Der Stand der Kunst in der Eigennamen-Erkennung

Mit einem Fokus auf Produktenamen-Erkennung

Lizentiatsarbeit der Philosophischen Fakultät
der Universität Zürich

Jeannette Roth

Zürich, Schweiz

Matrikelnummer 93-722-684

Angefertigt am
Institut für Computerlinguistik
der Universität Zürich
Prof. Dr. Michael Hess

Abgabe der Arbeit:

Dezember 2002

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1

Einleitung

Die Aufgabe, Eigennamen zu erkennen, wird häufig im Zusammenhang mit Informations-Extraktion (IE) angetroffen. In der IE wird versucht, aus Texten nicht-ambige Daten, die ein festgelegtes Format haben, zu extrahieren. Üblicherweise wird IE in verschiedene Teilaufgaben unterteilt und Eigennamen-Erkennung kann eine solche ausmachen oder wiederum Teil einer solchen sein. Aber auch viele andere computerlinguistische Anwendungen benötigen Eigennamen-Erkennung: Antwort-Extraktion, Information Retrieval, Textmining, Textzusammenfassung, maschinelle Übersetzung, Suchmaschinen usw.

Ziel dieser Arbeit ist es, den Stand der Kunst in der Eigennamen-Erkennung zu ermitteln. Da bei der Literatursuche sehr bald festgestellt wurde, dass die meisten Systeme, die Eigennamen erkennen können, sich auf die Personen-, Orts- und Organisationsnamen beschränken, wurde die Fragestellung ausgeweitet: Wie eignen sich die derzeitigen Ansätze, die ebenfalls wichtige Klasse von Produktenamen zu erkennen? Grund für die Wahl von Produktenamen ist eine frühere Seminararbeit, bei der ich ein Programm geschrieben habe, das Produktenamen erkennen sollte.[1] Da ich mit der Ausbeute (engl. *Recall*), die dieses Programm lieferte, nicht zufrieden war, wollte ich wissen, wie andere Systeme Produktenamen ermitteln.

Die Leitfragen dieser Arbeit lauten also:

- Welches sind in der Computerlinguistik die heute üblichen Ansätze oder Methoden, um in Texten Eigennamen automatisch zu erkennen? Welche Methoden eignen sich am besten?
- Wie eignen sich die derzeitigen Methoden zur Erkennung von Personen-, Orts- und Organisationsnamen, um Produktenamen zu erkennen?

Zur Beantwortung dieser Fragen stellten sich die Arbeiten, die anlässlich der im Jahre 1998 zum 7. Mal durchgeführten *Message Understanding Conference (MUC-7)* durchgeführt wurden, als äusserst nützlich heraus. Die MUC ist eine Konferenz, für die verschiedene Forschergruppen Systeme bauten, die gemäss genau definierten Anforderungen die Teilaufgaben lösen sollen, in die IE aufgeteilt wird. Darunter fiel auch besagte Teilaufgabe,

[1] Vgl. [Rot01].

Eigennamen zu erkennen. Die Funktionsweise der meisten Systeme wurde in Forschungsberichten, die über das Internet zugänglich sind, beschrieben. Von den Systemen selbst konnte ich, trotz Nachfragen bei den zuständigen Forschern, nur eines ausprobieren - eines, das im Internet öffentlich zugänglich ist.[2] Eine Evaluation durchzuführen - beispielsweise um die Frage der Eignung zur Produktenamenerkennung zu klären - fiel deshalb ausser Betracht.

Zwölf Forschergruppen - sowohl aus Firmen als auch aus Universitäten - bauten Systeme, die Eigennamen in englischen Texten erkennen, drei Gruppen Systeme für japanische und weitere drei für chinesische Texte (die Systeme für die Eigennamen-Erkennung in den asiatischen Sprachen wurden in einem separaten *Multilingual Entity Task (MET-2)* zusammengefasst). Auswahlkriterien für diese Arbeit waren einerseits die Verfügbarkeit der Systembeschreibungen (beispielsweise konnte über das System der Organisation *The MITRE Corporation* keine Beschreibung gefunden werden), andererseits das Ziel, ein möglichst breites Spektrum an angewandten Methoden abzudecken. So berücksichtige ich in meiner Arbeit die meisten der Systeme fürs Englische und auch eines fürs Japanische. Grund dafür ist die Methode, die dieses japanische System anwendet: Es wird mit Entscheidungs-Bäumen gearbeitet, einer wichtigen statistischen Verfahrensweise, die aber keines der anderen beschriebenen Systeme verwendet. Zusätzlich wird ein System vorgestellt, dessen Beschreibung als Klassiker in der Literatur über Eigennamen-Erkennung gelten darf (*Proper Name Facility von SPARSER*, vergleiche 5.1.1).

Die vorliegende Arbeit hat folgenden Aufbau: In Kapitel 2 wird versucht, den Begriff *Eigenname* gegen den damit eng verbundenen Begriff *Named Entity* abzugrenzen. Ziel dieser Einordnung ist auch festzulegen, was in dieser Arbeit unter *Eigenname* verstanden wird. Im gleichen Kapitel wird auch die MUC-7 etwas näher erklärt. Im folgenden Kapitel 3 wird der Frage nachgegangen, welchen Zweck Eigennamen-Erkennung erfüllt. Besonders von Interesse sind hier die Erklärungen zu den Fragen, was IE ist und welche Rolle dabei die Eigennamen-Erkennung spielt. In Kapitel 4 werden die Probleme eruiert, die mit der Eigennamen-Erkennung verbunden sind. Es folgt das umfangreiche Kapitel 5, in dem die für diese Arbeit ausgewählten Systeme eingehend beschrieben werden. Einige Systeme arbeiten mit statistischen Methoden, die bisweilen sehr komplex sein können. Die Ausführungen sind so gehalten, dass sie ein fortgeschrittener Computerlinguistik-Student auch dann verstehen kann, wenn er keine gründlichen Statistik-Kenntnisse besitzt. Dies wird dadurch erreicht, dass jeweils einführend die Grundlagen eines angewandten statistischen Konzepts erklärt und anschauliche Beispiele hinzugefügt werden. Im Schlusskapitel 6 werden die Systeme miteinander verglichen, beurteilt und das Fazit gezogen. Hier werden die Leitfragen der Arbeit beantwortet. Anschliessend ans Schlusskapitel folgt eine Zusammenfassung. Diese soll der Rekapitulation dienen, kann aber auch genutzt werden, um sich vor Beginn des Lesens einen Überblick über die Arbeit zu verschaffen. Im Anhang finden sich die Punkteverteilungen (engl. *scores*), die an der MUC-7 vorgenommen wurden, ebenso zwei gemäss den Punkteverteilungen erstellte Ranglisten (eine für die englischen, eine für die japanischen Systeme) und eine Liste der wichtigsten Abkürzungen.

[2] Es handelt sich hierbei um das im Unterkapitel 5.1.4 beschriebene System LT TTT.

Kapitel 2

Begriffsklärung: Eigenname und Named Entity

> *"Böse Wand", "Labyrinth" und "Nirwana", viele Orte tragen Namen, aber die Namen spielen eigentlich keine Rolle. Die Orte heissen ja nicht wirklich so, werden nur so genannt, damit man besser über sie reden kann. Und vielleicht auch, weil Namen Sicherheit geben, weil alles, was einen Namen trägt, zu einem Teil unserer Welt wird, die wir zu beherrschen glauben.*[1]

In einer Arbeit über Eigennamen-Erkennung sollte anfangs geklärt werden, was unter einem *Eigennamen* (engl. *Proper Name* oder *Proper Noun*[2]) zu verstehen ist. In der Literatur herrscht zwar ein gewisser Konsens darüber, welche Wörter zu den Eigennamen gezählt respektive nicht gezählt werden müssen. Allerdings sind die Grenzen von Autor zu Autorin verschieden. Meist wird auch gar nicht genau erklärt, was unter einem Eigennamen verstanden wird, es bleibt Aufgabe des Lesers zu interpretieren, was mit dem Begriff jeweils gemeint ist.

[1] Zitat aus *Eine Geschichte der Dunkelheit. Augenschein im Hölloch, der viertgrössten Karsthöhle der Welt.* Vom Schweizer Schriftsteller Peter Stamm. Gefunden in der SBB-Zeitschrift *Via*.

[2] Für den deutschen Begriff *Eigenname* werden im Englischen zwei Ausdrücke verwendet: *proper noun* und *proper name*. Dies ist die übliche Praxis und entspricht nicht der theoretischen Unterscheidung, die Quirk in der englischen Standardgrammatik macht: "We may therefore draw a distinction between a proper noun, which is a single word, and a name, which may or may not consist or more than one word."([Qui85], Seite 288).

2.1 Annäherung: Named Entities

Wo beginnen Namen[3], wo hören sie auf? Ist das Wort *Spaniel*, da er eine bestimmte Art von Hund bezeichnet, der Name eines Hundes? Oder ist erst die Bezeichnung für einen bestimmten Spaniel, beispielsweise *Cocker Spaniel* ein Name?[4] Cocker Spaniels können weiterhin in Englische und Amerikanische unterteilt werden - also darf erst beispielsweise der Ausdruck *Englischer Cocker Spaniel* als Eigenname betrachtet werden? Oder ist es erst dann zulässig, von einem Eigennamen zu sprechen, wenn damit ein ganz bestimmter, einzigartiger (engl. *unique*) Hund gemeint ist, also beispielsweise *Lassy*, *Bello*, *Rex* etc.?

Ein möglicher Ansatz, der Problematik näher zu kommen, ist sich zu fragen, was für eine Aufgabe Eigennamen haben. Mit einem Eigennamen will man eine so genannte *Named Entity (NE)*[5] bezeichnen. Eine *Enitität* (engl. *Entity*) ist ein "singuläres, von anderen Entitäten eindeutig unterscheidbares Exemplar aus einer Menge gleichartiger Personen, Gegenstände oder Begriffe aus der realen oder der Vorstellungswelt"[6]. Der Begriff *Named Entity*, der auf Deutsch keine Übersetzung hat, meint somit eine Entität (beispielsweise eine Person), der ein Name (beispielsweise *Irmgard Keller*) zugewiesen wurde. Diesen Überlegungen zufolge ist weder *Spaniel* noch *Englischer Cocker Spaniel* noch sonst irgendeine Bezeichnung für eine Rasse, Sorte oder dergleichen ein Eigenname, weil solche Begriffe nicht eine Entität, sondern eine Menge von Entitäten bezeichnen.

2.2 Duden: Eigennamen und Gattungsnamen

Dieses Verständnis von Eigenname teilt auch der Grammatik-Duden, der von den *Eigennamen* die *Gattungsnamen*[7] unterscheidet. Gemäss Duden werden mit *Eigennamen*

> [...] Lebewesen, Dinge u. a. bezeichnet, die so, wie sie sind, nur einmal vorkommen, z. B. bestimmte Menschen, Länder, Städte, Strassen, Berge, Gebirge, Flüsse, Seen, Meere, Fluren und andere Örtlichkeiten, Schiffe, Sterne, menschliche Einrichtungen und geistige Schöpfungen. Mit einem Eigennamen wird also etwas Bestimmtes, Einmaliges benannt; er ist in der Regel einzelnen Lebewesen oder Dingen zugeordnet und gestattet, diese zu identifizieren.[8]

Von der Gruppe der Eigennamen *ausgeschlossen* werden aber Tier-, Pflanzen-, Monats-, Wochentags-, Krankheits- und Verwandtschaftsbezeichnungen, also auch der in obiger Überlegung erwähnte *Englische Cocker Spaniel*. Solche Bezeichnungen gehören gemäss Duden der Gruppe der *Gattungsnamen* an. Mit Gattungsnamen werden alle Lebewesen oder Dinge einer Gattung benannt, wobei "unter Gattung eine Gruppe von Lebewesen oder

[3] Der Begriff *Name* ist synonym zum Begriff *Eigenname*.

[4] Es gibt auch Clumber-, Field-, Sussex- und Springer-Spaniels.

[5] Der Plural *Named Entities* wird mit *NEs* abgekürzt.

[6] http://www.accessarchive.com/FAQs/Modelling/German/Appendix_B_Woerterbuch.htm

[7] Statt *Gattungsnamen* können auch die Begriffe *Gattungsbezeichnungen* oder *Appelativa* verwendet werden.

[8] [Dud98], Seite 196.

Dingen" verstanden wird, "die wichtige Merkmale oder Eigenschaften gemeinsam haben. (Zum Beispiel zeichnet sich die Gattung Mensch unter anderem durch ihre 'Säugetierhaftigkeit' aus.)"[9] Dass die Unterscheidung zwischen Eigennamen und Gattungsnamen aber nicht immer einfach ist, zeigen folgende zwei Aussagen im Grammatik-Duden:

- "Bestimmte Substantive sind sowohl Eigenname wie Gattungsbezeichnung."[10] Als Beispiele werden semantisch ambige Wörter angeführt, die ursprünglich Eigennamen waren und im Laufe der Zeit ins allgemeinsprachliche Lexikon aufgenommen wurden: "Bayreuth ist das *Mekka* der Wagnerfreunde." oder "Dieser Lastkraftwagen ist ein *Diesel*." Dieselbe Problematik findet sich bei Personennamen, die ursprünglich Gattungsnamen waren, z. B. *Müller, Schmidt, Becker, Schreiner*.

