FALKO - Ein fehlerannotiertes Lernerkorpus des Deutschen

Peter Siemen, Anke Lüdeling, Frank Henrik Müller
siemen@informatik.hu-berlin.de, anke.luedeling@hu-berlin.de
Korpuslinguistik
Institut für deutsche Sprache und Linguistik
Humboldt-Universität zu Berlin

Abstract

Der vorliegende Artikel gibt einen Überblick über die Suche in einem fehlerannotierten Lernerkorpus. "Fehlerannotiert" bedeutet, dass die Fehler der Fremdsprachenlerner im Korpus annotiert sind. Das Falko-Korpus verfügt über eine nicht-hierarchisch strukturierte *multi-layer*–Architektur, die um beliebige Annotationsebenen erweiterbar ist. Neben den von den Lernern gemachten sprachlichen Fehlern sind im Korpus syntaktische Strukturen annotiert. Wir zeigen auf, welche Problematik die Suche in solch tief annotierten Daten mit sich bringt und wie im Falko-Korpus damit verfahren wird. In einer technischen Darstellung wird der derzeitige Entwicklungsstand bezüglich der Suche im Korpus skizziert, und es werden geplante Wachstumsschritte vorgestellt.

1 Überblick

In Abschnitt 2 werden die Eckdaten des Falko-Korpus genannt und die Grundannahmen und Fragestellungen bezüglich der vorgenommenen Fehlerannotation erläutert. Abschnitt 3 dient der Vorstellung des gewählten Datenmodells zur Erfassung der Korpusdaten bzw. zur Annotation derselben. In Abschnitt 4 werden die gewählten Werkzeuge und die Datenstruktur zur Korpussuche vorgestellt. In Abschnitt 5 folgt eine Darstellung des Webinterface und im letzten Abschnitt 6 wird ein Ausblick auf geplante Erweiterungen des Falko-Korpus gegeben.

2 Das Falko-Korpus

Lernerkorpora spielen in der Fremdspracherwerbsforschung und der Sprachvermittlung eine zunehmend größere Rolle. Während es für das Englische bereits größere Lernerkorpora gibt (siehe zum Beispiel Nesselhauf (2004) und Römer (2006)), gibt es für das Deutsche als Fremdsprache bisher kaum frei zugängliche systematisch erstellte Lernerkorpora (Lüdeling, erscheint a). Falko wird hier eine Lücke schließen.

Unser Projekt untersucht die Bereiche Design, Architektur, Suche und Auswertung von Lernerkorpora exemplarisch anhand des Falko-Korpus. Bestehende und bereits in der Lehre oder in der Forschung eingesetzte Komponenten sollen verwendet werden und, wenn nötig, für unsere Zwecke angepasst werden. Die Entwicklung eines lauffähigen Prototypen wird helfen, Grenzen und Möglichkeiten der technischen Umsetzung kennen zu lernen und das Falko-Korpus als öffentlich zugängliches fehlerannotiertes Lernerkorpus des Deutschen bekannt zu machen.

2.1 Korpusdesign

Falko enthält Texte von fortgeschrittenen Lernern des Deutschen als Fremdsprache. Es besteht aus mehreren Subkorpora, die unterschiedlich tief annotiert sind. Das zur Zeit am besten annotierte Subkorpus von Falko ist eine Sammlung von Zusammenfassungen von linguistischen und literaturwissenschaftlichen Fachtexten, die von Studierenden der Freien Universität Berlin erstellt wurden (siehe auch Lüdeling et al. (2005) und Lüdeling (erscheint a)). Die folgenden Ausführungen werden mit Beispielen aus diesem Subkorpus illustriert. Alle Subkorpora sind frei im Netz[1] verfügbar.

2.2 Fehlerannotation

Bestimmte Auswertungen auf Lernerkorpora setzen eine Fehlerannotatation voraus, also eine Kategorisierung der Fehler oder Abweichungen in den Lernertexten. Es ist aus verschiedenen Gründen schwierig, Fehler zu definieren und Fehlertagsets zu entwickeln (zum Status des Fehlers siehe Cherubim (1980) und Lennon (1991), zum Design von Fehlertagsets siehe Granger (2002) und Tono (2003)).

