations have been made in the study of Biard *et al.* (2006). Our discussions with the users and the practitioners tend to show that this situation, which obviously limits the number of saved keystrokes and likewise the communication speed, is due to a cognitive problem (cf. Koester & Levine, 1994): the users have difficulties writing a message and reading the list simultaneously. A possible solution to this problem could be to implement direct completion like in the VITIPI system (Boissière, 2000): instead of presenting a list of several word hypotheses on a specific sub key-pad, one can propose the most probable termination of the current word immediately after the latest typed letter.

But as we have already pointed out, due to the strongly differing physical preconditions, each user has her or his own preferences and needs, therefore there is no single optimal solution for the interface of an AAC device; only offering a multitude of possible configurations can respond to the various demands of AAC users.

We acknowledge that the observations made so far with the users of SIBYLLE are still rather coarse-grained and unsystematic. However, as already discussed in chapter 2, systematic user assessment in AAC is very hard to achieve, since it requires a large and rather homogeneous user base (i.e. users having similar impairments and demands), which is already difficult to acquire. Moreover, as AAC users cannot easily be interviewed or asked to fill in questionnaires, such studies require considerable effort. For these reasons we had to constrain the user assessment in the present work to descriptive observations of the users and the practitioners at Kerpape. The collaboration with the center will however continue, and we will also intensify our cooperations with other centers: For example, since the end of 2007 SIBYLLE is also used at the *Fraternité Chrétienne des Handicapés*, Papeete (Tahiti); in the framework of the already mentioned VOLTAIRE project we collaborate with the *Plateforme Nouvelles Technologies* at the hospital *Raymond Poincaré*, Garches (France). In this way we can hope to achieve a more user-centered evaluation in the future.

Another step towards a more systematic AAC user assessment was the project *ESAC-IMC*, a joint effort of several universities and research centers[9], which has been completed in 2007. It mainly aimed at:

- elaborating a typology of language troubles related to cerebral palsy and characterizing the requirements of each type of user with respect to an AAC system;
- collecting corpora and clinical profiles from AAC users in order to better understand their communication needs;
- establishing a common evaluation framework that applies to various AAC systems and facilitates direct comparisons.

A common XML interchange format was defined for the log files that are being recorded during the evaluation campaign. These log files keep track of:

- all actions performed by the user, such as the keys pressed, the text written, possible corrections and time stamps
- all replies and actions generated by the system (including predicted textual elements etc.)

The recordings with users from the rehabilitation center of Kerpape have begun recently, it can be hoped that their analysis will bring new insight in the way how AAC users actually communicate and in what sense systems can be improved.

6.4.2 Analysis of user input

While, for the time being, we cannot present any results from this evaluation campaign, we want to take a closer look onto some user data that have been collected manually in Kerpape (saved text files). Most of the input was created by children and adolescents who have been disabled from birth or early childhood on. Due to their strong impairment they acquired language in a much more passive way, by listening rather than practicing. This implies that their grammatical and lexical capacities are often retarded or even remain non-standard, despite full cognitive abilities. Existing AAC techniques can however only partially support a user, whose linguistic capacities do not conform to standard language. Until now the aspect of ungrammaticality has rarely been treated by researchers in this domain, it is however obvious that it has to be addressed in order to provide better assistance.

To illustrate this, let us regard an example from our corpora, written by an adolescent affected by cerebral palsy (from a birth defect):

[9]The project was financed by *Fondation Motrice*; the participating institutions were: IRIT Toulouse, LI Tours, VALORIA Vannes, CMRRF Kerpape, Bretagne; http://www.irit.fr/ESACIMC/

Original utterance: *tu a vu quil ya des famille qui porte plinte acose l'handicap de leur fils. j'ai pas tous de suite compris les tenen e les abouticen mais papa ma espliqué. je pence si on fait tous ça les génécologue serai derier les baros souven.*

Transliteration: *tu as vu qu'il y a des familles qui portent plainte à cause (de) l'handicap de leur fils. Je n'ai pas tout de suite compris les tenants et les aboutissants, mais papa m'a expliqué. Je pense (que) si on fait tous ça, les gynécologues seraient derriére les barreaux (plus) souvent.*

English: you have seen that there are families going to court because of the handicap of their son. I have not immediately understood the conditions and circumstances, but dad explained to me. I think that if all of us do that, the gynecologists would often be behind bars.

This short paragraph is quite representative for the kind of language we are confronted with. A first remarkable aspect of this short passage is its degree of elaborateness. We find relative clauses, idiosyncratic expressions (*"les tenen e les abouticen"*), use of tense and conditional mood (*'serai'*). This confirms once more the cognitive capacities of the author, who is able to express himself in a complex way. On the other hand we find a large number of grammatical and orthographical errors, which are however systematic in that the phonetic realization of the utterance is (almost) correct (e.g. *'plinte'/'plainte'* $\rightarrow$ [plɛ̃t]; *'baros'/'barreaux'* $\rightarrow$ [baʀo]). This phonetic style of writing is very typical for AAC language, it poses however a severe problem for a word prediction system. Obviously the user was reluctant to select the words from the prediction list, even though they were clearly present. We do not know if this is due to the already mentioned problem of task simultaneity or if the lexical representations of the user are deviant, so that he does not find the intended item in the list and continues selecting characters. However, as this kind of error is systematic in the above mentioned sense, one could develop a module that aims to replace the phonetic variant by its corresponding lexical form.

Another aspect that we have observed in our user data can be seen in the following example; the phrases have been uttered by a girl in a discussion group at the Kerpape school.

Original utterance: *je veux décrire Michelle être grosse être gentille être de bonne humeur être grande être forte être fille être de valeur. je ne peux pas dire pourquoi.*
je avoir mal aux épaules.
je être fatiguée.

Transliteration: Je veux décrire Michelle: elle est grosse, gentille, de bonne humeur, grande, forte, une fille et elle est une personne de valeur. Je ne peux pas dire pourquoi.
J'ai mal aux épaules.
Je suis fatiguée.

English: I want to describe Michelle: she is fat, friendly, good-humored, tall, strong, a girl, estimable. I cannot say why.
My shoulders hurt (*lit.*: I have (*inf.*) bad at the shoulders)
I am tired. (*lit.*: I be tired.)

The irregularities found in these phrases are again systematic: Many verbs are often not inflected, a typical simplification strategy, which is also often applied by foreign language learners. This behavior may be explained by two reasons: Either the user applies this simplification because she does not have learned proper inflection, or she has developed a strategy to enter the infinitive form more quickly (or with less effort) than one of the inflected forms. We deem the second explanation more probable, since we do find inflection in her utterance (*"je veux", "je peux"*); moreover the infinitive form is only used for the two most frequent verbs (*'être'* et *'avoir'*). This gives us already a hint how this problem can be addressed: One could imagine to integrate a correction routine replacing the infinitive by the intended inflected form. To some extent this could already be performed by our disabbreviation module, but more elaborate strategies should be developed in the future.

To get an impression how our word predictor deals with such *real* user data, we also used them as test corpora. We first separated the e-mails from the discussion group data, so that register influences can be recognized, but since such a separation is not realistic, we also tested on all data. The corpora are not very large anyway: the e-mail corpus consists of 1416 words, the discussion data of 855 words. This is certainly too small to make strong conclusions, still we can observe some important trends. Table 6.5 shows the ksr_5 results for these corpora, tested on the baseline model (4-gram) as well as on *SibyWord*:

Register	# words	OOV (%)	Baseline	*SibyWord* (+DUM/+LSA)
E-Mail	1416	20,6%	31,8%	37,2% (+5,4%)
Discussion	855	3,0%	50,9%	60,2% (+9,3%)
All	2271	14,1%	38,9%	45,1% (+6,2%)

Table 6.5: Results (ksr_5) of two user corpora

The results are quite contrastive: Whereas we observe an important loss of performance for the e-mail data (ksr_5: 37,2%), the results of the discussion data are comparable to the previously presented results (ksr_5: 60,2%). However, looking at the proportions of out-of-vocabulary words (OOV) we can immedately see, why the e-mail data score so much worse. Here, every 5^{th} word is unknown (20,6%), whereas the discussion data contain only 3% of OOV words. This underlines the necessity for the application of automated correction procedures, as mentioned above. Some work on automatic correction of AAC input has been presented by Boissière & Dours (2001) as well as by Sitbon *et al.* (2007).