- "Die Grenze zwischen Eigennamen und Gattungsbezeichnungen ist nicht immer leicht festzulegen."[11] Manifestiert hat sich diese Schwierigkeit vor allem in der deutschen Rechtschreibung, genauer in der Gross- oder Kleinschreibung von Adjektiven. Noch heute wird beispielsweise darüber diskutiert, ob man das *Schwarze Brett* oder das *schwarze Brett* schreiben solle. Gemäss der Neuregelung der deutschen Rechtschreibung handelt es sich nicht um einen Eigennamen, weil es nicht als eine Entität, sondern als Menge von Entitäten empfunden wird. Darum wird heute beim *schwarzen Brett* das Adjektiv - im Gegensatz zu früher[12] - kleingeschrieben. Weiterhin grossgeschrieben wird beispielsweise das Adjektiv in *das Schwarze Meer*, weil dieses Meer nur einmal existiert, das heisst eine Entität ist, und der Begriff deshalb als Eigenname betrachtet wird.

2.3 Grenzen fliessend

Wie eingangs erwähnt haben nicht alle Autorinnen und Autoren, die sich mit Eigennamen-Erkennung beschäftigen, dieselbe Auffassung, was sie zu den Eigennamen rechnen. So zählen beispielsweise Mani/McMillan[13] und Volk[14] Personen-, Orts-, Organisations- und Produktenamen zu den Eigennamen. Wacholder et al.[15] dagegen schliessen sich der englischen Standardgrammatik von Quirk[16] an, wo neben den bereits erwähnten Personen- und Ortsnamen auch Namen von Monaten (*September*), Wochentagen (*Thursday*), Festtagen (*Christmas*), Magazinen (*Vogue*) etc. als *proper nouns* bezeichnet werden. Dafür

[9] [Dud98], Seite 196.
[10] [Dud98], Seite 197.
[11] [Dud98], Seite 197.
[12] Die Neuregelung der deutschen Rechtschreibung ist seit 1. August 1998 in Kraft.
[13] Vgl. [Man96].
[14] Vgl. [Vol01]: Zwar handelt es sich hier um ein System zur Eigennamen-Erkennung im Deutschen; trotzdem kann es hier angeführt werden, da der Entscheid, welche Eigennamen ein System erkennen soll, nicht von der Sprache abhängt.
[15] Vgl. [Wac97].
[16] Vgl. [Qui85], Seite 288.

lassen Wacholder et al. die von Mani/McMillan und Volk gewählten Organisations- und Produktenamen ausser Acht.

Eine sehr breite Palette von NE-Kategorien wird in [Pai96] entworfen. Hier werden neun Kategorien von Named Entities aufgeführt, die in unterschiedlich viele Unterkategorien aufgeteilt werden (vergleiche Abbildung 2.1). In [Pai96] kommt man so auf 30 NE-Kategorien, wovon eine als Restgruppe dient: In die Kategorie *Misc.* werden alle NEs eingeteilt, die keiner anderen Kategorie zugeordnet werden können. Zu beachten ist bei dieser Einteilung, dass auch temporale Ausdrücke (*Date* und *Time*) zu den Eigennamen gerechnet werden.

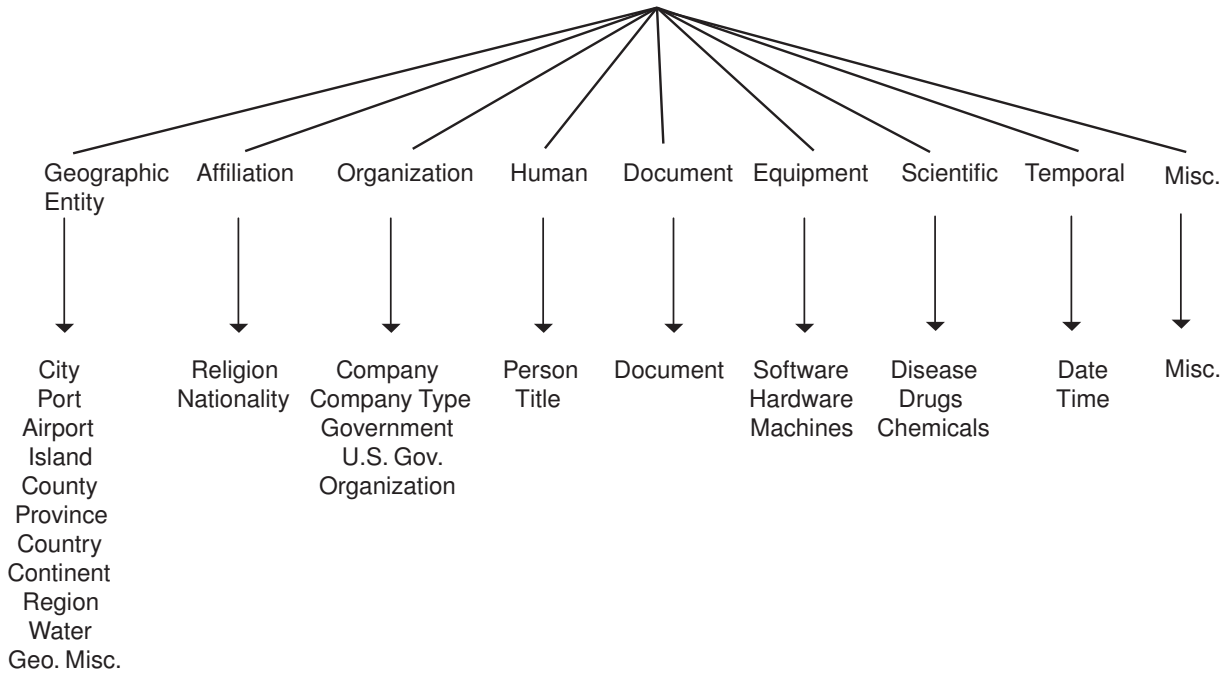

Abbildung 2.1: Kategorisierung der Named Entities nach Paik et al.[17]

Borthwick hingegen postuliert - sehr selbstsicher und bestimmt - sieben[18] Eigennamen-Kategorien:

> Named entity recognition (which might also be called 'proper name classification') is a computational linguistics task in which we seek to classify every word

[17]Die Abbildung ist [Pai96] entnommen. Dort ist sie mit dem Text "Proper Noun Categorization Scheme" beschriftet, was zeigt, dass auch in [Pai96], wie bereits in [Bor99], Zeichen mit Bezeichnetem vermengt wird.

[18]Es sind nicht acht, weil die Restgruppe "none-of-the-above" dabei ausgenommen ist. Vgl. das folgende Zitat.

> in a document as falling into one of the eight categories: person, location, organization, date, time, percentage, monetary value, and 'none-of-the-above'.[19]

Hier ist erstens abzulesen, dass versäumt wird, das sprachliche *Zeichen* (den *Eigennamen*) von dem *Bezeichneten* (der *Named Entity*) zu trennen. Diese Vermengung ist absolut üblich, und mit dem im Zusammenhang mit Eigennamen-Erkennung immer wieder angetroffenen Ausdruck *Named Entity Recognition (NER)* ist eigentlich die Erkennung von *Bezeichnern* für Named Entities gemeint.[20]

Zweitens zeigt sich ein weites Verständnis von Named Entities: Nicht nur Lebewesen und sonstige konkreten Dinge werden dazugezählt, sondern auch abstrakte wie Datumsangaben, Prozentangaben und Geldbeträge.

Dieses weite Verständnis von NEs ist auch bei der MUC-7[21] anzutreffen, allerdings in etwas abgeschwächter Form. Die Teilaufgaben[22], in die IE eingeteilt wird, werden an der MUC-7 in so genannten *Tasks*[23] bearbeitet. An einem dieser Tasks beschäftigte man sich mit der Erkennung der Bezeichner der von Borthwick genannten sieben Kategorien. Unter dem Titel *Named Entity Task (NE-Task)* sollten also Bezeichner für Organisationen, Personen, Orte, Datumsangaben, Zeitangaben, Prozentangaben und Geldbeträge erkannt werden. Dass unter *Entitäten* dann aber doch nur die drei erstgenannten verstanden werden, wird aus der *MUC-7 Named Entity Task Definition* klar:

> The Named Entity task consists of three subtasks (entity names, temporal expressions, number expressions). The expressions to be annotated are 'unique identifiers' of entities (organizations, persons, locations), times (dates, times), and quantities (monetary values, percentages).[24]

Gemäss dieser Definition haben Entitäten (Organisationen, Personen und Orte) als Bezeichner *Eigennamen* ("names"), im Gegensatz dazu werden Datums-, Zeit-, Prozentangaben und Geldbeträge lediglich als temporale und quantitative *Ausdrücke* ("expressions") bezeichnet. Ganz so weit wie Borthwick, der sogar "Named entity recognition" mit "proper name classification" gleichsetzt, geht man an der MUC-7 also nicht. Dennoch: Die Wortwahl "unique identifiers" weist darauf hin, dass auch temporale und quantitative Ausdrücke in gewisser Hinsicht als Bezeichner von etwas Einmaligem verstanden werden, schliesslich findet beispielsweise jeder Zeitpunkt nur ein Mal statt.

Als letztes Beispiel für die unterschiedlichen Ansichten, welche Ausdrücke als Eigennamen betrachtet werden sollen, soll [Cuc99] dienen, wo die abweichende Auffassung explizit formuliert ist:

---

[19] [Bor99], Seite 1.

[20] Wie der Begriff *Named Entity Recognition (NER)* in dieser Arbeit verwendet wird, ist in 2.4.1 nachzulesen.

[21] Was MUC-7 bedeutet wird in Kapitel 1 erläutert. Noch etwas ausführlichere Informationen finden sich in 2.4.3.

[22] Zur Einteilung von IE in Teilaufgaben vgl. Kapitel 1.

[23] An der MUC-7 wurden fünf verschiedene Tasks definiert. Mehr dazu vgl. Unterkapitel 3.3.

[24] [Chi97a].

> In this paper, we actually used for PNs [proper nouns] the MUC7 categories (plus one), for sake of comparison with available evaluations. Among the 7 MUC categories, only 4 are in fact 'real'[25] proper nouns: Organization (e.g. IBM) Location (e.g. Mount Everest) Money (e.g. Ecu) and Person (e.g. Mr. John Smith).

Interessant ist, dass all die Kategorien, die in [Chi97a] lediglich als "expression" bezeichnet werden, in [Cuc99] ebenfalls von den Eigennamen ausgeschlossen werden, mit Ausnahme der Währungsangaben.

Die unscharfen Grenzlegungen bei der Frage, was zu den Named Entities respektive zu den Eigennamen gehört und was nicht, führen dazu, dass jede Person, die sich mit dem Thema befasst, selber definieren muss, was sie unter Named Entities respektive Eigennamen versteht. Zu einem ähnlichen Schluss kommt Mikheev: "What counts as a Named Entity depends on the application that makes use of the annotations."[26] Zusammenfassend kann man sagen, dass ein Konsens darin besteht, Organisationen, Personen und Orte immer als NEs und ihre Bezeichner immer als Eigennamen zu betrachten. Weitere konkrete Dinge wie beispielsweise Produkte und abstrakte wie Datums-, Zeit- oder Währungsangaben werden von Fall zu Fall - je nach Bedarf - als Named Entities respektive deren Bezeichner als Eigennamen angesehen.

2.4 Entscheidung für Personen, Orte, Organisationen und Produkte

Die in der vorliegenden Arbeit gezogenen Grenzen lehnen sich an diejenigen des Grammatik-Dudens: Als Eigennamen werden Bezeichner für Personen, Orte, Organisationen und Produkte betrachtet. Datums-, Zeit-, Prozent- und Währungsangaben gelten zwar als Entitäten, aber *nicht* als Named Entities - und somit deren Bezeichner auch nicht als Eigennamen.

Diesem Entschluss liegen Gedankengänge vor allem philosophischer, aber auch pragmatischer Art zu Grunde.

2.4.1 Gründe philosophischer Art

Der eine Gedankengang beginnt mit der Frage, warum ein Ding (Gegenstand, Lebewesen, Ereignis) einen Namen erhält. In dem diesem Kapitel vorangestellten Motto sinniert Peter Stamm, dass Dingen Namen vergeben werden, damit man besser über sie reden kann, vielleicht auch weil sie einem eine gewisse Sicherheit verleihen. Die Sicherheit, die dem Menschen durch Namen verliehen wird, geht nach Stamm bereits in Richtung Weltbeherrschung. Wer nicht so weit gehen möchte, wird zustimmen, dass Gesprächspartner

[25] Fussnote in [Cuc99]: "Categories such as DATE, TIME, etc. reflect specific language patterns (e.g. 12-6-98) that are best handled with ad-hoc grammars."

[26] [Mik99].