[1] http://www2.hu-berlin.de/korpling/projekte/falko/

In bisherigen Lernerkorpora wurden Fehler (ähnlich wie Wortarten oder Lemmata in großen Zeitungskorpora) 'flach' annotiert; also entweder in einem Tabellenformat oder in einem XML-Baumformat. In Lüdeling et al. (2005) und Lüdeling (erscheint b) wird argumentiert, dass es sinnvoll ist, unterschiedliche Annotationskonventionen und Granularitätsebenen für verschiedene Fragestellungen zuzulassen.

Ein weiteres Problem der Fehlerannotation ist, dass jede Korrektur einer Lerneräußerung eine Hypothese darüber voraussetzt, was als korrekt angesehen wird. Oft kann man eine Lerneräußerung auf mehrere Weisen korrigieren; es existieren also unterschiedliche Zielhypothesen (Lüdeling, erscheint b). Eine Zielhypothese wird erstellt, wenn im Text eine nicht-zielsprachliche Formulierung bzw. ein nicht-zielsprachlicher Ausdruck vorkommt. Sie versucht, das zielsprachlich auszudrücken, was der Lerner vermutlich sagen wollte. Damit ist sie eine Interpretation. In Falko wird die Zielhypothese explizit angegeben. Jeder Fehler wird im Bezug auf diese Zielhypothese annotiert.

Ein Beispiel:

Originaltext: ...wenn es das konvenzionelle Wort „Lehrer"exiestiert...
Korrekturmöglichkeit 1: ...wenn das konventionelle Wort „Lehrer"existiert...
Korrekturmöglichkeit 2: ...wenn es das konventionelle Wort „Lehrer"gibt...
(Text 36, Position 124 ff)

Bei Zielhypothese 1 wird davon ausgegangen, dass der „Fehler" das unpersönliche Pronomen *es* ist, das hier nicht stehen darf. Bei Zielhypothese 2 hingegen wird davon ausgegangen, dass der „Fehler" in der Wortwahl liegt: statt *existiert* müsse dort *gibt* stehen. Die Architektur des Falko-Korpus ist darauf ausgelegt, abweichende Zielhypothesen aufzunehmen. Jede fehlerklassifizierende Annotationsebene referenziert explizit auf eine der Zielhypothesen.

Die Falko-Daten sind zusätzlich syntaktisch mit einer Felderstruktur annotiert - auch dies erfordert mehrere Annotationsebenen.

2.3 Anforderungen

Um dem Anspruch auf Erweiterbarkeit um beliebige Annotationsebenen gerecht zu werden, gilt es zunächst ein geeignetes Datenmodell zum Erfassen und zur Speicherung der Korpusdaten zu wählen. Falko nutzt eine Mehrebenenarchitektur, wie sie für multimodale Korpora entwickelt wurde (siehe zum Beispiel Carletta et al. (2003) und Wörner et al. (2006)). Aus den erfassten Daten muss außerdem eine Datenstruktur konstruiert werden, die eine effiziente Suche unter Berücksichtigung aller Annotationsebenen zulässt.

3 Datenmodell zum Erfassen der Falko-Korpusdaten

Das Falko-Korpus erfordert ein um beliebige und voneinander unabhängige Annotationsebenen erweiterbares Datenmodell.

3.1 EXMARaLDA

Wir verwenden für Falko das EXMARaLDA-Format.[2]

Im EXMARaLDA-Format definiert die Tokenfolge des zu annotierenden Textes eine sogenannte *timeline*. Jedes Token entspricht einem eindeutigen *event*, das durch seine Start- und End-Position auf der *timeline* beschrieben wird. Zu annotierende Token bzw. Tokenfolgen können dann als *event* bzw. *event*-Folge referenziert werden. Wir verwenden den EXMARaLDA-Partitureditor jexmaralda zur manuellen Annotation.