Another point to be observed in Table 6.5 is the advantage of *SibyWord* over the baseline. While the corpora themselves deviate strongly from the test corpora applied beforehand, the advantages of *SibyWord*are quite similar (+5,4% to +9,3%). This is astonishing since the corpora are much smaller, implying that the dynamic user model had much less data to train on. Still we measure for the discussion corpus the highest ksr_5 score of all corpora tested in this work (60,2%). As mentioned above, we do not want to make strong claims from these results, still they indicate that the adaptive features of *SibyWord* are indeed useful in a real-word situation. We saw however as well that it should be combined with automated correction procedures in order to reduce the amount of grammatical and lexical irregularities.

6.5 Conclusion

This chapter was devoted to the presentation of SIBYLLE, our AAC system which has been developed and applied since 2001. We described in detail the user interface and its configurability, the dynamic (prediction-based) key selection component *SibyLetter* as well as our adaptive word predictor *SibyWord*. After sketching the software architecture we discussed some issues and design criteria concerning the implementation of the system.

We then presented results of an extended quantitative evaluation of the communication enhancing parts of the system: The *dynamic key selection* shows an excellent performance, compared to static key selection methods; the *average scanning distance* ranges from 3 to 4 letters, which represents a 30% improvement with respect to a static (line-column) approach.

The keystroke saving capacities of the word predictor are also remarkable; the results range among the best that were so far reported in the literature (cf. sections 4.5.5 and 5.5.5), especially for German and French (ksr_5 from 51,8% to 59,4%). Moreover, the adaptation strategies of the word predictor significantly reduce the variability of performance over different registers. In the end we also presented some findings from the application of SIBYLLE at the rehabilitation center of Kerpape. According to the practitioners, the system is well appreciated by the users, particularly due to its high configurability, which enables to individually adapt its layout and its functions to the users' capacities and preferences.

In the last part we have discussed important features of real-use data, acquired from the application of SIBYLLE in Kerpape. Here, we saw that the utterances produced contain an important amount of grammatical and lexical irregularities, many of which are however systematic: A large part of the lexical errors are phonetic variants of the intended word, and grammatical errors are often due to missing verb inflection. While these aspects certainly have a negative impact on prediction, we showed that the adaptive features of *SibyWord* are able to partially compensate for them. Large improvements should however be achieved with the application of automated correction routines.

Chapter 7

Final remarks

Trigram models are state-of-the-art.
Trigram models are obviously braindamaged.
We did something slightly less braindamaged.

JOSHUA GOODMAN
(*A bit of progress*, 2001)

7.1 Conclusion

In the introduction three major objectives were formulated for this thesis. They concern: (i) the investigation of adaptive models for word prediction, (ii) the development of a *usable* assistive communication system, and (iii) the portability of this system to other languages. How far have we come?

We have evaluated two kinds of adaptive models, those adapting to the user's communication style and those adapting to the semantic context. With respect to the first group of models we have shown the *Dynamic User Model* (DUM) to perform very well: it is able to enhance keystroke savings to a considerable extent for very different test corpora (up to +9,4%) and to reduce the rate of out-of-vocabulary words by half. Moreover, the model could be shown to adapt quickly: After having integrated as little as 2.000 words an improvement of performance could be determined. The other two models (cache and user lexicon) also showed reliable improvements, but their gains were inferior in comparison to the DUM.

The other type of models aimed to adapt to the semantic context. Here we focused in particular on *Latent Semantic Analysis* (LSA), a technique which has achieved convincing results for measuring semantic similarity in a large variety of tasks. We addressed the problem of integrating information stemming from LSA with a general language model by proposing three kinds of approaches: a trigger approach, a n-best reranking method and different forms of interpolation. For the latter we developed a confidence scheme, based on the cluster density of a term in the semantic space. This weighting scheme in combination with geometric interpolation proved to perform best on all test corpora and languages. All in all the improvements achieved by the LSA-enhanced prediction model were not as high as for the DUM (+1,0% - +1,7% in ksr_5), these improvements were however very stable and could be shown to be almost complementary to the DUM.

The improvement achieved by the two adaptation methods (DUM+LSA) in combination is convincing. Especially on corpora from registers other than the training corpus we could see significant improvements. A major problem of stochastic language models could therefore be addressed: By adaptation the effect of training dependency can be reduced to a large extent.

Concerning the second objective, we refer to SIBYLLE, an assistive communication system, which incorporates the adaptive models investigated before. In order to meet the individual requirements of many users, we spent a lot of care on the configurability of the system. Apart from the word predictor a number of other components enhancing communicative abilities were implemented, e.g. a keypad comprising predefined phrases and a module for abbreviation expansion. In addition the system was enabled to work with external applications. This is a feature of special importance, since AAC users are particularly reliant on computer-based forms of communication (e.g. chatting or e-mailing).

SIBYLLE has shown its practical usability during several years of use in the rehabilitation center of Kerpape. Its high configurability enables the practitioners to find a suitable configuration for every patient, even in cases of severe visual impairment. Moreover, the facilities of the system can evolve with the user's development: At first the user can apply only very basic features, and after some time of training she or he can start to make use of more complex communication-enhancing techniques. Such a 'co-evolution' of the system is especially meaningful when applied with children.

The portability aspect has been addressed in a two-fold manner: On the one hand all models were tested on corpora from three languages: French, German and English. Whereas the results, relative to the corresponding baselines, were quite similar, the absolute results for German showed to be significantly lower. An analysis of the out-of-vocabulary words indicated that this is mainly caused by productive compounding: The OOV rates for German were roughly twice as high as for French or English, and for some corpora half of the OOV words were compounds. For this reason we implemented a partial selection strategy that enables to predict (and select) compounds part-wise and agglutinate them. This strategy led to some improvement in the results (up to +1,5% in ksr_5).

On the other hand the architecture of SIBYLLE was made generic, so that language-dependent and -independent components were kept separate from each other. Then the three afore mentioned languages were integrated into the system as well as the partial selection method. This enables SIBYLLE to be used by three important language communities. Furthermore, as for only data-based approaches are applied for prediction, the generic architecture of the system allows for introducing other (similar) languages with minimal effort.

7.2 Perspectives

As time is always too short, as the patience of supervisors and the endurance of candidates is finite, many ideas and plans remain unrealized at the end of a PhD thesis. Most of them are lost, but a few at least manage to enter the 'perspectives' section, waiting to get picked up from there some time later. We try to do this here, and we form two bundles, corresponding to the two research areas where we started from.

Looking to the NLP side, we dare to claim that the invisible wall of prediction performance that we have mentioned in the second chapter (2.5.2) has come quite near. We have tried many different training corpora, smoothing methods, adaptive and other optimization strategies, and we could see that significant improvements will not be accomplished anymore using stochastic language models of the n-gram type. Therefore the application of new models is needed. One promising approach is probably to apply n-grams of variable length (so called *multigrams*, cf. Deligne & Bimbot, 1995). It is clear that the predictive power of the context is not constant, therefore its length should be determined by information-theoretic means. Likewise the model should be enabled to predict more than one word. Frequently recurring phrasal elements like *"you're welcome"* or *"s'il te plaît"* and multi-word units should be predicted at once. Here methods for the identification and integration of such units ought to be investigated.