Missverständnisse besser vermeiden können - und somit sicherer über die Welt reden können - wenn sie bestimmten, für sie wichtigen Dingen klare Bezeichner - Namen - zuordnen.[27]

Damit ist auch bereits angetönt, welchen Dingen der Welt Namen zugeordnet werden, nämlich jenen, denen wir einen bedeutenden Stellenwert einräumen. Dazu gehören in erster Linie die Mitmenschen. Diese genau zu identifizieren ist dem Menschen ein Bedürfnis[28], darum werden *Personennamen* vergeben. Um sich in der Welt (physisch) bewegen zu können - auch ein Grundbedürfnis des Menschen - teilt der Mensch sie in geografische Objekte ein, denen er zur leichteren Orientierung *Ortsnamen*[29] verleiht.

Man könnte sich nun fragen, warum beispielsweise Berge einen Namen erhalten, Bäume in der Regel jedoch nicht. Wieder ist der Grund in der Grösse der Bedeutung für den Menschen zu suchen. Diese Bedeutungsgrösse ergibt sich aus verschiedenen Faktoren (die folgende Aufzählung versteht sich nicht als abgeschlossene Liste):

Dauerhaftigkeit: Ein Berg steht mehrere tausend Jahre am (mehr oder weniger) selben Ort, er bildet so einen Fixpunkt, an dem man sich orientieren kann. Ein Baum überdauert nur einen Bruchteil dieser Zeit, je nach Sorte und Verwendung zwischen 1 und ca. 1000 Jahre, wobei 1000 Jahre eher die Ausnahme bildet. Ein Berg wird nicht so einfach abgebaut oder versetzt, wie ein Baum gerodet werden kann.

“Kultur”: In bestimmten Fällen kann es aber trotzdem vorkommen, dass auch Bäume Namen erhalten. Beispielsweise darf der Ausdruck *Linde von Linn* als Eigennamen eines bestimmten Baumes in Linn[30] bezeichnet werden. Dieser Baum hat aus kulturellen Gründen eine solche Bedeutung erlangt, dass er als Wahrzeichen des Dorfes ins Dorfwappen aufgenommen wurde (vergleiche Abbildung 2.2) und auch ein beliebtes Ausflugsziel darstellt.

Verbundenheit: Tieren, zu denen die Menschen eine (engere) Beziehung aufbauen, geben sie einen Namen, weil diese Tiere im Leben dieser Menschen einen bestimmten Stellenwert eingenommen haben: Dem Vogel, der zufällig an seinem Fenster vorbeifliegt, gibt der Mensch keinen Namen, demjenigen Vogel, der jeden Morgen auf seinem Fenstersims sitzt und mit dem er ein regelmässiges Schwätzchen hält, hingegen schon.

[27] Jeder hat wohl schon einmal die Situation erlebt, in der man über eine Person sprechen möchte, deren Namen man nicht kennt. Beschreibungen wie Brille, Igelfrisur, etwa 1.80m gross, schaut so komisch, etc. tragen meist wenig zur klaren Identifikation bei.

[28] Darauf einzugehen, warum es dem Menschen ein Bedürfnis ist, seine Mitmenschen mittels Namen zu identifizieren und somit zu unterscheiden, ginge in dieser Arbeit zu weit. Wahrscheinlich ist dies aber auch nicht nötig, da jeder Leser selbst Mensch ist und genug Lebenserfahrung besitzt, um dieser Aussage zustimmen zu können.

[29] Mit dem Begriff *Ortsname* sind Bezeichner für alle geografischen Objekte gemeint, also nicht nur - wie der Begriff wahrscheinlich vermuten lässt - Städte, Dörfer etc. sondern auch Länder, Strassen, Berge, Gebirge, Flüsse, Seen, Meere, Fluren und andere Örtlichkeiten (vgl. die Aufzählung der Örtlichkeiten in [Dud98], Seite 196).

[30] Linn ist ein kleines Dorf im Kanton Aargau, Schweiz.

Abbildung 2.2: *Linde von Linn* ist ein Eigenname. Das Wahrzeichen der Aargauer Gemeinde Linn ziert auch das Gemeindewappen.

Damit ist ein wichtiges Charakteristikum von Namen angesprochen: Ein Name *wird* von einem oder mehreren Menschen *vergeben*. Zur Vergebung eines Namens fühlt sich ein Mensch dann veranlasst, wenn er zu der zu "taufenden" Entität eine besondere Beziehung hat, wenn sie ihm nicht gleichgültig ist.

Aus diesem Grund wurde für vorliegende Arbeit beschlossen, nur solche Dinge als *Named Enities (NEs)* anzusehen, zu denen ein Mensch eine besondere Beziehung aufbauen kann. Zu Datums-, Zeit-, Prozent- und Währungsangaben baut der Mensch kaum je eine besondere Beziehung auf, darum werden sie hier als "normale" Entitäten angesehen. *Eigennamen* sind die Bezeichner für NEs. Beim Begriff *Named Entity Recognition* muss allerdings vom vorgegebenen Kurs abgewichen werden: Bliebe man konsequent, so dürfte mit *Named Entity Recognition (NER)* ausschliesslich die Erkennung von NEs respektive von Eigennamen gemeint sein. Dies führt aber zu Kollisionen mit der Art, wie der Begriff in den vorgestellten Systembeschreibungen verwendet wird: Dort ist mit NER fast immer die Erkennung von Eigennamen *und* von Bezeichnern anderer Entitäten wie Datums-, Zeit-, Prozent- und Währungsangaben gemeint. Um Kollisionen zu vermeiden, schliesst man sich in dieser Arbeit dieser üblichen Verwendung an. In dem Falle, wo explizit nur die Erkennung von Eigennnamen gemeint ist, wird diese auch so genannt: *Eigennnamen-Erkennung*. Auch der Begriff *NE-Task* wird im gleichen Sinne verwendet, wie er an der MUC-7 definiert wurde, da er immer nur im Kontext der MUC-7 benutzt wird. Mit *NE-Task*, manchmal auch explizit als *MUC-7-NE-Task* bezeichnnet, ist also immer der MUC-7-Task gemeint, an dem Bezeichner für Organisationen, Personen, Orte, Datumsangaben, Zeitangaben, Prozentangaben und Geldbeträge erkannt werden mussten.

Nun könnte man allenfalls einwenden, dass Datumsangaben von Tagen, an denen wichtige Dinge geschehen sind, für den Menschen eine grosse Bedeutung erlangen können und er die Datumsangabe mit Gefühlen verbindet. Als Beispiel sei hier der 11. September (2001) genannt, an dem in New York das World Trade Center durch ein Attentat zerstört und dabei einige tausend Menschen getötet wurden. Viele Menschen vor allem der westlichen Welt waren dadurch sehr erschüttert und über Monate hinweg war dieses Ereignis das

dominante Thema in den Nachrichten. Der Ausdruck *11. September* wurde zum Inbegriff eines schrecklichen Ereignisses und verbunden mit Gefühlen wie Trauer, Verzweiflung, Wut, Unsicherheit. Von diesem Standpunkt aus gesehen könnte man den Ausdruck *11. September* als Name bezeichnen. Da aber ein solcher Fall eine Ausnahme bildet, wurde davon abgesehen, Datumsangaben als NEs anzusehen.

Organisationen wie Firmen, Vereine, Vereinigungen, Ministerien usw. werden deshalb als NEs verstanden, weil sie aus *Gruppen von Menschen*, die ein gemeinsames Ziel haben, gebildet werden. Jede Gruppe gibt sich selbst einen Namen - in vorliegender Arbeit wird dieser *Organisationsname* genannt - damit sie identifizierbar, von anderen Gruppen unterscheidbar ist. Dass Organisationen einen ähnlich wichtigen Stellenwert einnehmen wie Menschen, ist zudem daran zu erkennen, dass sie meistens auch als *juristische Personen* bezeichnet werden (gegenüber von Bezeichnung für Menschen als *natürliche Personen*).

Produktenamen bilden einen Sonderfall in der Reihe der Ausdrücke, die hier als Eigennamen betrachtet werden. Bei ihnen schlägt sich das Merkmal der Einzigartigkeit in einem anderen Sinne nieder: Ein Produktename bezeichnet nicht ein einzelnes Objekt, sondern eine *Menge gleicher Objekte*. Als Entität wird also eine Menge mehrerer Dinge, in diesem Falle Produkte, angesehen. Beispielsweise bezeichnet der Produktename *OmniBook XE3 F3956HT* ein Notebook der Firma Hewlett Packard, das ganz bestimmte Eigenschaften bezüglich Grösse des Bildschirms, Prozessorleistung, Speichergrösse etc. besitzt, durch die es sich von anderen Notebooks unterscheidet und somit *einzigartig* ist. Mit dem Produktenamen kann zwar ein einzelnes solches Notebook gemeint sein, meist werden damit aber alle Vertreter dieser Art von Notebooks bezeichnet. Letzteres entspricht nicht der Natur eines Eigennamens, der normalerweise wie oben gesehen ein *Individuum* bezeichnet, ist hier aber dennoch möglich, weil die Notebooks *OmniBook XE3 F3956HT*, generell also die Mitglieder der Menge, vollkommen gleich sind.[31]

2.4.2 Gründe pragmatischer Art

In der vorliegenden Arbeit spielten auch Gründe eher pragmatischer Art eine Rolle, warum für die vier Arten von Namen entschieden wurde.

Der *Konsens* in den untersuchten Systembeschrieben ist dahingehend, dass alle vorbehaltlos die drei Klassen Personen, Orte, Organisationen als NEs betrachteten. Diesem Konsens wurde hier durch die Wahl der entsprechenden Eigennamenarten Rechnung getragen. Produktenamen wurden dazugenommen, weil sie mit den anderen drei Eigennamenarten die Eigenschaft teilen, eine *offene Klasse* zu sein. Dieses Charakteristikum ist ein entscheidendes Kriterium zur Unterteilung in Eigennamen und Nicht-Eigennamen: Eigennamen werden laufend neue erfunden, Nicht-Eigennamen hingegen bilden eine geschlossene Klasse.

[31] Dass auch der Ausdruck *OmniBook XE3* - ein Bezeichner für eine Reihe von ähnlichen, zur Produktefamilie zusammengenommenen Produkten - als Produktename verstanden wird, ist der Gepflogenheit gleichzusetzen, auch Ausdrücke, die aus Menschen bestehende Familien bezeichnen, als Personennamen anzusehen.

Die häufigsten neuen Namen werden für Produkte kreiert, das heisst, Produktenamen sind am kurzlebigsten. Kaum ist einer entstanden, verschwindet er häufig nach kurzem Gebrauch auch schon wieder, sobald ein neues Produkt (mit neuem Namen), welches das alte übertrifft, auf den Markt kommt. Auch Organisationsnamen, insbesondere Firmennamen, sind schnelllebig, weil es üblich ist, dass Firmen umstrukturiert werden, fusionieren, sich neuen Aufgaben zuwenden und dies mit einem neuen Namen kommunizieren wollen. Weniger häufig - zumindest in der westlichen Kultur - kommt die Erfindung neuer Namen für Personen vor. Viele haben lange Tradition und werden über Jahrhunderte hinweg immer wieder verwendet. Im Gegensatz zu diesen nun als Eigennamen definierten Bezeichnern setzen sich Datums-, Zeit-, Prozent- und Währungsangaben - mit wenigen Ausnahmen - aus den Ziffern 0 bis 9 zusammen, kombiniert meist mit gewissen Satz- und Sonderzeichen und eventuell bestimmten Ausdrücken aus einer geschlossenen Liste. Beispielsweise besteht die geschlossene Liste für zusammengesetzte Datumsangaben aus den zwölf Monatsnamen; allerdings kann ein Datum auch in reiner Zahlenform angegeben werden. Andererseits gelten als Datumsangabe auch Ausdrücke wie *gestern*, *jetzt* - solche Ausdrücke bilden eine weitere geschlossene Klasse.

Unterstützend für den Entscheid für Produktenamen wirkt auch die Tatsache, dass sie meist mit Firmennamen (eine Untergruppe von Organisationsnamen) einhergehen und durch ihr häufiges Vorkommen einen genügend wichtigen Stellenwert erhalten, dass es sinnvoll erscheint, sie zu den hier betrachteten Eigennamen zu zählen.

Natürlich könnte man noch weitere Ausdrücke als Eigennamen bezeichnen, für die mit den gleichen Argumenten wie denjenigen, die für die gewählten vier Arten ins Feld geführt wurden, plädiert werden könnte. Beispielsweise könnte man Büchertitel, Religionen, Nationalitäten und anderes mehr als Eigennamen bezeichnen. Warum sie hier nicht als weitere, eigene Klasse von Eigennamen erwähnt sind, hat zwei Gründe: Zum einen ist vor allem die Klasse Produktename sehr flexibel - Büchertitel könnten zum Beispiel darunter gezählt werden. Zum anderen wurde der Entschluss zur Einschränkung auf die vier Klassen auf Grund der vorhandenen Literatur gefällt, die selten über die Kategorien Personen-, Orts-, Organisations- und Produktenamen hinausgeht. Wie bereits angetönt, beschränkt sich der Grossteil auf die ersten drei, nur einige wenige betrachten auch Produktenamen.

Trotz der Ausklammerung von Bezeichnern für Datums-, Zeit-, Prozent- und Währungsangaben aus der Gruppe der NEs können sie in der vorliegenden Arbeit nicht ausser Acht gelassen werden, da die meisten der vorgestellten Systeme an der MUC eingereicht wurden, wo für den NE-Task diese Ausdrücke auch erkannt werden mussten.