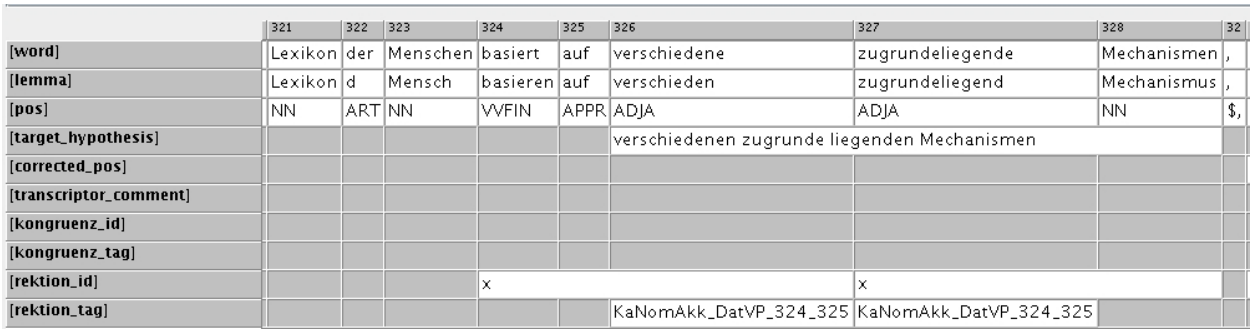
Es sind Annotationsebenen zu unterscheiden, deren Auszeichnungen ausschließlich einzelne Token referenzieren und solche Annotationsebenen, deren Auszeichnungen auch Folgen von Wörtern oder ganze Texte klassifizieren. Im folgenden Abschnitt wird erläutert, wie konfligierende Hierarchien bei der Annotation von Folgen von Wörtern entstehen können und wie damit umgegangen wird.

3.2 Konfligierende Hierarchien

Bei umfangreicher Mehr-Ebenen-Annotation können auftretende konfligierende Hierarchien der unterschiedlichen Annotationsebenen nicht ausgeschlossen werden. Das folgende Beispiel (Abbildung 1) veranschaulicht eine solche Situation:

Das *event* „verschiedenen zugrunde liegenden Mechanismen" der *target-hypothesis*-Ebene beginnt an Position 326 und endet bei 329. Das *event* „x" der

[2] EXMARaLDA steht für „Extensible Markup Language for Discourse Annotation". Es ist ein System von Konzepten, Datenformaten und Werkzeugen für die computergestützte Transkription und Annotation gesprochener Sprache. EXMARaLDA wird in einem Teilprojekt des Sonderforschungsbereichs „Mehrsprachigkeit" (SFB 538) der Universität Hamburg als zentrale Architekturkomponente einer Datenbank „Mehrsprachigkeit" entwickelt. Alle Komponenten des EXMARaLDA-Systems sind frei verfügbar. (Schmidt, 2001)



	321	322	323	324	325	326	327	328	32
[word]	Lexikon	der	Menschen	basiert	auf	verschiedene	zugrundeliegende	Mechanismen	,
[lemma]	Lexikon	d	Mensch	basieren	auf	verschieden	zugrundeliegend	Mechanismus	,
[pos]	NN	ART	NN	VVFIN	APPR	ADJA	ADJA	NN	$,
[target_hypothesis]						verschiedenen zugrunde liegenden Mechanismen			
[corrected_pos]									
[transcriptor_comment]									
[kongruenz_id]									
[kongruenz_tag]									
[rektion_id]				x			x		
[rektion_tag]						KaNomAkk_DatVP_324_325	KaNomAkk_DatVP_324_325		

Abbildung 1: Beispiel zu konfligierenden Hierachien

rektion-id-Ebene beginnt an Position 324 und endet an Position 327.