Another very auspicious approach has already been mentioned: Maximum entropy models (cf. 3.6) offer an optimal way to combine information from various information sources, they are guaranteed to find the optimal model for the given training data, and they allow for an optimal adaptation to user input (cf. section 4.2.2). However, as mentioned before, such models are (at present) computationally too demanding for large-scale applications. Still, this path is worthwhile to follow, either in order to find strategies for cutting down the computational cost (cf. for example Chen *et al.*, 1998; Peters & Klakow, 1999; Chen & Rosenfeld, 2000) or to wait until computational power has achieved the level needed for ME approaches.

A more theoretic construction site remains in the optimization of training models. Most data-based NLP approaches employ in a rather blind fashion large corpora of general language for training (such as we did). However a thorough model of language generality is far from being conceived. Many corpora (such as the BNC) claim to contain a representative sample of English, but this may only be true with respect to qualitative considerations; from a quantitative perspective, present corpora are everything but *well-balanced*. Therefore truly reliable probability estimates, those on which most information-theoretic assumptions are based (cf. the *Shannon-McMillan-Breiman* theorem; section 3.1.1) cannot be determined for human language. Corpora like the BNC are probably leading into the right direction, their distributional properties however have to be controlled better. Moreover, such corpora still remain to be developed for other languages.

An interesting research direction is here of course the exploitation of the internet as a corpus, since no larger and no more diverse and general corpus is available. However, it also offers a lot of pitfalls, which is nicely shown in Kilgarriff's (2007) article on *Googleology*.

Turning to the AAC side, the most promising element of research is surely the user interface. Here an interesting idea would be to extend the scope of adaptation: One could imagine an interface taking into account every action by the user and reconfigurating its display accordingly. A simple example might be to equip each key with a click count, and this count could be used to minimize the scanning distance by arranging the keys with respect to their selection frequency.

A major weakness of our system (as of many others) still concerns the integration of the word prediction component. As mentioned in the last chapter (cf. 6.4) we recognized in the work with the users that words are sometimes not selected even though they are clearly visible in the list. We gave two possible explanations for this behavior: On the one hand the attention of the users might be consumed by either the text insertion or the key selection process, so that they neglect the prediction list (problem of task simulaneity). Possible solutions were already mentioned: One could include the prediction directly within the editor (direct completion) like in Boissière's VITIPI system, or in the letter keypad, so that the user will only have to focus her/his attention to the selection frame. On the other hand users might have, due to difficulties in language acquisition, a deviating orthographic representation of the intended word (the '*phonetic*' style of writing, as illustrated in section 6.4.2). In this case prediction will not help, since misspelled words can hardly be predicted. However, to assist the text production process, it could be reasonable to integrate a correction module that is able to replace a misspelled phonetic variant by its orthographically correct form.

In any case, something that we have clearly felt during our work is that significant improvement in the area of AAC can only be achieved when research is more firmly centered on the user and her or his capacities and needs. In most research disciplines abstraction from real life is the crucial means for progress. AAC however is an area that can only be pursued with real people, and it will make the greater advances the more it focuses on them.

Bibliography

ABNEY, S. 1991. Parsing by chunks. *In:* BERWICK, R., S. ABNEY, & TENNY, C. (eds), *Principle based parsing*. Kluwer Academic.

AÏT-MOKHTAR, S., CHANOD, J.-P., & ROUX, C. 2002. Robustness beyond shallowness: incremental deep parsing. *Natural Language Engineering*, **8**(2/3), 121–144.

ANSON, D. K., MOIST, P., PRZYWARA, M., WELLS, H., SAYLOR, H., & MAXIME, H. 2005. The effects of word completion and word prediction on typing rates using on-screen keyboards. *In: Proceedings of RESNA'05*. Arlington, Virginia: RESNA Press.

ASHA. 2002. *Roles and responsibilities of speech-language pathologists with respect to alternative communication: Position statement*. American Speech-Language-Hearing Association (ASHA) - Supplement 25.

BACCHIANI, M., & ROARK, B. 2003. Unsupervised language model adaptation. *Pages 224–227 of: Proceedings of ICASSP'03*.

BAKER, B. 1982. Minspeak. *Byte*, **7**(9), 186–202.

BARONI, M., MATIASEK, J., & TROST, H. 2002. Wordform- and class-based prediction of the components of German nominal compounds in an AAC system. *In: Proceedings of the 19th COLING*.

BASILI, R., & ZANZOTTO, F. M. 2002. Parsing engineering and empirical robustness. *Natural Language Engineering*, **8**(2/3), 97–120.

BAUBY, JEAN-DOMINIQUE. 1997. *Le scaphandre et le papillon*. Robert Laffont.

BÉCHET, F., DE MORI, R., & JANISZEK, D. 2004. Data augmentation and language model adaptation using singular value decomposition. *Pattern Recognition Letters*, **25**(2004), 15–19.

BÉCHET, N. 2006. *HandiAS, un système de prédiction linguistique de mots*. Tech. rept. Rapport de Master Recherche, Université de Tours.

BECK, C., SEISENBACHER, G., EDELMAYER, G., & ZAGLER, W.L. 2004. First user test results with the predictive typing system FASTY. *In: Proceedings of ICCHP'04*.

BELATAR, M., & POIRIER, F. 2007. Uniglyph : une méthode universelle pour la saisie de texte sur dispositifs mobiles. *In: Actes IHM'07*.

BELLEGARDA, J. 1997. A Latent Semantic Analysis framework for large-span language modeling. *Pages 1451–1455 of: Proceedings of Eurospeech'97*.

BELLEGARDA, J. 2000. Exploiting latent semantic information in statistical language modeling. *Proceedings of the IEEE*, **88**(8), 1279–1296.

BELLEGARDA, J. 2004. Statistical language model adaptation: review and perspectives. *Speech Communication*, **42**(2004), 93–108.

BERARD, C., & NEIMEIJER, D. 2004. Evaluating effort reduction through different word prediction systems. *In: Proceedings of the IEEE Int. Conference on Systems, Man and Cybernetics*, vol. 3.

BERGER, A. 1998. *Convexity, maximum likelihood and all that*. Tutorial, Carnegie Mellon University. Available online at: http://www.cs.cmu.edu/-afs/cs/user/aberger/www/ps/convex.ps.

BERGER, A., DELLA PIETRA, S., & DELLA PIETRA, V. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, **22**(1), 1–14.

BERRY, M. 1997. Survey of public-domain Lanczos-based software. *Pages 332–334 of:* BROWN, J., CHU, M., ELLISON, D., & PLEMMONS, R. (eds), *Proceedings of the Cornelius Lanczos centenary conference*.

BERRY, M., DRMAC, Z., & JESSUP, E. 1999. Matrices, vector spaces, and information retrieval. *SIAM Review*, **41**(2), 335–362.

BIARD, N., DUMAS, C., BOUTEILLE, J., POZZI, D., LOFASO, F., & LAFFONT, I. 2006. Apports de l'évaluation en situation de vie à partir d'une étude sur l'intérêt de la prédiction de mots auprés d'utilisateurs de synthèse vocale. *Pages 132–137 of: 4ème conférence nouvelles technologies au service de l'homme*.

BIBER, D. 1988. *Variation across speech and writing*. Cambridge University Press.

BIBER, D. 1993. Using register-diversified corpora for general language studies. *Computational Linguistics*, **19**(2), 219–241.