2.4.3 MUC-7 und Fokus Produktenamen-Erkennung

Ebenfalls aus pragmatischen Gründen werden in dieser Arbeit vorwiegend NER-Systeme vorgestellt, die am NE-Task der MUC-7 mitmachten. Pragmatisch darum, weil dank einheitlicher Zielvorgaben[32] die Leistungen der Systeme gut verglichen werden können.

Da es das Ziel dieser Arbeit ist, den Stand der Kunst in der Eigennamen-Erkennung

[32] Vgl. [Chi97a].

zu ermitteln, war es auch wichtig, einen Ort zu finden, an dem ein breites, repräsentatives Spektrum von Methoden bereitgestellt wird. Die etablierte[33] Konferenz MUC kann als eine Art Wettbewerb[34] angesehen werden, die Forschergruppen aus aller Welt - sowohl aus Firmen als auch aus Universitäten - anspornte, ein NER-System zu bauen, dessen Leistung die der anderen übertreffen soll. Diese Vorraussetzungen garantierten sowohl viele verschiedene als auch hochstehende Ideen und Methoden zur Erkennung von Eigennamen. Mit der Wahl von MUC-Systemen konnte also die Forderung nach einem breiten, repräsentativen Spektrum erfüllt werden.

Obwohl wie oben ausgeführt Produktenamen mit Personen-, Orts- und Organisationsnamen vieles gemeinsam haben, werden sie am MUC-7-NE-Task nicht zu den zu erkennenden Eigennamen gezählt. Auch in der übrigen Literatur zur NER ist von Produktenamen viel seltener die Rede als von den drei anderen. Dass Produktenamen häufig ausgeklammert werden, liegt wohl weniger daran, dass sie weniger wichtig wären, als vielmehr daran, dass sie schwieriger zu erkennen sind. Die Hauptschwierigkeit, sie von Organisationsnamen abzugrenzen, entsteht wohl dadurch, "dass es bereits für Menschen offensichtlich keine einfache Aufgabe ist, zu merken, ob ein Name ein Produkt oder eine Firma meint. Für ein Computersystem ist dies ein praktisch unlösbares Problem. Das System müsste über eine Unmenge an semantischer Information verfügen [...]"[35]. Weil die Erkennung von Produktenamen dennoch nützlich ist, die in dieser Arbeit vorgestellten Systeme sich aber auf Personen-, Orts- und Organisationsnamen beschränken, soll in dieser Arbeit neben dem Vorstellen der angewandten Methoden ein spezieller Fokus darauf gerichtet werden, welche der Methoden sich für Produktenamen-Erkennung am besten eignen.

[33] Dass die MUC eine etablierte Stellung einnimmt, ist daran zu erkennen, dass in Forschungsartikeln immer wieder auf sie hingewiesen wird respektive an ihr teilnehmende Forschergruppen zitiert werden.

[34] Im Folgenden wird auch häufig vom MUC-7-Wettbewerb gesprochen werden.

[35] [Rot01], S3f.

Kapitel 3

Zweck der Eigennamen-Erkennung

Maschinelle Sprachverarbeitung findet heute viele, unterschiedlich komplexe, Anwendungen: Suchmaschinen (zum Beispiel im Internet), Systeme zur maschinellen Übersetzung, zur automatischen Erstellung eines Indexes, zur Textzusammenfassung, -filterung oder -klassifikation, zum Beantworten von Fragen (als so genannte Frage-Antwort-Systeme respektive Q-&-A-Systeme) usw. Eigennamen-Erkennung ist ein wichtiger Teilschritt auf dem Wege zur maschinellen Sprachverarbeitung, weil sie zum einen die syntaktische und semantische Analyse unterstützt und zum anderen eine Teilaufgabe in einer komplexeren Anwendung, wie zum Beispiel der Informations-Extraktion (IE)[1], löst.

3.1 Unterstützung der syntaktischen Analyse

Grundlage für alle Anwendungen ist die *maschinelle Analyse* von Texten. Eine wichtige Rolle spielt dabei die *syntaktische Analyse* - das *Parsing* - von Sätzen. Diese komplexe Aufgabe kann durch vorgängige Erkennung von Named Entities vereinfacht werden. Werden beispielsweise Kombinationen von Zahlen, Satzzeichen und bestimmten Buchstabenfolgen als eine Datumsentität erkannt, so dient dies dem Parser als wichtiger Vorverarbeitungsschritt: Diese Folge von Tokens muss der Parser nicht mehr einzeln analysieren, sondern sie werden ihm bereits als Einheit dargeboten. Der gleiche Effekt liegt bei Eigennamen vor: Diese können aus mehreren Wörtern bestehen, teilweise auch zusätzlich Zahlen oder Sonderzeichen enthalten (wie zum Beispiel das Sonderzeichen '&' bei Firmennamen, die mit '& Co.' enden). Bei vorgängiger Erkennung werden solche mehrteilige Namen dem Parser ebenfalls als Einheit übergeben.

Dass beispielsweise ein Chunk-Parser ohne vorgängige NER nachweislich viele Fehler macht, die mit einer solchen nicht passieren würden, zeigt Steiner in einer Arbeit über Chunk-Analyse[2]. Der Chunk-Parser scheitert genau daran, dass er grössere Einheiten nicht als solche erkennt, sondern sie fälschlicherweise als mehrere kleinere Einheiten interpretiert. Zur Lösung des Problems schlägt Steiner vor, die "Vorstrukturierung der NE-Ausdrücke

[1] Einführende Informationen zur IE finden sich im Kapitel 1, ausführlichere in Unterkapitel 3.3.
[2] Vgl. [Ste01].

in einem separaten Modul vor der Chunk-Analyse vorzunehmen."[3]

Die korrekte Erkennung von NEs bewirkt auch, dass das Problem der syntaktischen Ambiguität eingedämmt wird: Die meist grosse Anzahl an möglichen syntaktischen Analysen eines Satzes kann so reduziert werden, da *vor* der syntaktischen Analyse möglicherweise ambige Elemente schon bestimmt sind. Dieser willkommene Effekt setzt allerdings voraus, dass die NER auch wirklich korrekt ist: Ansonsten tritt der ungewünschte Fall ein, dass die richtigen statt der falschen Parse-Bäume herausgefiltert werden, was die Leistung des Gesamtsystems erheblich herabsetzen würde. Bei der NER ist also vor allem auf deren hohe Präzision zu achten, wenn man die Leistung eines Sprachanalyse-Systems verbessern möchte.

3.2 Unterstützung der semantischen Analyse

Aber nicht nur die syntaktische, sondern auch die *semantische Analyse* wird durch vorgängige NER unterstützt. Für die meisten Anwendungen ist es wesentlich, dass Eigennamen als solche erkannt werden. Ob *Brown* ein Eigenname oder eine Farbe ist, ist ein bedeutender Unterschied. Dank korrekter semantischer Klassifizierung können beispielsweise bei Suchmaschinen viele irrelevante Fundstellen ausgeschlossen werden. Ein System zur maschinellen Übersetzung vom Deutschen ins Englische wird dank Eigennamen-Erkennung nicht versuchen, *Martin Volk* fälschlicherweise als *Martin People* oder ähnliches zu übersetzen. Und im Bereich von Frage-Antwort-Systemen ist NER besonders bei W-Fragen[4] nützlich. Ein *Wer* zu Beginn einer Frage deutet darauf hin, dass in der Antwort ein Eigenname verlangt wird, mit grosser Wahrscheinlichkeit ein Personenname[5]. Ein *Wo* sucht nach einer Ortsangabe - in vielen Fällen ist dies der Eigenname eines Landes, einer Stadt etc., was in der Antwort einen Ortsnamen verlangt. Dass auch Datums-, Zeit-, Prozent- und Währungsangaben in Frage-Antwort-Systemen von Bedeutung sind, weil durch sie Antworten auf *Wann-* respektive *Wieviel*-Fragen ermöglicht werden, ist der Grund, warum viele NER-Systeme diese ebenfalls als NEs betrachten und zu erkennen suchen.[6]

3.3 Hauptanwendung: Eigennamen-Erkennung als Teilaufgabe in komplexerer Anwendung

Aus der obigen Ausführung wird ersichtlich, dass Eigennamen-Erkennung meist eine *Teilaufgabe in einer komplexeren Anwendung* darstellt. Typischste komplexe Anwendung ist

[3] [Ste01], Seite 247.

[4] W-Fragen sind Fragen, die mit einem Fragewort beginnen, das seinerseits mit dem Buchstaben *W* beginnt, also z.B. *Wo, Wer, Was, Wieviel, Wann* etc. Im Gegensatz dazu gibt es die Ja-Nein-Fragen, die mit einem Verb beginnen, z.B. *Hast du schon gegessen?*.

[5] Es ist auch denkbar, dass bei einer Wer-Frage nach anderen Eigennamen gesucht wird, beispielsweise nach einem Firmennamen oder einem Ortsnamen.

[6] Vgl. dazu die Ausführungen in Kapitel 2.

die Informations-Extraktion (IE). In der IE wird versucht, aus Texten nicht-ambige Daten, die ein festgelegtes Format haben, zu extrahieren. Die so gewonnenen Daten können dem Benutzer direkt angezeigt werden, für spätere Analyse in einer Datenbank oder Tabelle gespeichert werden oder zur Erstellung von Indexen benutzt werden. In einer Beschreibung der IE bezeichnet Cunningham die NER zwar als einen der "five types of information extraction"[7] - was nach dem Verständnis einer *eigenständigen* Anwendung aussieht - zeigt aber durch die analoge Benennung "[five types of] information extraction *tasks*", dass er NER als *eine* Teilaufgabe (*Task*[8]) in der fünfteiligen Vorgehensweise bei der IE versteht.

Dasselbe Verständnis von IE, die in fünf Tasks unterteilt werden kann, herrschte auch an der MUC-7 vor. NER entspricht dabei dem NE-Task[9], und als die vier anderen Tasks von IE werden die folgenden genannt:[10]

- *Coreference Resolution (CO-Task)*
  Auflösung von Koreferenzen. Typisch ist hier das Erkennen, welche Pronomen mit welchen Named Entities zu verbinden sind.
- *Template Element construction (TE-Task)*
  In vorher festgelegte Schablonen (engl. *Templates*) werden Informationen (Attribute) über die Named Entities eingetragen.
- *Template Relation construction (TR-Task)*
  Findet Relationen zwischen Entitäten.
- *Scenario Template production (ST-Task)*
  Die Entitäten werden in grössere Zusammenhänge eingebettet. Diese Einbettung wird gemäss vorher festgelegten Szenarien vorgenommen.

Die fünf Tasks sind in der dargestellten Reihenfolge nacheinander angeordnet und gewinnen immer komplexere Information (der NE-Task ist dabei der erste). Diese Anordnung verdeutlicht die Rolle, die die NER in komplexen Sprachverarbeitungssystemen wie der IE spielt: Werden ausgewählte, durch NER erkannte Entitäten und Pronomen miteinander koreferenziert, dann den Entitäten Attribute und Beziehungen untereinander zugeordnet, um sodann ganze Szenarien zu konstruieren, so ist einzusehen, dass NER einen wesentlichen Beitrag zum Verständnis von Texten leistet - und das ist schliesslich das Ziel eines ausgereiften Sprachverarbeitungssystems.

3.4 Direkte Verwendung auf Nebenschauplatz

Wie Cunningham ordnet auch Borthwick NER bei der IE ein: "In the taxonomy of computational linguistics tasks, it [NER] falls under the domain of 'information extraction'."[11].

[7] [Cun99], Seite 3.
[8] Der Begriff *Task* wurde bereits in Unterkapitel 2.3 eingeführt.
[9] Zum Begriff *NE-Task* (Named Entity Task) vgl. auch Unterkapitel 2.3.
[10] Vgl. zum Beispiel [Cun99], Seite 3.
[11] [Bor99], Seite 1.

Gleichzeitig zeigt Borthwick aber auch, dass die Verwendung von Eigennamen-Erkennung sehr direkt sein kann, nicht zwingend einer unter vielen anderen Teilschritten einer komplexen Anwendung sein muss. Als Beispiele nennt er die Erstellung einer Liste mit Eigennamen, die man vor dem Lesen eines Artikels sichten könnte, um sich vorab über Personen, Orte und Firmen ins Bild zu setzen. Auch wäre denkbar, dass Magazine über Menschen (zum Beispiel *People Magazine*) oder Printmedien mit Schwerpunkt Wirtschaft (zum Beispiel *The Wall Street Journal*) Namen von Personen respektive Firmen dank Eigennamenerkennung fett hervorheben können.

Solche eigenständigen Anwendungen von Eigennamen-Erkennung sind allerdings - wie an der geringen Anzahl von Beispielen auch leicht festzustellen ist - dünn gesät. Hauptsächlich dient NER als Verarbeitungsschritt in einem umfangreichen Sprachverarbeitungssystem wie ein IE-System eines ist.