Eine solche Struktur in einem flachen Annotationsschema mit Hilfe von SGML/XML-Tags abzubilden (Abbildung 2), würde die hierarchische Struktur verletzen. Das Dokument ließe sich nicht mehr mit Werkzeugen zur Verarbeitung von SGML/XML-Dokumenten parsen und der weitere Bearbeitungsaufwand stiege um ein Vielfaches. Einen Ausweg liefert die von uns gewählte *stand-off*-Annotation. Daher werden oft standoff-Architekturen verwendet. Das EXMARaLDA-Format ist jedoch kein *stand-off*-Format.[3] Text und Annotationen werden in demselben Dokument gespeichert. Es lassen sich jedoch im EXMARaLDA-Format genau wie in einer *stand-off*-Architektur beliebige Annotationsebenen zur Textebene abspeichern.

3.3 Anforderungen an die Architektur des Falko-Korpus

Das Falko-Korpus soll auf jedem Entwicklungsstand um beliebige Annotationsebenen erweiterbar sein. Zudem ist es wichtig, dass alle Annotationsebenen einzeln oder in Kombination durchsucht werden können. Die Softwarekomponenten sollen den spezifischen Bedürfnissen angepasst werden können, d.h. sie müssen als Quellcode verfügbar sein und von Grund auf selbst entwickelt werden. Wir wollen eine skalierbare (mehrbenutzerfähige) und robuste Architektur und Benutzerschnittstelle, die eine breite Anwenderakzeptanz findet.

Das EXMARaLDA-Format und der Partitureditor jexmaralda geben uns die Werkzeuge, das Falko-Korpus auf beliebigen Ebenen zu annotieren. Das Suchwerkzeug, das die EXMARaLDA-Gruppe als Prototyp (Zecke[4]) zur Verfügung stellt, genügt leider nicht unseren Anforderungen. Einerseits ist der Quellcode des Programms nicht öffentlich verfügbar. Daher sind wir nicht in der Lage, das Programm unseren Bedürfnissen anzupassen. Andererseits handelt es sich um eine *stand-alone*-Anwendung, d.h. sie ist nicht als Serveranwendung für den Mehrbenutzerbetrieb konzipiert.

4 Datenstruktur zur Suche im *multi-layer*-annotierten Korpus

Obwohl das verwendete *multi-layer*-EXMARaLDA-Datenformat XML-konform ist, können durch die hierarchische Entkoppelung der Annotationsebenen von der linearen Tokenfolge des annotierten Textes standarisierte XML-Anfragesprachen wie XQUERY bzw. XPATH nicht effektiv zur Suche eingesetzt werden. Wir werden im folgenden Abschnitt die von uns verwendete Datenstruktur zur Suche im Falko-Korpus erläutern.

4.1 Anforderungen an die Anfragesprache

Um das Falko-Korpus für umfangreiche korpuslinguistische Analysen verwenden zu können, muss die Anfragesprache zur Korpussuche einigen Anforderungen genügen. Jede Annotationsebene soll einzeln und/oder in Kombination spezifiziert werden können. Außerdem soll die Suche mit Hilfe von Regulären Ausdrücken unterstützt werden – ebenso natürlich die Suche nach Wortfolgen. Anfragen wie z.B. „Finde alle Vorkommen des Lemmas ’allgemein’ direkt gefolgt von einem weiteren Adjektiv innerhalb eines Rektionsfehlers, den

[3] Bei der *stand-off*- oder externen Annotation werden die Auszeichnungen separat von den eigentlichen Daten gespeichert. Das hat gegenüber *inline*-Annotation unter anderem den Vorteil, dass verschiedene Annotationshierarchien auf die gleichen Daten angewendet werden können.

[4] Die ZECKE (“Ziemlich einfaches Konkordanzwerkzeug für EXMARaLDA“) ist ein Prototyp für ein Suchwerkzeug auf EXMARaLDA-Daten.

```
...
Menschen/NN/Mensch
<rektion_id value="x">
  basiert/VVFIN/basieren
  auf/APPR/auf
  <target_hypothesis value="verschiedenen zugrundeliegenden Mechanischmen">
    verschiedene/ADJA/verschieden
</rektion_id>
    zugrundeliegende/ADJA/zugrundeliegend
    Mechanismen/NN/Mechanismus
  </target_hypothesis>
...
```

Abbildung 2: Verletzung der hierachischen Struktur bei der *inline*-Annotation

eine russische Muttersprachlerin, die im zweiten Jahr Deutsch lernt, in einem Nebensatz gemacht hat!“ sollen ermöglicht werden.