BIGI, B., BRUN, A., HATON, J.-P., SMAÏLI, K., & ZITOUNI, I. 2001a. Dynamic topic identification: Towards combination of methods. *Pages 255–257 of: Proceedings of the Recent Advances in NLP workshop*.

BIGI, B., BRUN, A., SMAÏLI, K., & HATON, J.-P. 2001b. A hierarchical approach for topic identification. *In: Proceedings of the international workshop on speech and computer*.

BIRBAUMER, N., KUBLER, A., GHANAYIM, N., HINTERBERGER, T., PERELMOUTER, J., KAISER, J., IVERSEN, I., KOTCHOUBEY, B., NEUMANN, N., & FLOR, H. 2000. The thought translation device (TTD) for completely paralyzed patients. *IEEE Transactions on Rehabilitation Engineering*, **8**(2), 190–193.

BLACHE, P., & RAUZY, S. 2006. Mécanismes de contrôle pour l'analyse en grammaires de propriétés. *In: Actes de taln'06*.

BLACHE, P., & RAUZY, S. 2007a. Le module de reformulation iconique de la Plateforme de Communication Alternative. *In: Actes de TALN'07*.

BLACHE, P., & RAUZY, S. 2007b. Le moteur de prédiction de mots de la Plateforme de Communication Alternative. *Traitement Automatique des Langues*, **48**(2), 47–70.

BLISS, C. K. 1965. *Semantography (Blissymbolics)*. Sydney, Australia: Semantography Press.

BOISSIÈRE, P. 2000. VITIPI : Un système d'aide à l'écriture basé sur l'auto-apprentissage et adapté à tous les handicaps moteurs. *Pages 81–86 of: Actes HANDICAP'00*.

BOISSIÈRE, P., & DOURS, D. 2000. VITIPI: A universal writing interface for all. *In: Proceedings of the 6th ERCIM workshop*.

BOISSIÈRE, P., & DOURS, D. 2001. Comment vitipi un système d'assistance à l'écriture pour les personnes handicapées peut offrir des propriétés intéressantes pour le taln ? *In: Proceedings of taln'01*.

BOISSIÈRE, P., & DOURS, D. 2002. Proposal of an evaluation framework for writing assistance systems: Application to VITIPI. *Pages 276–278 of: Proceedings of ICCHP'02*.

BRANIGAN, H., & PEARSON, J. 2006. Alignment in human-computer interaction. *Pages 140–157 of:* FISCHER, K. (ed), *Proceedings of the Hanse Wissenschaftskolleg workshop "How People Talk to Computers, Robots, and Other Artificial Communication Partners"*.

BRANTS, T., POPAT, A., XU, P., OCH, F., & DEAN, J. 2007. Large language models in machine translation. *Pages 858–867 of: Proceedings of EMNLP'07*.

BRESNAN, J. 2001. *Lexical functional syntax*. Blackwell Publishers.

BRILL, E., & MOORE, R. C. 2000. An improved error model for noisy channel spelling correction. *In: Proceedings of the 38th ACL*.

BRILL, E., FLORIAN, R., HENDERSON, C., & MANGU, L. 1998. Beyond n-grams: Can linguistic sophistication improve language modeling? *In: Proceedings of the 36th ACL*.

BROWN, P., COCKE, J., DELLA PIETRA, S., DELLA PIETRA, V., JELINEK, F., LAFFERTY, J., MERCER, R., & ROOSSIN, P. 1990. A statistical approach to machine translation. *Computational linguistics*, **16**(2), 79–85.

BROWN, P., DELLA PIETRA, V. J., MERCER, R. L., DELLA PIETRA, S. A., & LAI, J. C. 1992. An estimate of an upper bound for the entropy of english. *Computational Linguistics*, **18**(1), 31–40.

BURNARD, L., & ASTON, G. 1998. *The BNC Handbook: Exploring the British National Corpus with SARA*. Edinburgh University Press.

CARLBERGER, A., CARLBERGER, J., HUNNICUTT, S., PALAZUELOS-CAGIGAS, S., & AGUILERA-NAVARRO, S. 1997. Profet, a new generation of word prediction: An evaluation study. *In: Proceedings of Natural Language Processing for Communication Aids*.

CEDERBERG, S., & WIDDOWS, D. 2003. Using LSA and noun coordination information to improve the precision and recall of automatic hyponymy. *In: Proc. of CoNNL'03*.

CHARNIAK, E. 2001. Immediate-head parsing for language models. *In: Proceedings of the 39th ACL*.

CHELBA, C. 1997. A structured language model. *Pages 498–503 of: Proceedings of the 35th ACL and 8th EACL*.

CHELBA, C., & JELINEK, F. 1998. Exploiting syntactic structure for language modeling. *Pages 225–231 of: Proceedings of the 36th ACL.*

CHELBA, C., & JELINEK, F. 2000. Structured language modeling. *Computer speech and language*, **14**(4), 283–332.

CHEN, L., GAUVAIN, J.-L., LAMEL, L., & ADDA, G. 2003. Unsupervised language model adaptation for broadcast news. *Pages 220–223 of: Proceedings of ICASSP'03.*

CHEN, S., & GOODMAN, J. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, **1999**(13), 359–394.

CHEN, S., & ROSENFELD, R. 2000. A survey of smoothing techniques for ME models. *IEEE Transactions on Speech and Audio Processing*, **8**(1), 37–50.

CHEN, S., SEYMORE, K., & ROSENFELD, R. 1998. Topic adaptation for language modeling using unnormalized exponential models. *In: Proceedings of ICASSP'98.*

CHOI, J.-K., & OH, Y.-H. 2006. N-gram adaptation with dynamic interpolation coefficient using information retrieval technique. *IEICE Transactions on Information and Systems*, **E89-D**(9), 2579–2582.

CHOMSKY, N. 1956. Three models for the description of language. *IRE Transactions On Information Theory*, **IT**(2), 113–124.

CHOMSKY, N. 1969. Quine's empirical assumptions. *Pages 53–68 of:* DAVIDSON, D., & HINTIKKA, J. (eds), *Words and objections. Essays on the work of W. V. Quine*. D. Reidel, Dordrecht.

CHURCH, K., & HANKS, P. 1989. Word association norms, mutual information and lexicography. *Pages 76–83 of: Proceedings of the 27th ACL.*

CLARKSON, P., & ROBINSON, A.J. 1997. Language model adaptation using mixtures and an exponentially decaying cache. *In: Proceedings of ICASSP'97.*

CLARKSON, P., & ROSENFELD, R. 1997. Statistical language modeling using the CMU-Cambridge Toolkit. *Pages 2707–2710 of: Proceedings of Eurospeech'97.*

COCCARO, N., & JURAFSKY, D. 1998. Towards better integration of semantic predictors in statistical language modeling. *In: Proceedings of ICSLP'98.*

COLLINS, A. M., & LOFTUS, E. F. 1975. A spreading-activation theory of semantic processing. *Psychological Review*, **82**(1), 407–428.

COPESTAKE, A. 1997. Augmented and alternative NLP techniques for augmentive and alternative communication. *In: Proceedings of the ACL workshop on Natural Language Processing for Communication Aids.*

COVER, T. M., & KING, R. C. 1978. A convergent gambling estimate of the entropy of english. *IEEE Transactions on Information Theory*, **IT-24**(4), 413–421.

CRAMER, I., & FINTHAMMER, M. 2008. An evaluation procedure for Word Net based lexical chaining: Methods and issues. *In: Proceedings of the Global WordNet Conference.*

CRYSTAL, D. 1991. *A dictionary of linguistics and phonetics*. 3rd edn. Blackwell, Oxford.

CUNNINGHAM, H. 1999. A definition and short history of language engineering. *Natural Language Engineering*, **5**(1), 1–16.