Kapitel 4

Probleme bei der Eigennamen-Erkennung

Bei der Bestimmung von Eigennamen stellen sich teilweise die gleichen Probleme wie bei der Bestimmung gewöhnlicher Nomen und Nominalphrasen, vor allem im Bereich struktureller und semantischer Ambiguität. [Cuc99] meint zwar, Eigennamen seien im Vergleich zu anderen Wortkategorien (zum Beispiel zu Verben oder gewöhnlichen Nomen) weniger ambig (er spricht damit die semantische Ambiguität an), oft gäbe es für Eigennamen nur eine angemessene Bedeutung. Dennoch: Eigennamen zu erkennen birgt Probleme mannigfaltiger Art in sich, darunter auch solche, die für Eigennamen spezifisch sind.

4.1 Strukturelle Ambiguität

Vor allem Wacholder et al. weisen auf das Problem struktureller Ambiguität hin, das Eigennamen mit normalen Nominalphrasen gemein haben. Die folgende Auflistung ist vorwiegend aus [Wac97] übernommen, ergänzt um einige weitere Punkte und Beispiele aus anderen Arbeiten[1].

4.1.1 PP-Attachment

Das Problem, ob eine Präpositional-Phrase (PP), die auf eine Nominalphrase(NP) folgt, mit dieser eine neue grössere NP bildet, oder ob die PP uneingebunden bleibt, trifft man nicht nur bei normalen Nominalphrasen an. Als Beispiel für die beiden möglichen Strukturen soll folgende Gegenüberstellung dienen: *Midwest Center for Computer Research* versus C*arnegie Hall for Irwin Berlin*. Beim ersten Ausdruck handelt es sich gesamthaft um einen Eigennamen, wobei die PP ein Bestandteil davon ist. Der Ausdruck hat folgende Struktur: `NP`[*Midwest Center* `PP` [*for* `NP` [*Computer Research*]]]. Beim zweiten Ausdruck gehört die PP nicht zur ersten NP, sondern steht im Strukturbaum auf dersel-

[1] Vgl. [Bor99] und [Mik99].

ben Ebene wie diese: `NP` [*Carnegie Hall*] `PP` [*for* `NP` [*Irwin Berlin*]]. Die erste NP ist ein selbstständiger Eigenname, ebenso diejenige, die mit der Präpositon eine PP bildet.

Diese Ambiguitäten sind nicht immer zu lösen, wie das auch beim PP-Attachment bei normalen Nomen oft nicht möglich ist (vergleiche den viel zitierten "Teleskop-Satz": *Er sah den Jungen mit dem Fernglas.*). Beispielsweise ist es ohne Wissen über den offiziellen Namen einer Organisation kaum möglich herauszufinden, ob die in der PP angegebene Ortsangabe Teil des Namens ist oder nicht: In *City University of New York* gehört die PP *of New York* zum Eigennamen, wohingegen in *The Museum of Modern Art in New York City* die hintere PP *in New York City* nicht dazugehört. Hier ist zur exakten Festlegung der Grenzen des Eigennamens Weltwissen nötig. Weitere Beispiele: *Western Co. of North America*; *Commodity Exchange in New York*; *Hebrew University in Jerusalem, Israel*; *Music Masters of Milan.*

4.1.2 Konjunktions-Skopus

Ein weiteres Problem, das Eigennamen mit Nominalphrasen gemein haben, ist bei Eigennamen mit Konjunktionen anzutreffen. Konjunktionen können entweder innerhalb eines Eigennamens stehen, also Bestandteil davon sein, oder aber lediglich zwei eigenständige Eigennamen verbinden. Die beiden Ausdrücke *Victoria and Albert Museum* und *IBM and Bell Laboratories* haben die gleiche Wortartenstruktur (Nomen Konjunktion Nomen Nomen), jedoch handelt es sich im ersten Fall um den Namen eines (einzigen) Museums, im zweiten Fall dagegen um zwei Namen zweier Computerfirmen, die miteinander durch die Konjunktion *and* verbunden wurden, weil der Kontext es so verlangt.

In [WAC97] ist angemerkt, dass wenige Lösungsansätze beständen, Konjunktions-Skopus-Ambiguitäten aufzulösen.

4.1.3 Genitiv-s

Eine ähnliche Ambiguität wird durch das Genitiv-s[2] ausgelöst. Dieses kann eine Beziehung zwischen zwei Namen ausdrücken (*Israel's Shimon Peres*) oder Teil eines einzigen Namens sein (*Donoghue's Money Fund Report*).

4.1.4 Innere strukturelle Ambiguität

Als letzte strukturelle Ambiguität wird in [WAC97] die innere angeführt ("strucural ambiguity, involving the internal structur of the proper name"[3]). Folgende Beispiele sollen die angesprochene Ambiguität veranschaulichen: *Professor of Far Eastern Art John Blake* und *Professor Art Klein.* In jedem der beiden Ausdrücke ist das Wort *Art* enthalten; einmal ist es Bestandteil des Personennamens, einmal nicht: Die unterschiedlichen Strukturen zeigen dies auf: [[*Professor* [*of Far Eastern Art*]] *John Blake*] und [*Professor* [*Art Klein*]].

[2] Wacholder et al. bezeichnen dieses Genitiv-s als *possessive pronoun.* Vgl. [WAC97].

[3] [WAC97].

Ähnlich ist die Schwierigkeit, die genaue Menge der Wörter zu markieren, die den Eigennamen bilden. Beispiel: Die ganze Wortfolge *Arthur Andersen Consulting* ist als Organisationsname zu markieren, nicht schon *Arthur Andersen* als Personenname. Der umgekehrte Fall findet sich aber in *Canada's Parliament*: *Canada* sollte als Ortsname und *Parliament* als Organisationsname bezeichnet werden, *nicht* die ganze Wortfolge als Organisationsname.[4]

4.2 Semantische Ambiguität

Auch Probleme bezüglich semantischer Ambigiutät kommen sowohl bei normalen Nomen als auch bei Eigennamen vor. Wiederum hält sich die folgende Auflistung vor allem an [Wac97], ergänzt um weitere Punkte und Beispiele aus anderen Arbeiten[5]. Ergänzend wird in 4.2.3 eine Gegenposition zu [Wac97] vorgestellt, die Cucchiarelli et al.[6] vertreten.

4.2.1 Mehrere Kategorien / Metonymie

Viele Eigennamen können gleichzeitig mehreren Kategorien zugeordnet werden. Grund dafür ist unter anderem, dass Firmen nach deren Gründern oder Standorten benannt werden, Orte gerne nach berühmten Personen und Produkte nach den Herstellerfirmen oder umgekehrt.

Ein schönes Beispiel ist der Eigenname *Ford*, der auf vier verschiedene Entitäten referieren kann: Person (*Gerald Ford*), Firma (*Ford Motors*), Produkt (Automarke), Ort (*Ford, Michigan*). *Washington* kann ein Personenname (sogar Vor- und Nachname), aber auch eine Ortsbezeichnung sein (wobei wiederum noch nicht klar ist, ob es eine Stadt oder einen Staat bezeichnet). *Philip Morris* bezeichnet sowohl eine Person als auch die von ihr gegründete Firma. *Coca Cola* steht einerseits für ein Getränk (Produkt), anderseits für die Herstellerfirma. Hier kann schon von Metonymie gesprochen werden.

Weitere Beispiele für Metonymie von Eigennamen: *United States* kann auf eine geografische Entität (Ortsangabe) referieren (*He grew up in the United States.*) oder aber auf eine politische Entität respektive Gemeinschaft (*United States sends more troops to Philippines in rebel fight.*). Desgleichen kann *Wall Street Journal* auf eine Zeitung, auf ihren Inhalt und auf eine Firma referieren. Weitere Beispiele: *MTV* (Fernsehsender, Inhalt, Herstellerfirma), *Boeing* (Flugzeug/Rakete, Herstellerfirma, Firmengründer).

Als wichtigen Faktor zur Auflösung der genannten Ambiguität nennen Wacholder et al. Kontextwissen. Beispielsweise wird der Eigenname *Paris* wohl in den meisten Fällen auf die Hauptstadt von Frankreich verweisen. Geht es aber in einem Text um griechische Mythologie, so ist die Chance gross, dass es sich dann bei *Paris* um einen Personennamen handelt.

[4] Vgl. [Mik99].
[5] Vgl. [Bor99], [McD93] und [Mik99].
[6] Vgl. [Cuc99].

4.2.2 Eigenname versus allgemeinsprachliches Lexikon

In [McD93] wird ausgeführt, dass jedes Wort des allgemeinsprachlichen Lexikons zum Eigennamen gemacht werden kann. Das zu diesem Thema vielzitierte Beispiel lautet: "Her name was *April Wednesday*.". Auch in [Wac97] wird auf das Problem der beliebigen Namensbildung hingewiesen und angemerkt, es sei beliebt, neue unkonventionelle Namen zu erfinden, die semantisch mehrdeutig sind, wie beispielsweise bei den Firmennamen *Thinking Machines*, *Mr. Tall* oder *The House*. Ähnliches steht in [McD93] und als Beispiel wird der etwas spezielle Fall angeführt, wenn wegen gesetzlicher Vorschriften jeder Entität ein anderer Name gegeben werden muss (beispielsweise bei Rennpferden oder Booten). Dann bedeutet dies häufig, dass gewöhnliche Worte in Namen umgemünzt werden.

Doch existieren auch zahlreiche weniger ausgefallene Ausdrücke, die sowohl dem allgemeinsprachlichen Lexikon angehören als auch Eigennamen darstellen. *Candy* ist beispielsweise ein weiblicher Vorname, ist aber auch der englische Ausdruck für *Süssigkeit* respektive *kandieren* etc. *Hope* bedeutet *Hoffnung, hoffen...*, ist aber auch ein Ortsname. *Smith*, *Miller*, *Carpenter* können Berufsbezeichnungen oder Personennamen sein.

Normalerweise wird im Englischen die Problematik dadurch gelöst, dass der Begriff als Eigenname gross-, als allgemeinsprachlicher Ausdruck kleingeschrieben wird. Am Satzanfang ist jedoch auch das allgemeinsprachliche Wort grossgeschrieben, zudem in Überschriften. Das kann neue Ambiguitäten verursachen:
New Coke: Der Eigenname lautet *New Coke* (Produkt).
New Sears: Der Eigenname lautet lediglich *Sears* (Firma).
The New York Times: Der ganze Ausdruck ist ein Name (inkl. *The*).
The IBM: Nur *IBM* ist ein Name.

Die geschilderte Beliebigkeit der Verwendung jedes Wortes als Eigenname lässt sowohl McDonald[7] als auch Mikheev et al.[8] zum Schluss kommen, dass Listen mit Eigennamen unnütz seien. Wollte man eine Liste aller möglichen Eigennamen erstellen und zur Erkennung einsetzen, so würde man massive und willkürliche Ambiguität erzeugen.

Am Rande sei noch der Fall erwähnt, in dem es zwar nicht um Ambiguität geht, dennoch darum, ob ein Ausdruck als Eigenname oder als Ausdruck des allgemeinsprachlichen Lexikons verstanden werden soll. Diese Schwierigkeit liegt beispielsweise beim Ausdruck *Internet* vor, das ein Prinzip meinen könnte und somit ein Ausdruck des allgemeinsprachlichen Lexikons ist, oder aber als Entitätsbezeichnung ein Produktename ist (es gibt ja nur ein einziges Internet).

4.2.3 Gegenposition

Eine abweichende Auffassung von semantischer Ambiguität haben Cucchiarelli et al[9]. Deren Aussage, Eigennamen seien weniger ambig als beispielsweise Verben oder "normale

[7] Vgl. [McD93].
[8] Vgl. [Mik99].
[9] Vgl. [Cuc99].

Nomen", wirkt nach all den von Wacholder et al. aufgezeigten Problemen fast provokativ.[10] Cucchiarelli et al. verweisen auf die These der *einen Bedeutung pro Dokument* (engl. *one-sense-per-document*) von Gale et al.[11] und begründen so eine hohe Performanz beim kontextbasierten Taggen von unbekannten Eigennamen.

Aufgrund der geringen semantischen Ambiguität kommen Cucchiarelli et al. dann zum Schluss, dass es beim semantischen Taggen von Eigennamen weniger darum gehe, mehrdeutigen Wörtern das angemessene Tag zuzuweisen, sondern vielmehr darum, einem *unbekannten* (wahrscheinlich nicht-ambigen) Wort das richtige Bedeutungs-Tag zuzuweisen. Des Weiteren sei zu beachten, dass während jedes mehrdeutige Wort sein eigenes Bedeutungsinventar[12] habe, müssen die geeigneten Tags für einen unbekannten Eigennamen alle aus der gleichen [riesigen] Menge von möglichen Bedeutungen gewählt werden.[13]

Doch Cucchiarelli et al. schliessen die Problematik der semantischen Ambiguität nicht ganz aus, verweisen sie vielmehr in einen bestimmten Bereich, nämlich in den technischen. Als Beispiel führen sie an, dass es selbst für einen Linguisten nicht trivial sei zu entscheiden, ob der Ausdruck *Mac GS Viewer* ein Stück Hard- oder Software bezeichnet. Allerdings könne man sich aber drüber streiten, ob ein solcher Unterschied relevant sei. Schliesslich entwerfen sie ein Szenarium, in dem die Unterscheidung durchaus sinnvoll scheint, nämlich in einer technischen Anwendung, beispielsweise einer On-line-Hilfe.