4.2 Verwendete Komponenten

Neben dem Suchwerkzeug ’Zecke’ der EXMARaLDA-Gruppe ist uns kein weiteres bekannt, das mit einem *multi-layer*-Datenmodell arbeitet. Alle von uns untersuchten Korpussuchwerkzeuge (Xaira[5] und NITE XML Toolkit (NXT)[6]) waren nicht in der Lage, *stand-off*-Formate mit konfligierenden Hierarchien zu verarbeiten.

Die Corpusworkbench (CWB) (Evert, 2005) ist ein Softwarepaket zur Indexierung und Volltextsuche von bzw. in großen Textsammlungen. Die CWB ist weit verbreitet, und es kann eine allgemeine Grundkenntnis über die in der CWB implementierte Anfragesprache CQP (Christ et al., 1999) vorausgesetzt werden. CQP unterstützt die Suche mittels Regulärer Ausdrücke und die Suche nach Tokenfolgen.

Um tief annotiertes Textmaterial mit der CWB zu indexieren, ist es erforderlich, die Daten in ein spezielles Format umzuwandeln. Wir haben einige Python-Skripte implementiert, mittels derer aus den annotierten Lernertexten im EXMARaLDA-Format das korrekte Eingabeformat für die CWB hergestellt wird.

Aufgrund der ungewöhnlich umfangreichen Annotationen des Falko-Korpus erreichen die CQP-Suchanfragen eine solche Komplexität, dass eine visuelle Anfrageunterstützung unumgänglich ist. Wir haben uns entschieden, für das Falko-Korpus und dessen Subkorpora die CWB durch korporaspezifische Java-Struts-Module und ein Java-Serverpages-(JSP)-Webinterface zu erweitern.

4.3 Datenstruktur zur Suche

Von der CWB zu indexierende Daten müssen als ASCII-Textdokument in der Form “ein-Token-pro-Zeile “ vorliegen. Tokenweise Annotationen (positionale Attribute) stehen in definierter Reihenfolge durch TABS getrennt in der Zeile des zu annotierenden Tokens. Wortfolgen werden durch umschließende XML-Tags zeilenweise, d.h. tokenweise annotiert (strukturelle Attribute). Die XML-Tags dürfen sich nicht gegenseitig überschneiden. (siehe Abbildung 3)

Um konfligierende Annotationen (wie in Abbildung 1) von Wortfolgen zu indexieren, können folglich keine XML-Tags verwendet werden. Wir kodieren daher alle strukturellen Attribute, die sich innerhalb eines Textes möglicherweise überschneiden werden, als positionale Attribute und indexieren diese fortlaufend mit 1 beginnend (siehe Abbildung 4). Da die CWB die Verwendung von Regulären Ausdrücken innerhalb von Spezifikationen von positionalen Attributen unterstützt, ermöglicht unsere Kodierung Abfragen wie: “Suche alle Vorkommen von Nomina innerhalb eines Matrixsatz-Vorfeldes!“ (CQP: *[pos=“NN“& matrixsatz-feld=“MS_VF#.*“]*). Mittels des Regulären Ausdrucks *matrixsatz-vorfeld=“MS_VF#.*“* werden alle Vorkommen von Nomina innerhalb eines Matrixsatz-Vorfeldes geliefert – unabhängig von der Position des Nomens innerhalb des Matrixsatz-Vorfeldes.