DAGAN, I., LEE, L., & PEREIRA, F. 1999. Similarity-based models of word cooccurrence probabilities. *Machine Learning*, **34**(1-3), 43–69.

DARROCH, J., & RATCLIFF, D. 1972. Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics*, **43**(1), 1470–1478.

DE MORI, R., & FEDERICO, M. 1999. Language model adaptation. *In:* PONTING, K. (ed), *Computational Models of Speech Pattern Processing*. Springer Verlag.

DEERWESTER, S., DUMAIS, S., LANDAUER, T., FURNAS, G., & HARSHMAN, R. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society of Information Science*, **41**(6), 391–407.

DELIGNE, S., & BIMBOT, F. 1995. Language modeling by variable length sequences: theoretical formulation and evaluation of multigrams. *Pages 169–172 of: Proceedings of ICASSP'95*.

DELLA PIETRA, S., DELLA PIETRA, V., MERCER, R., & ROUKOS, S. 1992. Adaptive language model estimation using minimum discriminant estimation. *Pages 103–106 of: Proceedings of the workshop on speech and natural language at HLT'92*.

DEMASCO, P., & MCCOY, K. 1992. Generating text from compressed input: an intelligent interface for people with severe motor impairments. *Communications of the ACM*, **35**(5), 68–78.

DEMPSTER, A., LAIRD, N., & RUBIN, D. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, **39**(1), 1–38.

DENG, Y., & KHUDANPUR, S. 2003. Latent semantic information in maximum entropy language models for conversational speech recognition. *Pages 56–63 of: Proceedings of HLT-NAACL*.

DUMAIS, S. 1995. Latent Semantic Indexing (LSI): TREC-3 Report. *Pages 219–230 of:* HARMAN, D. (ed), *Proceedings of the 3rd Text REtrieval Conference (TREC-3)*, vol. 500-226. NIST Special Publication.

DUNNING, T. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational linguistics*, **19**(1), 61–74.

DVORAK, A., MERRICK, N., DEALEY, W., & FORD, G. 1936. *Typewriting behavior*. American Book Company.

EUROPEAN COMMISSION. 2007. *Situation of disabled people in the European Union: the European Action Plan 2008-2009*. Commission Staff Working Document.

FAZLY, A., & HIRST, G. 2003. Testing the efficacy of part-of-speech information in word completion. *In: Proc. of the 11th EACL, Workshop on Text Entry Methods*.

FEDERICO, M. 1996. Bayesian estimation methods for n-gram language model adaptation. *Pages 240–243 of: Proceedings of ICSLP'96*.

FELLBAUM, C. (ed). 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

FILLMORE, C. J. 1968. The case for case. *Pages 1–88 of:* BACH, E., & HARMS, R. (eds), *Universals in linguistic theory*. New York: Holt, Rinehart and Winston.

FOLTZ, P., KINTSCH, W., & LANDAUER, T. 1998. The measurement of textual coherence with Latent Semantic Analysis. *Discourse Processes*, **25**(2&3), 285–307.

FOULDS, R. 1973. *Tufts Interactive Communicator: A communicative assist for the non-verbal severely handicapped*. Tech. rept. Tufts-New England Medical Center, Boston, MA.

GARAY-VITORIA, N., & ABASCAL, J. 2006. Text prediction systems: a survey. *Universal Access in the Information Society*, **4**(3), 188–203.

GOLDSMITH, J., & REUTTER, T. 1998. Automatic collection and analysis of German compounds. *Pages 61–69 of:* BUSA, F. ET AL. (ed), *The Computational Treatment of Nominals*. Montreal, Canada: Université de Montreal.

GOODMAN, J. 2000. Putting it all together: language model combination. *Pages 1647–1650 of: Proceedings of ICASSP'00*.

GOODMAN, J. 2001. *A bit of progress in language modeling, ext. version*. Tech. rept. Microsoft Research MSR-TR-2001-72.

GREFENSTETTE, G. 1994. *Corpus-derived first, second and third order affinities*.

GUENTHNER, F., KRÜGER-THIELMANN, K., PASERO, R., & SABATIER, P. 1992. Communication aids for ALS patients. *Pages 303–307 of: Proceedings of the 3rd ICCHP*.

GUENTHNER, F., LANGER, S., KRÜGER-THIELMANN, K., PASERO, R., RICHARDET, N., & SABATIER, P. 1993. *KOMBE. Communication Aids for the Handicapped*. Tech. rept. 92-55. Centrum für Informations- und Sprachverarbeitung (CIS), Ludwig-Maximilians-Universität München.

HALLIDAY, M. A. K. 1978. *Language as social semiotic: The social interpretation of language and meaning*. Baltimore: University Park Press.

HAMP, B., & FELDWEG, H. 1997. GermaNet - a lexical-semantic net for German. *In: Proceedings of the ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*.

HAWKINS, F. 1993. *Human factors in flight*. 2nd edn. Ashgate Publishing.

HECKATHORNE, C., & CHILDRESS, D. 1983. Applying anticipatory text selection in a writing aid for people with severe motor impairment. *IEEE Micro*, **3**(3), 17–23.

HIRST, G., & ST-ONGE, D. 1998. Lexical chains as representations of context for the detection and correction of malapropisms. *In:* FELLBAUM, C. (ed), *WordNet: An electronic lexical database*. Cambridge, MA: The MIT Press.

HOLLAND, J. 1992. *Adaptation in natural and artificial systems*. 3rd edn. The MIT Press.

HORN, G. 1983. *Lexical-functional grammar*. Berlin: Mouton de Gruyter.

HRUSKA, S., KISKOWSKI, M., LEFEAUX, J., MCCLEARY, K., NGOUYASSA, D., & SMITH, B. 2000. Optimizing language models for speech recognition. *In: Proceedings of the IMA Summer Workshop*.

HUFFMAN, D. A. 1952. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, **40**(9), 1098–1101.

HUNNICUTT, S. 1981. A multi-language, portable text-to-speech system for the disabled. *STL - quarterly progress and status report*, **22**(2-3), 8–16. KTH, Stockholm, Sweden.

HUNNICUTT, S. 1986. Lexical prediction for a text-to-speech system. *In:* HJELMQUIST, E., & NILSSON, L.-G. (eds), *Communication and Handicap: Aspects of Psychological Compensation and Technical Aids*. Elsevier Science Publishers.

HUNNICUTT, S. 1989. Using syntactic and semantic information in a word prediction aid. *Pages 1191–1193 of: Proceedings of Eurospeech'89*.

JAYNES, E. 1957. Information theory and statistical mechanics. *Physics reviews*, **106**(1), 620–630.

JELINEK, F. 1990. Self-organized language models for speech recognition. *Pages 450–506 of:* WAIBEL, A., & LEE, K.-F. (eds), *Readings in speech recognition*. Morgan Kaufman Publishers.

JELINEK, F., & MERCER, R. 1980. Interpolated estimation of Markov source parameters from sparse data. *Pages 381–397 of: Proceedings of the workshop on pattern recognition in practice*.

JELINEK, F., MERIALDO, B, ROUKOS, S., & STRAUSS, M. 1991. A dynamic language model for speech recognition. *Pages 293–295 of: Proceedings of the NLT-Workshop on Speech and Natural Language*.

JOSHI, A. K., LEVY, L. S., & TAKAHASHI, M. 1975. Tree adjunct grammars. *Journal of Computer and System Sciences*, **10**(1), 136–163.

JURAFSKY, D., & MARTIN, J. 2000. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Prentice Hall.

KATZ, S. 1987. Estimation of probabilities from sparse data for the language model component of a speech recogniser. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **35**(3), 400–401.