4.3 Weitere Probleme

4.3.1 Breites Spektrum an Arten von Eigennamen

Wie in Kapitel 2 ausgeführt, wird man sich beim Erstellen eines Eigennamen-Erkennungs-Systems auf bestimmte Arten von Eigennamen spezialisieren, je nachdem, wofür das System verwendet wird. Eine Spezialisierung bedeutet immer auch ein Beschränkung, und da es ein breites Spektrum an Dingen gibt, die Namen haben - zum Beispiel Filme, Bücher, Gesetze, astronomische Phänomene (*the Milky Way*) - wird es immer Namen geben, die sich ausserhalb der festgelegten Kategorien befinden und sich somit nicht einordnen lassen.

4.3.2 Dinge, die nach Personen benannt sind

Dinge, die nach ihren Erfindern oder Entdeckern benannt sind, können hinsichtlich der Abgrenzung zu Personennamen Schwierigkeiten bereiten: Wenn man nur die Personennamen als zu erkennenden Kategorie bestimmt hat, sind die nach ihnen benannten Dinge von den Eigennamen auszuschliessen. Auf dieses Problem wird in [Mik99] hingewiesen, und als Beispiele werden die bekannten Personennamen *Nobel* respektive *Alzheimer* genannt, die

[10] Als einzige semantische Ambiguität gestehen Cucchiarelli et al. die Metonymie zu.

[11] Vgl. [Gal92].

[12] Wobei das Bedeutungsinventar sehr individuell und fein abgestuft sein kann.

[13] Vgl. [Cuc99], Seite 2: "Furthermore, each polysemous word has its own sense inventory, while the appropriate tag, for an unknown proper noun, must be selected from the same 'bag' of possible senses."

als solche erkannt werden müssen. Die nach ihnen benannten Dinge *Nobel Prize* respektive *Alzheimer's* sollten allerdings nicht als Eigennamen markiert werden, es sei denn, ein System sei dafür ausgelegt, Namen von Preisen respektive Krankheiten zu erkennen.

Diese Abgrenzung erweist sich vornehmlich deshalb als schwierig, weil wie im Beispiel von *Nobel Prize* nicht nur der Personenname (*Nobel*), sondern auch der "Dingbezeichner" (*Prize*) grossgeschrieben ist. Im Beispiel von *Alzheimer's* birgt das *'s* das Problem in sich, dass es als normales Genitiv-s und somit *Alzheimer* fälschlicherweise als Personenname interpretiert werden kann.

4.3.3 Offene Klasse

Eigennamen bilden eine offene Klasse - täglich werden neue Namen erfunden. In [Cuc99] wird darauf hingewiesen, dass Eigennamen einen beträchtlichen Prozentsatz des Wortschatzes bei Subsprachen ausmachen, vor allem bei technischen Subsprachen. Besonders illustrativ für die Unbegrenztheit der Klasse der Eigennamen ist der Hinweis in [McD93], dass beim Abarbeiten eines Textes die Anzahl der Eigennamen mit konstanter Rate wächst, wohingegen die Anzahl der neuen Wörter des allgemeinsprachlichen Lexikons asymptotisch wächst. Daraus folge, dass man nicht mit einer "lexikalisierten Grammatik" - damit ist eine Liste von Eigennamen gemeint - arbeiten könne, da man zum Zeitpunkt des Erstellens dieser Liste nicht wissen könne, welche Eigennamen vorkommen. Diese Meinung wird auch in [Mik99] vertreten, wobei die Aussage noch mit dem Hinweis auf die Kurzlebigkeit von Eigennamen (vor allem Firmen- und Produktenamen)[14] und auf die zahlreichen Varianten eines Namens (vor allem Firmennamen[15]) unterstrichen wird. Wollte man diese Probleme mit Listen lösen, käme man nicht weit.[16]

4.3.4 Grossschreibung

Auf das Problem der Grossschreibung sowohl von Eigennamen als auch generell von Begriffen am Satzanfang, in Titeln und solchen, die, obwohl sie keine Eigennamen sind, dennoch grossgeschrieben werden[17], wurde schon in Unterkapitel 4.2 hingewiesen. Würde man also einfach annehmen, dass alle grossgeschriebenen Wörter respektive Wortfolgen Eigennamen seien, würde dies zu vielen Fehlern führen. Auch eine vermeintliche Lösung, alle Wörter am Satzanfang nicht als Eigennamen zu betrachten, würde nicht zum gewünschten Resultat führen, da unter den ausgeschlossenen Wörtern auch Eigennamen respektive Teile davon wären, da diese durchaus auch am Satzanfang stehen können. Stattdessen muss jedes grossgeschriebene Wort, ob am Satzanfang, in Titeln oder sonst wo, in einem ersten

[14] Zur Kurzlebigkeit vgl. Unterkapitel 2.4.2.

[15] Als Beispiel für viele Varianten eines Firmennamens soll dasjenige aus [Mik99] dienen: *The Royal Bank of Scotland plc* oder *The Royal Bank of Scotland* oder *The Royal* oder *The Royal plc*.

[16] Vgl. [Mik99].

[17] Vgl. das Beispiel mit *Nobel Prize* und *Alzheimer's*. [Wac97] nennt weitere Beispiele von grossgeschriebenen Nicht-Eigennamen: *Minimum Alternative Tax*; *Annual Report*; *Chairman*.

Schritt als potenzieller (Teil eines) Eigenname(ns) betrachtet werden und in einem weiteren Schritt durch ein geeignetes Ausschlussverfahren müssen die Nicht-Eigennamen(teile) wieder gestrichen werden.

4.3.5 Wenig morphologische Merkmale

Kober et al. führen aus, dass bei Eigennamen - abgesehen vom Genitiv-s - morphologische Markierungen systematisch fehlten. Morphophonologisch beziehungsweise morphographematisch würden Eigennamen praktisch keinen systematischen Restriktionen unterliegen.[18]

4.4 Weitere Anforderungen

Die bisher geschilderten Probleme bezogen sich immer auf die Anforderung an ein System, in einem gegebenen Textkorpus Eigennamen richtig zu erkennen. Borthwick geht weiter, indem er fordert, ein NER-System müsse auch *portabel* sein in andere Sprachen und andere Domänen, und zwar möglichst ohne grössere Kosten (das heisst minimaler Programmieraufwand durch eine Computerlinguistin oder einen Computerlinguisten).[19]

Ein leistungsfähiges NER-System zeichne sich des Weiteren dadurch aus, dass es bei geringfügigen Änderungen von Regeln keinen oder nur kleinen Programmieraufwand brauche, um es den neuen Umständen anzupassen. Beispielsweise könne es sein, dass ein NER-System das Wort *Ford* in *Ford Taurus* nicht taggt, weil das System das Wort als Teil eines Produktenamens betrachtet und diese in einer spezifischen Anwendung nicht markiert werden sollen. Nun kann es aber sein, dass in einer anderen Anwendung *Ford* durchaus getaggt werden soll, weil es in diesem Fall als Referenz auf eine Firma betrachtet wird.[20]

[18] In [Kob99] wird ein System zur Personennamenerkennung des Deutschen dokumentiert, eingangs werden jedoch allgemeine Betrachtungen über Eigennamen angestellt. Natürlich wird nicht vergessen zu erwähnen, dass Eigennamen das morphographematische Merkmal der Grossschreibung besitzen, das im Deutschen jedoch allen Nomen angehört.

[19] Vgl.[Bor99], Seite vi.

[20] Vgl.[Bor99], Seite 3.

Kapitel 5

Methoden zur Erkennung von Eigennamen

Grundsätzlich lassen sich die Methoden zur Erkennung von Eigennamen in zwei Gruppen einteilen: in solche mit *regelbasiertem* und solche mit *statistisch basiertem* Ansatz. In diesem Kapitel werden elf NER-Systeme vorgestellt, sieben davon regelbasiert, vier statistisch. Allerdings ist diese Zweier-Aufteilung nicht eindeutig, einige Systeme wenden auch beide Methodenarten an.[1] Die Zuteilung dieser Systeme zur einen oder anderen Gruppe erfolgte nach deren Schwerpunktlegung.

Obwohl alle Systeme - ausgenommen PNF[2]- an der MUC-7 teilgenommen haben, sind die einzelnen System-Beschriebe, die anlässlich der Konferenz veröffentlicht wurden, sehr verschieden in ihrer Ausführlichkeit. Wenn sie zu wenig detailliert waren, wurde versucht, offene Fragen zu klären, indem nach weiterer Literatur derselben Autoren respektive über dasselbe System recherchiert wurde. Gelang dies, erhielt man manchmal auch zusätzliche nützliche Informationen, die das Verständnis erleichterten. Wenn es als sinnvoll erachtet wurde, diese Zusatzinformationen wiederzugeben, so wurde dies getan. Diese Methode, teilweise weiterführende Literatur hinzuzuziehen und davon nur ausgewählte Information hier anzufügen, geht allerdings auf Kosten der Konformität der System-Beschreibungen dieser Arbeit. So kommt es auch, dass am Ende eines Kapitels gelegentlich stichwortartige, gelegentlich ausführlichere Bemerkungen, manchmal die nährere Beleuchtung eines Aspekts oder nichts derartiges angefügt ist. Diese inkonsistenten Anmerkungen - oft in Form von Assoziationen - dienen als Vorüberlegungen zur systematischen Analyse in Kapitel 6.

Ziel von jeder Beschreibung ist die *Erklärung der Funktionsweise des Systems*. Wenn für nötig gehalten, werden auch Grundlagen erklärt - dies vor allem im Bereich der Statistik. Im Vorspann einer Beschreibung ist jeweils tabellenartig angegeben, aus welchen Quellen die Informationen stammen, welche Eigennamen respektive NEs das System erkennt und für welche Sprachen das System einsetzbar ist.

[1] In Kapitel 6 wird ersichtlich werden, welche dieser Systeme beide Methodenarten einsetzen und bei welchen es sich lohnt, ihren hybriden Charakter zu verfolgen.

[2] Vgl. Unterkapitel 5.1.1.

5.1 Regelbasierte Systeme

5.1.1 Proper Name Facility (PNF) von SPARSER

Quellen	[McD93]
Erkannte NEs/Eigennamen	Personennamen, Firmennamen, Ortsnamen. Ob weitere Eigennamen erkannt werden, ist unklar.
Sprachen	Englisch

Ein Klassiker in der Literatur über Eigennamen-Erkennung ist [McD93]. McDonald beschreibt darin nicht nur die Eigennamen-Erkennungs-Komponente *Proper Name Facility (PNF)* von *SPARSER* - ein System zum Verstehen von natürlicher Sprache[3]- sondern stellt wegweisende theoretische Überlegungen an, die von vielen nachfolgenden NER-Arbeiten aufgenommen wurden.

Obwohl bereits vor Veröffentlichung von [McD93] NER-Systeme gebaut wurden, die mit gleichen oder ähnlichen Methoden wie PNF respektive SPARSER arbeiten, ist McDonalds Aufsatz wichtig, weil darin erstmals die *Kernidee* der Methoden analysiert und ihr ein Fachbegriff verliehen wird.

a) Interne und externe Evidenz

In [McD93] wird erkannt, dass die Kernidee eines jeden NER-Systems das Ausnutzen von *interner* und *externer Evidenz* ist (engl. *internal and external evidence*). Die Begriffe *intern* und *extern* beziehen sich dabei auf die Einheit eines Eigennamens: interne Evidenz nutzt Hinweise *innerhalb* des Eigennamens aus, externe Evidenz den *Kontext* desselben.

Interne Evidenz ist dann gegeben, wenn Teile des Eigennamens, der im Normalfall aus einem oder mehreren aufeinanderfolgenden grossgeschriebenen Wörtern besteht, deutliche Hinweise auf seine Klasse geben. Das kann beispielsweise *Co.* oder *Ltd.* sein im Falle eines Firmennamens oder im Falle eines Personennamens ein bekannter Vorname oder ein initialer Buchstabe (zum Beispiel in *Georg W. Bush* deutet der initiale Buchstabe *W.* auf einen Personennamen hin). Interne Evidenz wird von vielen der beschriebenen Systeme unter anderem dadurch ausgenutzt, indem mit Listen gearbeitet wird: beispielsweise Listen mit Vornamen, anhand derer Personennamen erkannt werden, oder sogar Listen mit vollständigen Namen wie zum Beispiel Ortsnamenverzeichnisse.[4]

Externe Evidenz nützt den Kontext des Eigennamens aus. Hier wird von der Feststellung ausgegangen, dass Namen Mittel sind, um auf Individuen eines spezifischen Typs zu referieren (Personen, Kirchen, Musik-Gruppen etc.), die auch spezifische Eigenschaften besitzen und in Zusammenhang mit spezifischen Ereignissen vorkommen. Das Vorhanden-

[3] Vgl. [McD93]: McDonald nennt SPASER ein "natural language understanding system".