[5] an der Oxford University entwickeltes Suchwerkzeug für das British National Corpus

[6] NXT ist eine Sammlung von Programmierbibliotheken und Softwarewerkzeugen zur Repräsentation, Manipulation, Suche und Analyse von komplex annotierten Korpora und wurde von Jonathan Kilgour und der Language Technology Group in Edinburgh entwickelt.

```
<!-- A Thrilling Experience -->
<story num="4" title="A Thrilling Experience">
<p>
<s>
Tick    NN      tick
.       SENT    .
</s>
<s>
A       DT      a
clock   NN      clock
.       SENT    .
</s>
<s>
Tick    VB      tick
,       ,       ,
tick    VB      tick
.       SENT    .
</s>
</p>
...
</story>
```

Abbildung 3: Inputformat der CWB

Die Funktion des Indexes ist es, die einzelnen zusammengehörigen aber als positionale Attribute kodierten Annotationen über Wortfolgen zählen zu können ("Suche alle Matrixsatz-Vorfelder!", CQP: *matrixsatz-vorfeld="MS_VF#1"*). Auf diese Weise wird jedes annotierte Matrixsatz-Vorfeld nur genau einmal aufgeführt und nicht so oft, wie es Token lang ist. Des Weiteren liefert uns die Indexierung der Wortfolgen-Annotationen eine Möglichkeit, annotierte Wortfolgen ihrer Länge entsprechend zu suchen. (Bsp.: "Suche alle Matrixsatz-Vorfelder der Länge 2!"(CQP: *[matrixsatz-vorfeld=MS_VF#1] [matrixsatz-vorfeld=MS_VF#2] [matrixsatz-vorfeld!=MS_VF#3]*))

Lernerspezifische Annotationen beziehen sich immer auf ganze Texte und können als strukturelle Annotationen kodiert werden, da einzelne Texte sich nicht überschneiden können. Lernerspezifische Anfragen werden wie folgt formuliert. "Suche alle Vorkommen von Adverbien innerhalb eines Matrixsatz-Vorfeldes innerhalb aller bulgarischen Lernerinnentexte!" (CQP: *[pos="ADV" & matrixsatz-feld="MS_VF#.*"] :: match.learner_l1="bg" & match.learner_sex="w"*

5 Visuelle Anfrageunterstützung im Falko-Webinterface

Das Falko-Korpus wird über ein frei zugängliches Webinterface genutzt. Das Webinterface ist auf der Basis von Java-Serverpages (JSP) und Struts entwickelt. Der modulare Aufbau erlaubt es, für jedes Subkorpus eine spezifische Suchmaske bereit zu stellen. Die unterschiedlichen Korpus-Suchmasken enthalten zahlreiche Auswahlmenüs, die sich in lernerspezifische, syntax- und fehlerspezifische und Outputoptionen gliedern. Die zu formulierende CQP-Anfrage umfasst nur noch Angaben zur Wort-, Wortarten- und/oder Lemma-Ebene. Alle weiteren Anfragespezifikationen werden über grafische Menüs per Mausklick getroffen. Geübte Nutzer können den vollen Funktionsumfang von CQP nutzen, weniger geübte Nutzer können viele Einschränkungen über das Webinterface vornehmen.

Die CQP-Anfrage und die weiteren Spezifikationen werden an eine im Webinterface integrierte Java-Komponente gesendet, die eine entsprechende CQP-Anfrage erstellt. Die Java-Komponente ruft ihrerseits die CWB auf und parst das Ergebniss der Anfrage. Ein weiteres wichtiges Feature des Webinterfaces ist es, dass es entsprechend den getroffenen Output-Optionen die einzelnen Treffer benutzerfreundlich im Browser darstellt. Es können beliebig viele Annotationsebenen zusätzlich zu der Wort-Ebene angezeigt werden. Zu jedem Treffer lassen sich außerdem alle Metainformationen und/oder der gesamte Lernertext anzeigen.

6 Zusammenfassung und Ausblick

In den letzten Jahren sind für viele unterschiedliche Bereiche Mehrebenenkorpusarchitekturen entwickelt worden. Bisher gibt es aber kaum geeignete Suchmöglichkeiten auf diesen Korpora (auch wenn mit Zecke, Xaira und anderen im Moment mehrere Initiativen daran arbeiten). In diesem Papier haben wir anhand des Lernerkorpus Falko gezeigt, wie durch Rekombination und Anpassung von existierender Software, Kreativität und Zielstrebigkeit eine frei zugängliche Arbeitsumgebung für umfangreiche korpuslinguistische Analysen von nicht-hierarchisch strukturierten, *multi-layer*-fehlerannotierten Korpora entstehen kann.