KHUDANPUR, S., & WU, J. 2000. Maximum entropy techniques for exploiting syntactic, semantic and collocational dependencies in language modeling. *Computer speech and language*, **14**(1), 355–372.

KILGARRIFF, A. 2007. Googleology is bad science. *Computational linguistics*, **33**(1), 147–151.

KINTSCH, W. 2000. Metaphor comprehension: A computational theory. *Psychonomic bulletin and review*, **7**(1), 257–266.

KLAKOW, D. 1998. Log-linear interpolation of language models. *Pages 1695–1699 of: Proceedings of the 5th ICSLP*.

KNESER, R., & NEY, H. 1995. Improved backing-off for m-gram language modeling. *Pages 181–184 of: Proceedings of ICASSP'95*.

KOEHN, P., & KNIGHT, K. 2003. Empirical methods for compound splitting. *In: Proceedings of EACL'03*.

KOESTER, H., & LEVINE, P. 1994. Modelling the speed of text entry with a word prediction interface. *IEEE Transactions on Rehabilitation Engineering*, **2**(3), 177–187.

KONTOSTATHIS, A., & POTTENGER, W. 2003. A framework for understanding LSI performance. *In: Proceedings of ACM SIGIR Workshop on Mathematical Methods in Information Retrieval.*

KONTOSTATHIS, A., & POTTENGER, W. 2005. Identification of critical values in Latent Semantic Indexing. *In:* LIN, T. Y., OHSUGA, S., J., LIAU C., & S., TSUMOTO (eds), *Data mining: Foundations, methods, and applications.* Springer Verlag.

KUHN, R. 1988. Speech recognition and the frequency of recently used words: A modified Markov model for natural language. *Pages 348–350 of: Proceedings of COLING'88.*

KUHN, R., & DE MORI, R. 1990. A cache-based natural language model for speech reproduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **12**(6), 570–583.

KUKICH, K. 1992. Techniques for automatically correcting words in text. *ACM Computing Surveys*, **24**(4), 377–439.

KUPIEC, J. 1989. Probabilistic models of short and long distance word dependencies in running text. *Pages 290–295 of: Proceedings of the DARPA Workshop on Speech and Natural Language.*

LANCZOS, C. 1950. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *J. res. nat. bur. standards*, **45**(1), 255–282.

LANDAUER, T., & DUMAIS, S. 1997. A solution to Plato's problem: The Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*, **104**(1), 211–240.

LANDAUER, T., LAHAM, D., REHDER, B., & SCHREINER, M. E. 1997. How well can passage meaning be derived without using word order? A comparison of Latent Semantic Analysis and humans. *Pages 412–417 of:* SHAFTO, M. G., & LANGLEY, P. (eds), *Proceedings of the 19th annual meeting of the Cognitive Science Society.* Mawhwah, NJ: Erlbaum.

LANDAUER, T., FOLTZ, P. W., & LAHAM, D. 1998. Introduction to Latent Semantic Analysis. *Discourse Processes*, **25**(1), 259–284.

LANGER, S. 1998. Zur Morphologie und Semantik von Nominalkomposita. *In: Tagungsband der 4. Konferenz zur Verarbeitung natürlicher Sprache (KONVENS).*

LANGER, S., & HICKEY, M. 1998. Using semantic lexicons for full text message retrieval in a communication aid. *Natural Language Engineering*, **4**(1), 41–55.

LARSON, M., WILLETT, D., KÖHLER, J., & RIGOLL, G. 2000. Compound splitting and lexical unit recombination for improved performance of a speech recognition system for German parliamentary speeches. *In: Proceedings of the 6th ICSLP.*

LE PÉVÉDIC, BRIGITTE. 1997. *Prédiction morphosyntaxique évolutive dans un système d'aide à la saisie de textes pour des personnes handicapées physiques.* Ph.D. thesis, Université de Nantes.

LEACOCK, C., & CHODOROW, M. 1998. Combining local context and wordnet similarity for word sense identification. *In:* FELLBAUM, C. (ed), *WordNet: An electronic lexical database.* Cambridge, MA: The MIT Press.

LESHER, G., & RINKUS, G.J. 2001. Domain-specific word prediction for augmentive communications. *In: Proceedings of the RESNA'02 Annual Conference*.

LESHER, G., MOULTON, B.J., & HIGGINBOTHAM, D.J. 1999. Effects of ngram order and training text size on word prediction. *Pages 52–54 of: Proceedings of the RESNA'99 Annual Conference*.

LESHER, G., MOULTON, B.J., HIGGINBOTHAM, D.J., & ALSOFROM, B. 2002. Limits of human word prediction performance. *In: Proceedings of CSUN'02*.

LI, J., & HIRST, G. 2005. Semantic knowledge in word completion. *In: Proceedings of ASSETS'05*.

LIDSTONE, G. J. 1920. A note on the general case of the Bayes-Laplace formula for inductive or a posteriori probabilities. *Transactions of the faculty of actuaries*, **8**(1), 182–192.

LIN, D. 1998. An information-theoretic definition of similarity. *In: Proceedings of the 15th int. conference on machine learning*.

LUND, K., & BURGESS, C. 1996. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behaviour research methods, instruments and computers*, **28**(2), 203–208.

MAHAJAN, M., BEEFERMAN, D., & HUANG, X. 1999. Improved topic-dependent language modeling using information retrieval techniques. *In: Proceedings of ICASSP'99*.

MANNING, C., & SCHÜTZE, H. 1999. *Foundations of statistical natural language processing*. MIT Press, Cambridge, MA.

MARKOV, A. A. 1913. Essai d'une recherche statistique sur le texte du roman "Eugene Onegin" illustrant la liaison des épreuves en chaîne. *Bulletin de lAcadémie Impériale des Sciences de St.-Pétersbourg*, **7**(3), 153–162.

MATIASEK, J., & BARONI, M. 2003. Exploiting long distance collocational relations in predictive typing. *In: Proceedings of the EACL'03 Workshop on Language Modeling for Text Entry Methods*.

MAUREL, D., & LE PÉVÉDIC, B. 2001. The syntactic prediction with token automata: application to HandiAS system. *Theoretical computer science archive*, **267**(1-2), 121–129.

MAUREL, D., B., FOURCHE, & BRIFFAULT, S. 2000. HandiAS: Aider la communication en facilitant la saisie rapide de textes. *In: Actes du Colloque Handicap'00*.

MCCAULEY, L. 2004. Using Latent Semantic Analysis to aid speech recognition and understanding. *In: Proceedings of the international conference on wireless networks*.

MCCOY, K., & DEMASCO, P. 1995. Some applications of natural language processing to the field of augmentative and alternative communication. *Pages 97–112 of: Proceedings of the IJCAI'95 Workshop on Developing AI Applications for Disabled People*.

MCQUEEN, M., & MANN, S. 2000. A language model based optical character recogniser (OCR) for reading incidental text. *In: Proceedings of NACCQ'00*.

MILLER, G. 1956. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review*, **63**(1), 81–97.

MOORE, R., APPELT, D., DOWDING, J., GAWRON, J., & MORAN, D. 1995. Combining linguistic and statistical knowledge sources in natural-language processing for ATIS. *In: Proceedings of the Spoken Language Systems Technology Workshop*.

MORRIS, C., NEWELL, A., BOOTH, L., RICKETTS, I., & ARNOTT, J. 1992. Syntax PAL: A system to improve the written syntax of language-impaired users. *Assistive Technology*, **4**(2), 51–59.

MOULTON, B., LESHER, G., & HIGGINBOTHAM, D. 1999. A system for automatic abbreviation expansion. *Pages 55–57 of: Proceedings of the RESNA'99 Annual Conference*.