[4] SPARSER arbeitet ohne Eigennamenlisten, trotzdem ist diese Möglichkeit in [McD93] erwähnt.

sein solcher Eigenschaften oder Ereignisbeschriebe in syntaktischer Beziehung zu einem Eigennamen liefert Hinweise auf dessen Kategorie und wird externe Evidenz genannt. Operationalisiert wird das Ausnutzen externer Evidenz mit Hilfe von *kontextsensitiven Ersetzungsregeln* (engl. *rewrite rules*), wie beispielsweise

(1) `name -> location / __ "office"`

Diese Ersetzungsregel besagt, dass ein Name (`name`) dann ein Ortsname (`location`) ist, wenn auf ihn das Wort `office` folgt. Links des Slashs (`/`) steht der ausführbare Teil der Regel, rechts das Muster als Bedingung, wobei die Unterstriche als Stellvertreter für den Namen stehen.

b) PNF: Begrenzen, klassifizieren, aufzeichnen

Nach den theoretischen Erkenntnissen wird in [McD93] die PNF beschrieben. McDonald betont, dass die Integration von PNF ins SPARSER-Gesamtsystem essentiell ist - PNF würde sonst nicht die gleiche Leistung erbringen, da externe Evidenz nicht ausgenutzt werden könnte.

PNF wird dann aufgerufen, wenn beim Scannen des Inputtextes[5] ein grossgeschriebenes Wort auftaucht. Die erste PNF-Funktion *Begrenzen* (engl. *delimit*) liest so viele der folgenden Wörter ein, bis eines kleingeschrieben ist. Alle benachbarten grossgeschriebenen Wörter bilden eine Einheit, die als potenzieller, noch nicht klassifizierter Eigenname markiert wird.

Es folgt die *Klassifizierungs-Phase* (engl. *classify*) von PNF, wobei nun interne Evidenz ausgenutzt wird. Indikatoren wie eingebettete Referenzen auf Ortsnamen (zum Beispiel *Cambridge* in *Cambridge Savings Bank*), stilisierte Modifizierer von Personen (zum Beispiel *Jr.*, *Mr.*, *Sir* oder *Dr.*) oder Indikatoren[6] wie *Church, Bank* werden als Hinweise auf eine spezifische Eigennamenklasse genutzt.[7] Funktionale Wörter wie beispielsweise Artikel, die grossgeschrieben wurden, weil sie am Satzanfang stehen, werden als solche erkannt und aus der Menge der Eigennamen wieder ausgeschlossen. Kann aus Mangel an Indikatoren keine Klassifikation vorgenommen werden, so bleibt die Sequenz der grossgeschriebenen Wörter vorerst als *Name* markiert. Später wird SPARSER versuchen, mittels externer Evidenz (also mittels den oben erläuterten kontextsensitiven Ersetzungsregeln) die Klassifikation vorzunehmen. Wenn das auch nicht klappen sollte, werden statistische Heuristiken angewandt. Jedoch werden diese in [McD93] nicht weiter erläutert.

Die *Aufzeichnungs-Phase* (engl. *record*) von PNF hat mit einem *Diskursmodell* zu tun, das parallel zum Eigennamenerkennungs-Prozess erstellt wird. Allerdings wird der klassifizierte Eigenname nicht direkt in das Diskursmodell eingetragen, so dass er dort als Zeiger

[5] Der Vollständigkeit halber sei hier erwähnt, dass der Inputtext bereits vom SPARSER-Tokenizer bearbeitet wurde, wobei eine Chart erstellt wurde. Diese Chart ist nun der eigentliche Input für PNF.

[6] *Indikatoren* oder auch *Indikatorwörter* können ebenso als *Schlüsselwörter* bezeichnet werden.

[7] Die Liste der Indikatoren ist noch länger, hier soll nur das Prinzip verdeutlicht werden.

auf ein Objekt der Welt dienen würde. Vielmehr wird der Eigenname als *Namens-Objekt* (engl. *name object*) in ein *semantisches Modell* eingetragen. Der Grund für diesen "Umweg" liegt darin, dass in der Welt, die durch das Diskursmodell abgebildet werden soll, mehrere verschiedene Objekte - oft auch Objekte aus verschiedenen Klassen - oft den gleichen Namen haben. Diesem Umstand wird nun dadurch Rechnung getragen, dass einem Namens-Objekt, das im semantischen Modell nur ein Mal vorkommt, mehrere Objekte des Diskursmodells zugeordnet werden können. So können beispielsweise dem einzigartigen (engl. *unique*) Namens-Objekt *Arthur Andersen* zwei konkrete Individuen zugeordnet werden: Einmal eine Person namens *Arthur Andersen* und ein weiteres Mal eine Firma namens *Arthur Andersen Ltd.*

Zur weiteren Erkennung von Eigennamen wird nur das semantische Modell benötigt. Dieses ist so angelegt, dass nachfolgende Namen, die auf dasselbe Individuum referieren, aber etwas anders aussehen, auch erkannt werden können: Wird ein Name zum ersten Mal erkannt, werden verschiedene mögliche anderslautende Formen davon erstellt. Ein Beispiel: Wenn *Sumitomo Electric Industries, Ltd.* als Firmenname erkannt wird, werden mögliche Kurzformen davon wie beispielsweise *Sumitomo Electric* vorgemerkt. Falls dieser Bezeichner nachfolgend auftauchen würde, würde er als Firmenname desselben Individuums erkannt werden. Gleichzeitig muss PNF aber fähig sein, zwischen Namen zu unterscheiden, die gleiche Teile enthalten, aber nicht auf dasselbe Individuum referieren (zum Beispiel müssen die beiden Firmennamen *Sumitomo Electric Industries, Ltd.* und *Sumito Wiring Systems* unterschieden werden). Auch soll PNF sogar Namen von Tochterfirmen einer bereits erwähnten Firma ableiten können. Wie dies jedoch erreicht wird, geht aus [McD93] nicht hervor.

c) Bemerkungen

- PNF verwendet keine Listen.
- PNF arbeitet mit einer kleinen Domäne. Dies ist ein weiterer Grund, warum Zahlen über Performanz nicht mit denjenigen der MUC-Systeme vergleichbar sind.
- Das semantische Modell und das Diskursmodell wurden von nachfolgenden NER-Systemen kaum aufgegriffen.
- McDonald sagt, dass ein System, das nur NEs erkennen will, "nichts tauge"[8]. Allerdings wurden von anderen Forschern viele solche Systeme entwickelt.

[8] Vgl. [McD93], S. 29

5.1.2 BSEE von FACILE / CONCERTO

Quellen	[Bla98]; [Rin99]
Erkannte NEs/Eigennamen	Gemäss MUC-NE-Task Definition (vergleiche [Chi97a])
Sprachen	Deutsch, Englisch, Italienisch, Spanisch

a) NER-Systemarchitektur

Das NER-System, mit welchem die Forschergruppe *UMIST*[9] am MUC-7-NE-Task teilnahm, wurde als Komponente des grösseren Systems *FACILE* entwickelt, eines Text-Kategorisierungssystems. In einem anderem Projekt namens *CONCERTO* wurde das NER-System erweitert und als *Basic Semantic Element Extractor (BSEE)* bezeichnet. Im Folgenden soll das BSEE vorgestellt werden. Es besteht aus 3 Modulen (vergleiche Abbildung 5.1):

1. Grundlegende Vorverarbeitung (engl. *Basic Preprocessing*)
2. Nachschlagen in der Datenbank (engl. *Database Lookup*)
3. NE-Analysator (engl. *Named Entity Analyser*)

1. Grundlegende Vorverarbeitung

In der *grundlegenden Vorverarbeitung* wird der zu markierende Input-Text in Tokens aufgesplittet, wobei mehrwortige NEs noch nicht wie gewünscht *ein* Token sind. Pro Token wird eine *Kante* (engl. *Edge*) in einer *Chart* erstellt, die durch die grundlegende Vorverarbeitung initiiert und mittels den beiden folgenden Modulen vervollständigt wird (vergleiche Abbildung 5.1). Ein Beispiel einer Kante für das Token *Reliance* zeigt die Tabelle 5.1.

Die Werte dieser Kante werden mit Hilfe verschiedener Tools ermittelt, zum Beispiel *Text Zoner* (liefert Wert für *Zone*), Morphosyntaktischer Tagger (liefert Wert für *Syn*) und Morphologischer Analysator. Letzterer liefert teilweise mehrere unterschiedliche Werte, was durch die Aufteilung in *Good-morph* und *Other-morph* zum Ausdruck kommt.[11]

2. Nachschlagen in der Datenbank

In der Datenbank (engl. *Database*) sind 2 Gruppen von Ausdrücken gespeichert: Bereits bekannte *Eigennamen* (*Proper Nouns*) und so genannte *Indikatorwörter* (engl. *Clue Words*).

[9] Nähere Angaben zu den Forschergruppen finden sich im Anhang B.

[10] Die Abbildung ist [Rin99] entnommen.

[11] *Good-morph* enthält denjenigen Wert der morphologischen Analyse, der mit demjenigen des Taggers übereinstimmt, *Other-morph* die anderen. Steht in *Other-morph* der Wert NIL, so ergibt die morphologischen Analyse keine anderen Werte.

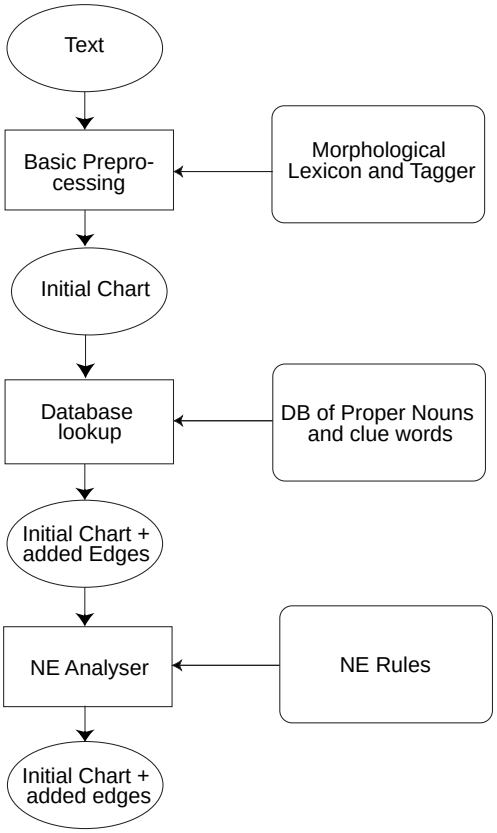

Abbildung 5.1: Architektur des Basic Semantic Element Extractor (BSEE)[10]

Wenn beim *Nachschlagen in der Datenbank* festgestellt wird, dass ein Token ein Eigenname ist, so erhält es in seiner Kante als *Sem*-Wert den entsprechenden semantischen Wert, zum Beispiel `PER` für Personenname oder `ORG` als Name einer Firma, Organisation etc. Wenn im Input-Text mehrere Worte einen Eigennamen bilden, der als *ein* Token in der Datenbank vorkommt, so werden die ursprünglichen Tokens durch dieses ersetzt.

Ein Indikatorwort ist in der Regel ein Wort, das darauf hinweist, dass *vor* oder *nach* ihm eine NE stehen muss. Wird im Input-Text beispielsweise ein Indikatorwort gefunden, das darauf hinweist, dass das nachfolgende grossgeschriebene (ein- oder mehrwortige) Token ein Personenname ist (zum Beispiel das Indikatorwort *Mr.*), so erhält dieses Indikatorwort in seiner Kante als *Sem*-Wert einen speziellen Wert, der ihn als Indikatorwort für einen Personennamen auszeichnet. Als Konvention wurde festgelegt, das sich der *Sem*-Wert für ein Indikatorwort aus dem Zeichen `^` und dem *Sem*-Wert der bezeichneten Art von Eigennamen zusammensetzen soll - für das Indikatorwort *Mr.* wäre dies somit `^PER`. Dass das folgende Token dann den entsprechenden semantischen Wert - in unserem Beispiel also `PER` - erhält, dafür sorgen bestimmte Regeln im *NE-Analysator* (siehe folgenden Abschnitt).

Feld	**Wert**	**Bemerkungen**
Start	28	Gibt Position an, wo das Token beginnt
End	36	Gibt Position an, wo das Token endet
Zone	Text	Zone, in der das Token steht
Sep	040	Separator (meist White-space)
Orth	C	C=capitalised, L=lower case, A=all capitalised
Token	*Reliance*	So sieht das Token im Input-Text aus
Norm	*reliance*	Die "normalisierte" Form des Tokens
Syn	NP	Präferiertes syntaktisches Tag
Sem	()	Noch kein Wert; dieser wird erst entweder durch das *Nachschlagen in der Datenbank* oder durch Regelanwendung im *NE-Analysator* hinzugegefügt. Mögliche Werte sind beispielsweise `ORG` für Organisationsname, `PER` für Personenname etc. Diese semantischen Werte (Tags) sind diejenigen, die man schlussendlich finden will!
Good-morph	NP	Morphologie des präferierten SYN-Wertes
Other-morph	NIL	Morphologie des nicht-präferierten SYN-Wertes

Tabelle 5.1: Kante für das Token *Reliance* (dt. *Vertrauen*)

3. NE-Analysator

Der NE-Analysator fügt für NEs neue Kanten in die Chart ein. Dazu werden sprachspezifische kontextsensitive Regeln (engl. *NE Rules*) benutzt. Ein Beispiel:

(2) `[syn=NP, sem=PER] (0.9) => [sem=^PER] \ [orth=C]+ /`

Erläuterung: Der Pfeil `=>` trennt linke und rechte Seite der Regel voneinander. Links stehen die Werte, die für ein Token - mit einer Wahrscheinlichkeit von 0.9[12] - in die Kante eingetragen werden müssen, wenn die Bedingungen rechts zutreffen. Wichtig auf der rechten Seite sind die Slashs `\` und `/`, die bedeuten, dass es sich beim zu bestimmenden Ausdruck um dasjenige Token handelt, das dazwischen steht. Was davor oder dahinter steht, ist Kontext. In Regel (2) hat ein ein- oder mehrwortiges Token[13] mit einer Wahrscheinlichkeit von

[12]Zum Zustandekommen dieses Wahrscheinlichkeitswerts vgl. Abschnitt *c) Kommentar*, Seite 33.