Die Suche ist natürlich auch auf andere Korpora anwendbar. Wir arbeiten zur Zeit an einer Verbesserung des Interface und der Darstellungsmöglichkeiten. Zusätzlich sollen Funktionalitäten zur statistischen Analyse der Daten direkt in das Webinter-

face eingebunden werden.

Das Falko-Korpus selbst wird stetig um neues Textmaterial und neue Annotationsebenen erweitert.

References

Jean Carletta, Stefan Evert, Ulrich Heid, Jonathan Kilgour, Judy Robertson, and Holger Voormann. 2003. The NITE XML Toolkit: flexible annotation for multi-modal language data. *Behavior Research Methods, Instruments, and Computers*, 35(3):353–363.

Dieter Cherubim. 1980. *Fehlerlinguistik. Beiträge zum Problem der sprachlichen Abweichung*. Niemeyer, Tübingen.

Oliver Christ, Bruno Schulze, Anja Hoffmann, and Esther König, 1999. *The IMS Corpus Workbench, Corpus Query Processor (CQP)*. University of Stuttgart, Institute for Natural Language Processing (IMS), Stuttgart, Germany.

Stefan Evert, 2005. *The CQP Query Language Tutorial*. University of Stuttgart, Institute for Natural Language Processing (IMS), Stuttgart, Germany.

Sylviane Granger. 2002. A bird's-eye view of learner corpus research. In Sylviane Granger, Joseph Hung, and Stephanie Petch-Tyson, editors, *Computer Learner Corpora, Second Language Acquisition and Foreign Language Teaching*, pages 3–33. John Benjamins, Amsterdam.

Paul Lennon. 1991. Error and the very advanced learner. *International Review of Applied Linguistics*, 29(1):31–44.

Anke Lüdeling. erscheint, a. Das Zusammenspiel von qualitativen und quantitativen Methoden in der Korpuslinguistik. In Gisela Zifonun and Werner Kallmeyer, editors, *Jahrbuch des Instituts für deutsche Sprache*. de Gruyter, Berlin.

Anke Lüdeling. erscheint, b. Mehrdeutigkeiten und Kategorisierung: Probleme bei der Annotation von Lernerkorpora. In Patrick Grommes and Maik Walter, editors, *Fortgeschrittene Lernervarietäten*. Niemeyer, Tübingen.

Anke Lüdeling, Maik Walter, Emil Kroymann, and Peter Adolphs. 2005. Multi-level error annotation in learner corpora. In *Proceedings of the 2005 Corpus Linguistics Conference*, Birmingham.

Nadja Nesselhauf. 2004. Learner corpora and their potential in language teaching. In John Sinclair, editor, *How to Use Corpora in Language Teaching*, pages 125–152. John Benjamins, Amsterdam.

Ute Römer. 2006. Pedagogical Applications of Corpora: Some Reflections on the Current Scope and a Wish List for Future Developments. *Zeitschrift für Anglistik und Amerikanistik*, 54(2):122–134.

Thomas Schmidt, 2001. *EXMARaLDA 1.0, Dokumentation*. Universität Hamburg, Sonderforschungsbereich 538: Mehrsprachigkeit, Hamburg, Germany.

Yukio Tono. 2003. Learner corpora: design, development, and applications. In *Proceedings of the 2003 Corpus Linguistics Conference*, pages 800–809, Lancaster.