NEWELL, A. 1974. The talking brooch: A communication aid. *In:* COPELAND, K. (ed), *Aids for the severely handicapped*. London, UK: Sector Publishing.

NEWELL, A., LANGER, S., & HICKEY, M. 1998. The rôle of natural language processing in alternative and augmentative communication. *Natural Language Processing*, **4**(1), 1–16.

NEY, H., & ESSEN, U. 1991. On smoothing techniques for bigram-based natural language modelling. *Pages 825–829 of: Proceedings of ICASSP'91*, vol. 2.

NEY, H., ESSEN, U., & KNESER, R. 1994. On structuring probabilistic dependencies in stochastic language modeling. *Computer Speech and Language*, **8**(1), 1–38.

NICOLAS, P., LETELLIER-ZARSHENAS, S., SCHADLE, I., ANTOINE, J.-Y., & CAELEN, J. 2002. Towards a large corpus of spoken dialogue in French that will be freely available: the Parole Publique project. *Pages 649–655 of: Proceedings of LREC'02*.

NIESLER, T., & WOODLAND, P. 1996. A variable-length category-based n-gram language model. *In: Proceedings of ICASSP'96*.

ORDELMAN, R., VAN HESSEN, A., & DE JONG, F. 2003. Compound decomposition in Dutch large vocabulary speech recognition. *In: Proceedings of EuroSpeech'03*.

PETERS, J., & KLAKOW, D. 1999. Compact maximum entropy language models. *In: Proceedings of the IEEE workshop on automatic speech recognition and understanding*.

PICKERING, M., & GARROD, S. 2004. Toward a mechanistic psychology of dialogue. *Behavioral and Brain Sciences*, **27**(1), 169–225.

PINKER, S. 1994. *The language instinct: How the mind creates language*. New York: Harper Collins.

PLAUT, D. C. 1995. Semantic and associative priming in a distributed attractor network. *Pages 37–42 of: Proceedings of the 17th Annual Conference of the Cognitive Science Society*.

POLLARD, C., & SAG, I. 1994. *Head-driven phrase structure grammar*. Chicago: University of Chicago Press.

POPOVIĆ, M., STEIN, D., & NEY, H. 2006. Statistical machine translation of German compound words. *Pages 616–624 of: Proceedings of FinTAL'06*. Turku, Finland: LNCS, Springer Verlag.

PUCHER, M. 2007. *Semantic similarity in automatic speech recognition for meetings*. Ph.D. thesis, Technische Universität Graz, Austria.

RAPP, R. 2002. The computation of word associations: Comparing syntagmatic and paradigmatic approaches. *In: Proceedings of COLING'02.*

RAPP, R. 2003. Word sense discovery based on sense descriptor dissimilarity. *In: Proceedings of the Machine Translation Summit IX.*

RESNIK, P. 1995. Using information content to evaluate semantic similarity in a taxonomy. *In: Proceedings of IJCAI'95.*

ROARK, B. 2001. Probabilistic top-down parsing and language modeling. *Computational Linguistics*, **27**(2), 249–276.

ROSENFELD, R. 1994. *Adaptive statistical language modelling: a maximum entropy approach.* Ph.D. thesis, Carnegie Mellon University, Pittsburgh, PA.

ROSENFELD, R. 1996. A maximum entropy approach to adaptive statistical language modeling. *Computer Speech and Language*, **10**(1), 187–228.

ROSENFELD, R. 2000. Two decades of statistical language modeling: Where do we go from here? *Proceedings of the IEEE*, **88**(8), 1270–1278.

ROTTMANN, H. 2004. *Warum ausgerechnet .Net?* Vieweg.

RUPPENHOFER, J., ELLSWORTH, M., PETRUCK, M., JOHNSON, C., & SCHEFFCZYK, J. 2006. *FrameNet II: Extended Theory and Practice.* http://framenet.icsi.berkeley.edu/.

SALTON, G., & MCGILL, M. 1983. *Introduction to modern information retrieval.* New York: McGraw-Hill.

SCHADLE, I. 2003. *Sibylle: Système d'aide à la communication pour les personnes handicapées.* Ph.D. thesis, Université de Bretagne Sud.

SCHADLE, I., LE PÉVÉDIC, B., ANTOINE, J.-Y., & POIRIER, F. 2001. SibyLettre - système de prédiction de lettre pour l'aide à la saisie de texte. *In: Actes de TALN'01.*

SCHADLE, I., ANTOINE, J.-Y., LE PÉVÉDIC, B., & POIRIER, F. 2004. SibyMot: Modélisation stochastique du langage intégrant la notion de chunks. *In: Actes de TALN'04.*

SCHÜTZE, H. 1998. Automatic word sense discrimination. *Computational Linguistics*, **24**(1), 97–123.

SEYMORE, K., & ROSENFELD, R. 1996. Scalable backoff language models. *Pages 232–235 of:* BUNNELL, H., & IDSARDI, W. (eds), *Proceedings of ICSLP'96.*

SEYMORE, K., & ROSENFELD, R. 1997. *Large-scale topic detection and language model adaptation.* Tech. rept. CMU-CS-97-152. School of Computer Science, Carnegie Mellon University, Pittsburgh.

SHANNON, C. E. 1948. A mathematical theory of communication. *Bell systems technical journal*, **27**(623-656), 379–423.

SHANNON, C. E. 1950. The redundancy of English. *Pages 248–272 of: Cybernetics. the MACY conferences*, vol. 1. Diaphanes Verlag Berlin/Zürich.

SHANNON, C.E. 1951. Prediction and entropy of printed English. *Bell systems technical journal*, **30**(1), 50–64.

SITBON, L., BELLOT, P., & BLACHE, P. 2007. Traitements phrastiques phonétiques pour la réécriture de phrases dysorthographiées. *In: Actes de TALN'07*.

SMAÏLI, K., LAVECCHIA, C., & HATON, J.-P. 2006. Linguistic features modeling based on partial new cache. *In: Proceedings of LREC'06*.

SOUKOREFF, R. W., & MACKENZIE, I. S. 2003. Metrics for text entry research: an evaluation of MSD and KSPC, and a new unified error metric. *Pages 113–120 of: Proceedings of the sigchi conference on human factors in computing systems*.

SPÄRCK-JONES, K., & GALLIERS, J. 1996. *Evaluating natural language processing systems - an analysis and review*. Berlin: Springer.

STOLCKE, A. 1998. Entropy-based pruning of backoff language models. *Pages 270–274 of: Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*.

STOLCKE, A. 2002. SRILM - an extensible language modeling toolkit. *In: Proceedings of ICSLP'02*.

STORY, R. 1996. An explanation of the effectiveness of Latent Semantic Indexing by means of a Bayesian regression model. *Information Processing and Management*, **32**(3), 329–344.

STUM, G., & DEMASCO, P. 1992. Flexible abbreviation expansion. *In:* PRESPERIN, J. (ed), *Proceedings of RESNA'92*. Washington, D.C.: RESNA Press.

SWIFFIN, A., ARNOTT, J., PICKERING, J., & NEWELL, A. 1987a. Adaptive and predictive techniques in a communication prosthesis. *Augmentative and alternative communication*, **3**(4), 181–191.

SWIFFIN, A., ARNOTT, J., & NEWELL, A. 1987b. The use of syntax in a predictive communication aid for the handicapped. *Pages 124–126 of: Proceedings of the 10th annual conference on rehabilitation technology*.

TERRA, E., & CLARKE, C. L. 2003. Frequency estimates for statistical word similarity measures. *Pages 165–172 of: Proceedings of NAACL'03*.

TILLMANN, C., & NEY, H. 1997. Statistical language modeling and word triggers. *Pages 22–27 of: Proceedings of the Intl. workshop Speech and Computer*.

TODMAN, J., RANKIN, D., & FILE, P. 1999. The use of stored text in computer-aided conversation: A single-case experiment. *Journal of language and social psychology*, **18**(1), 287–309.

TRNKA, K., & MCCOY, K. 2007. Corpus studies in word prediction. *In: Proceedings of ASSETS'07*.

TRNKA, K., YARRINGTON, D., MCCOY, K., & PENNINGTON, C. 2006. Topic modeling in fringe word prediction for AAC. *In: Proceedings of the 2006 International Conference on Intelligent User Interfaces*.

TRNKA, K., YARRINGTON, D., MCCAW, J., MCCOY, K., & PENNINGTON, C. 2007. The effects of word prediction on communication rate for AAC. *Pages 173–176 of: Proceedings of NAACL'07*. Rochester, New York: Association for Computational Linguistics.

TROST, H., MATIASEK, J., & BARONI, M. 2005. The language component of the FASTY text prediction system. *Applied Artificial Intelligence*, **19**(8), 743–781.

WADE-STEIN, D., & KINTSCH, E. 2003. *Summary Street: Interactive computer support for writing*. Tech. rept. University of Colorado.

WANDMACHER, T. 2005. How semantic is Latent Semantic Analysis? *In: Proceedings of TALN/RECITAL'05*.

WANDMACHER, T., BÉCHET, N., BARHOUMI, Z., F., POIRIER, & ANTOINE, J.-Y. 2007. Système Sibylle d'aide à la communication pour personnes handicapées: modèle linguistique et interface utilisateur. *In: Actes de TALN'07*.

WANDMACHER, T., OVCHINNIKOVA, K., & ALEXANDROV, T. 2008. Does latent semantic analysis reflect human associations? *In: Proceedings of the Lexical Semantics workshop at ESSLLI'08*.

WARD, D., BLACKWELL, A., & MCKAY, D. 2000. Dasher: A data entry interface using continuous gestures and language models. *Pages 129–137 of: Proceedings of the 13th annual ACM Symposium on User Interface Software and Technology*.

WIDDOWS, D. 2004. *Geometry and meaning*. CSLI Publications.

WIEMER-HASTINGS, P. 1999. How latent is Latent Semantic Analysis? *Pages 932–941 of: Proceedings of IJCAI'99*.

WIEMER-HASTINGS, P., WIEMER-HASTINGS, K., & GRAESSER, A. 1999. Improving an intelligent tutor's comprehension of students with Latent Semantic Analysis. *In:* LAJOIE, S., & VIVET, M. (eds), *Artificial intelligence in education*. Amsterdam: IOS Press.

WITTEN, I. H., & BELL, T. C. 1991. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, **37**(4), 1085–1094.

WOODLAND, P., ODELL, J., HAIN, T., MOORE, G., NIELSER, T., TUERK, A., & WHITTAKER, E. 1998. Improvements in accuracy and speed in the HTK Broadcast News Transcription System. *In: Proceedings of Eurospeech'98*.

WU, Z., & PALMER, M. 1994. Verb semantics and lexical selection. *In: Proceedings of the 32nd ACL*.

XU, P., CHELBA, C., & JELINEK, F. 2002. A study on richer syntactic dependencies for structured language modeling. *Pages 191–198 of: Proceedings of the 40th ACL*.

ZAGLER, W. L., BECK, C., & SEISENBACHER, G. 2003. FASTY - faster and easier text generation for disabled people. *In: Proceedings of AAATE'03*.

Related publications

WANDMACHER, T., & ANTOINE, J.-Y. 2005. Exploiting the context for automatic text prediction. *In: Proceedings of the 9. annual meeting on health, science and technology.*

WANDMACHER, T., & ANTOINE, J.-Y. 2006a. Adaptation de modèles de langage à l'utilisateur et au registre de langage : expérimentations dans le domaine de l'aide au handicap. *In: Actes de TALN'06.*

WANDMACHER, T., & ANTOINE, J.-Y. 2006b. Training language models without appropriate resources: experiments with an aac system for disabled people. *In: Proceedings of LREC'06.*

WANDMACHER, T., & ANTOINE, J.-Y. 2007a. How to enter text without typing? - Usage aspects of the Sibylle AAC system. *In: Proceedings of HUMAN'07.*

WANDMACHER, T., & ANTOINE, J.-Y. 2007b. Methods to integrate a language model with semantic information for a word prediction component. *In: Proceedings of EMNLP'07.*

WANDMACHER, T., & ANTOINE, J.-Y. 2007c. Modèle adaptatif pour la prédiction des mots". *Traitement Automatique des Langues*, **48**(2), 71–95.

WANDMACHER, T., ANTOINE, J.-Y., SCHADLE, I., & KRÜGER-THIELMANN, K. 2006. Sibylle AAC system: Exploiting syntax and semantics for word prediction. *In: Proceedings of the 12th Biennal ISAAC.*

WANDMACHER, T., ANTOINE, J.-Y., & POIRIER, F. 2007a. SIBYLLE: A system for Alternative Communication Adapting to the Context and its User. *In: Proceedings of ACM-ASSETS'07.*

WANDMACHER, T., BÉCHET, N., BARHOUMI, Z., F., POIRIER, & ANTOINE, J.-Y. 2007b. Système Sibylle d'aide à la communication pour personnes handicapées: modèle linguistique et interface utilisateur. *In: Actes de TALN'07.*

WANDMACHER, T., ANTOINE, J.-Y., POIRIER, F., & DÉPARTE, J.-P. 2008. Sibylle: An assistive communication adapting to the context and its user. *ACM Transactions on Accessible Computing*, **1**(1), 6:1 – 6:30.

List of Tables

List of Figures

Tonio Wandmacher

Adaptive word prediction and its application in an assistive communication system

Résumé

Ce travail étudie les capacités de méthodes d'adaptation pour la prédiction de mots. Le premier groupe de méthodes traite de l'adaptation aux préférences lexicales et syntaxiques de l'utilisateur d'un système de communication assistée. Au sein de ce groupe de méthodes, nous avons étudié le modèle cache, le lexique auto-adaptatif et le modèle d'utilisateur dynamique (MUD), intégrant toute saisie de l'utilisateur. Le deuxième groupe de méthodes rassemble des approches qui ont pour objectif d'exploiter le contexte sémantique. Dans ce contexte, nous avons en particulier étudié l'Analyse Sémantique Latente (LSA), un modèle vectoriel qui se base sur les propriétés distributionnelles. Dans la dernière partie nous présentons un système d'aide à la communication, dans lequel nous avons implémenté les méthodes d'adaptation. Après une description de l'interface utilisateur nous avons exposé quelques expériences réalisées avec ce système, qui est utilisé dans un centre de rééducation fonctionnelle.

Mots-clés : prédiction de mots, modélisation du langage, modélisation de l'utilisateur, communication assistée, Analyse Sémantique Latente

Résumé en anglais

This thesis investigates the capacities of adaptive methods for word prediction. We present and evaluate several adaptation methods: First, we consider strategies enabling to adapt to the lexical and syntactic preferences of the user of an AAC system. Here we investigate the cache model, an auto-adaptive user lexicon and the dynamic user model (DUM), which integrates every input of the user. The second class of methods aims to adapt to the semantic context. Here we focus in particular on Latent Semantic Analysis (LSA), a vectorial model establishing semantic similarity from distributional properties. In the last part an assistive communication system is presented that implements the previously investigated adaptation methods. After a description of the user interface we report results from the application of this system in a rehabilitation center.

Keywords: word prediction, language modeling, user modeling, augmentative and alternative communication, Latent Semantic Analysis