Kai Wörner, Andreas Witt, Georg Rehm, and Stefanie Dipper. 2006. Modelling Linguistic Data Structures. In *Proceedings of Extreme Markup Languages 2006*, Montreal.

```
<learner id="001" l1="bg" l2="en" l3="de" l4="ru" sex="w" birth="1882">
Der                 d                   ART         x#1         VF_MS#1
Mensch              Mensch              NN          x#2         VF_MS#2
kann                können              VMFIN       x#3         LSK_MS#1
klassifizieren      klassifizieren      VVINF       x#4         RSK_MS#1
,                   ,                   $,          x#5         NF_MS#1
zum                 zum                 APPRART     x#6         NF_MS#2
Beispiel            Beispiel            NN          x#7         NF_MS#3
ein                 ein                 ART         x#8         NF_MS#4
Haus                Haus                NN          x#9         NF_MS#5
und                 und                 KON         x#10        NF_MS#6
eine                ein                 ART         x#11        NF_MS#7
Scheune             Scheune             NN          x#12        NF_MS#8
....
</learner>
```

Abbildung 4: Indexierung des Falko-Korpus

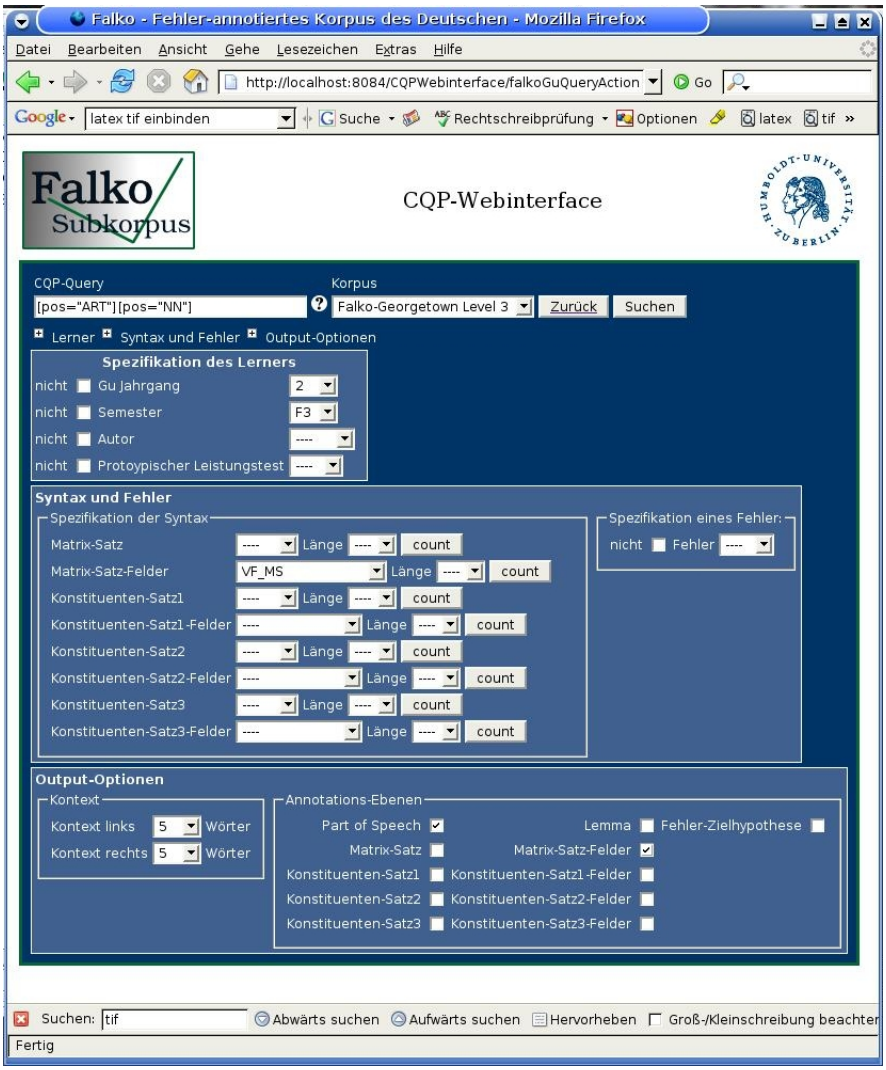


Abbildung 5: Falko-Suchmaske