[13]Dass das Token aus einem oder mehreren Worten bestehen kann, wird durch das Plus-Zeichen `+` angezeigt.

0.9 den syntaktischen Wert `NP` und den semantischen Wert `PER`, wenn es grossgeschrieben ist (`[orth=C]`) und davor ein Token mit dem semantischen Wert `[sem=^PER]` steht. Angewandt auf das oben genannte Beispiel, wo dem Indikatorwort *Mr.* beim Nachschlagen in der Datenbank der semantische Wert `^PER` zugewiesen wurde, bedeutet dies Folgendes: Trifft der NE-Analysator beispielsweise auf die Wortfolge

(3) *Mr. Charly Brown is wise.*

so wird *Charly Brown* der syntaktische Wert `NP` und der semantische Wert `PER` zugewiesen, weil diese beiden Wörter grossgeschrieben sind und unmittelbar dem Indikatorwort *Mr.* folgen.

b) Koreferenz-Auflösung

Das Auflösen von *Koreferenzen* ist eine wirkungsvolle Methode, weitere Eigennamen zu finden, die in anderer Ausprägung schon einmal gefunden wurden; also wenn beispielsweise *Bill Clinton* bereits als `PER` erkannt wurde, kann *Clinton* auch als `PER` erkannt werden, wenn kein Indikatorwort darauf hinweist. Ein Beispiel:

(4) *The president of the USA, Bill Clinton, visited Europe. Clinton left the USA ...*

Angenommen, *president* ist als Indikatorwort in der Datenbank mit dem semantischen Wert `^PER` versehen und *USA* mit `LOC` (für *location*). Die erste Regel, die *Bill Clinton* als `PER` erkennen soll, lautet:

(5)
```
[sem=PER, firstname=_F, surname=_S] =>
  [sem=^PER] [token="of"] [token="the"] [sem=LOC]
  \   [orth=C, token=_F], [orth=C, token=_S] /
```

Hier wird *Bill Clinton* nicht nur als Personenname (`sem=PER`) erkannt, sondern *Bill* mittels Unifikation von `_F` der Variablen `firstname` und *Clinton* mittels Unifikation von `_S` der Variablen `surname` zugeordnet. Wozu, wird in der nächsten Regel, einer Koreferenzregel, klar:

(6)
```
[sem=PER] => \  [orth=C, token=_S] /
    >>  [sem=PER, surname=_S]
```

Dank dieser Regel wird in (4) das zweite Vorkommen von *Clinton* (ohne *Bill* davor!) ebenfalls als Personenname erkannt, auch wenn kein Indikatorwort davor steht. Denn Regel (6) besagt, dass ein grossgeschriebenes Wort dann ein Personenname ist, wenn es koreferiert (konkret: `_S` wird unifiziert) mit einem zuvor vorgekommenen Teil eines Tokens (hier: mit dem Nachnamen *Clinton*, der in der Variablen `surname` gespeichert ist). Das Zeichen `>>` bedeutet Koreferenz.

c) Bemerkungen

- Aus den Forschungsberichten geht nicht klar hervor, wozu die Gewichtung dient (vergleiche die Wahrscheinlichkeitsangabe von 0.9 in Abschnitt *3. NE-Analysator*). Zum einen wird nicht allen Regeln eine Wahrscheinlichkeit zugeordnet, zum anderen wird nicht genauer ausgeführt, wozu und wie die Gewichtungen verwendet werden. Eine Nachfrage bei F. Rinaldi (Mitautor von [Rin99]) ergab, dass die Idee der Gewichtung tatsächlich nur angedacht, aber nicht konsequent umgesetzt wurde. Auch wurden die Werte willkürlich festgelegt. Ihr Zweck soll an folgendem Beispiel erläutert werden: Liegt im zu markierenden Inputtext der Ausdruck *Mr. Miller Ltd.* vor, so weist *Mr.* als Indikatorwort darauf hin, dass *Miller* ein Personenname ist. Aber gleichzeitig zeigt das Indikatorwort *Ltd.* an, dass der davor stehende Ausdruck ein Organisationsname sein muss. Dieser Widerspruch sollte durch die Gewichtung der konkurrierenden Regeln aufgelöst werden, indem die gewichtigere Wertzuweisung vorgezogen wird.

- Die Performanz des BSEE ist unmittelbar abhängig von der Anzahl und Qualität sowohl der beiden Listen[14] in der Datenbank als auch der Regeln[15] im NE-Analysator. Um die Performanz zu beurteilen, müssten diese Ressourcen untersucht werden. Nebst dem Umstand, dass es schwierig ist, in diese nur intern verbreiteten Daten Einsicht zu erhalten, wäre auch das Vergleichen mit den Ressourcen anderer gleich oder ähnlich arbeitender Systeme ein kompliziertes Unterfangen, da diese Ressourcen sehr umfangreich zu sein scheinen. Dazu kommt die Schwierigkeit der verschiedenen Versionen der Listen, die mit den verschiedenen Versionen eines Systems einhergehen.

- Sowohl die Liste der Eigennamen als auch die Liste der Indikatorwörter wurden manuell erstellt. Allerdings bediente man sich bei der Erstellung der Eigennamenlisten bereits vorhandener Listen. Eine gute Quelle dafür war das Internet, wo beispielsweise auf Seiten von Fluggesellschaften Länderlisten gefunden werden konnten. Interessant ist der Umstand, dass die Forschergruppe eine eigene Liste mit Ortsnamen erstellt hat, obwohl die Organisatoren der MUC eine umfangreiche Ortsnamenliste (engl. *Gazetteer*) an die Teilnehmer abgegeben hatten. Der Grund für den Entscheid, die von der MUC-Organisation erhaltene Ortsnamenliste nicht zu verwenden, liegt darin, dass diese viel zu gross war; beinah jede noch so kleine Ortschaft der Welt war vorzufinden. Nun sind sehr viele Ortsnamen auch Wörter des allgemeinsprachlichen Lexikons und mit der MUC-Ortsnamenliste wurden viele allgemeinsprachliche Wörter fälschlicherweise als Ortsnamen markiert. Die Präzision verschlechterte sich derart, dass man sich entschloss, eine eigene Ortsnamensliste zu erstellen.

- Die Regeln im NE-Analysator hat die Forschergruppe gänzlich von Hand - ohne Beiziehen irgendwelcher Quellen - erstellt.[16]

[14] Vgl. Nachschlagen in der Datenbank: Liste mit Indikatorwörtern und Liste der bereits bekannten Eigennamen.

[15] Die "normalen" Regeln und die Koreferenzregeln.

[16] Wie die Ressourcen entstanden sind, geht nicht aus den Forschungsberichten hervor. Dies konnte dank

5.1.3 LaSIE

Quellen	[Hum98]; [Hum00]; [Gai95]; [Tak96]
Erkannte NEs/Eigennamen	Gemäss MUC-NE-Task Definition (vergleiche [Chi97a])
Sprachen	Englisch

Vorbemerkung: [Gai95] und [Tak96] beschreiben *LaSIE 1.0* (MUC 6), [Hum98] und [Hum00] beschreiben das neuere *LaSIE 2.1* (MUC 7). Da die NER in den älteren Berichten verständlicher beschrieben ist, wird hier vorwiegend [Gai95] und [Tak96] zusammengefasst. Abweichungen von LaSIE 2.1 zu LaSIE 1.0 werden im Abschnitt *Was ist neu bei LaSIE 2.1?* (Seite 38) beschrieben.

a) Systemarchitektur

LaSIE ist ein umfassendes IE-System, das alle MUC-Tasks erfüllt. Es besteht aus zahlreichen Modulen und die NER findet nicht nur an einem Ort statt, sondern verteilt über drei verschiedene Module:

1. Lexikalische Vorverarbeitung
2. (NE-)Parsing
3. Diskurs-Interpretation

1. Lexikalische Vorverarbeitung

Bei der *lexikalischen Vorverarbeitung* (engl. *Lexical Preprocessing*) werden mehrere Schritte ausgeführt:

i. Tokenisierung
ii. Wortarten-Tagging
iii. Morphologische Analyse
iv. NE-Phrasen-Tagging
v. Initialisierung der Chart (engl. *Chart Seeding*)

Anfrage bei F. Rinaldi in Erfahrung gebracht werden (Mitautor von [Rin99]).

ad ii. Wortarten-Tagging: Das *Wortarten-Tagging* wird vom einem leicht angepassten Brill-Tagger[17] durchgeführt. Das Tagset enthält für Eigennamen zwei Tags: `NNP` für singuläre Eigennamen und `NNPS` für plurale. Ein Wort wird als Eigenname auf folgende Weise getagt: Entweder wenn das Wort im Tagger-Lexikon[18] als Eigenname enthalten ist oder wenn es nicht enthalten, aber im Text grossgeschrieben ist. Letzterer Schritt bewirkt, dass unbekannte, grossgeschriebene Wörter automatisch als (nicht weiter spezifizierte und somit nicht klassifizierte) Eigennamen getagt werden.

ad iv. NE-Phrasen-Tagging: Ziel ist es, NE-Phrasen[19] zu identifizieren und zu taggen (das heisst auch zu klassifizieren). Dazu werden zweierlei Arten von Listen verwendet: Listen mit *Namen*[20]*, Währungen und Zeit-Ausdrücken*[21] und Listen mit *Auslöser-Wörtern* (engl. *Triggerwords*).

Unter den Namenslisten finden sich neben Listen von Organisations-, Orts- und Personennamen auch Listen mit *Titeln* (*Mr.*, *President*) und *Firmenbezeichnern* (engl. *Company Designators*: *Co.*, *PLC*, *Ltd.*).

Auslöser-Wörter sind Wörter, die als Teil von mehrwortigen Eigennamen vorkommen. Beispiele: *Agency*, *Ministry* (in Namen staatlicher Institutionen, also Organisationsnamen), *Airline*, *Association* (in Namen von Firmen, also Organisationsnamen), *Gulf*, *Mountain* (in Ortsnamen).[22]

Die mittels NE-Phrasen-Tagging gefundenen Eigennamen und Auslöser-Wörter werden speziell getagt ("as result of the list lookup stage"[23]) und später beim NE-Parsing in den NE-Regeln weiterverwendet (vergleiche anschliessenden Abschnitt *2. (NE-)Parsing*).

Warum die Listen in zwei und nicht in drei Gruppen eingeteilt werden, ist nicht verständlich. Aus den Listen mit Titeln und Firmenbezeichnern könnte eine eigene Gruppe gemacht werden, da sie nicht gleich behandelt werden können wie die Listen der eigentlichen Eigennamen.[24] Wahrscheinlich ist auch die Forschergruppe selbst zum Schluss gekommen, dass ihre Einteilung nicht einsichtig ist, und hat deshalb in LaSIE 2.1 überhaupt keine Gruppen mehr gebildet.[25]

---

[17]Folgende Anpassungen wurden vorgenommen: Einführen neuer Tags für Datumsangaben, SGML-Markup, und für mehrere Satzzeichen, die vom originalen Brill-Tagger gleich behandelt würden. Zusätzlich wurden der Regelbasis des Taggers mehrere Lexikon- und Kontextregeln hinzugefügt. Vgl. [Gai95], Seite 3.

[18]Das Tagger-Lexikon ist Bestandteil des Brill-Taggers, das wie der Tagger selbst leicht angepasst wurde (z. B. wurden Einträge hinzugefügt).

[19]Unter NE-Phrasen fallen sowohl die ein- als auch die mehrwortigen Eigennamen.

[20]Im Original - vgl. [Tak96]- als *Lists of names* bezeichnet. Insgesamt enthalten diese Listen etwa 5'500 Einträge.

[21]Da hier nur die Erkennung von Eigennamen von Interesse ist, wird die Methode zur Erkennung anderer NEs bei LaSIE ab jetzt nur dann erwähnt, wenn es für den Zusammenhang notwendig ist.

[22]Die Listen der Auslöser-Wörter enthalten insgesamt 153 Einträge.

[23][Gai95].

[24]Vgl. bei BSEE von FACILE / CONCERTO (Unterkapitel 5.1.2), wo Titel und Firmenbezeichner als *Indikatorwörter* (engl. *Clue Words*) bezeichnet und separat behandelt werden.

[25]Der Vollständigkeit halber sei noch erwähnt, dass die in LaSIE 1.0 mit *Triggerwords* (dt. *Auslöser-Wörter*) bezeichneten Wörter in LaSIE 2.1 *Keywords* heissen.

2. (NE-)Parsing

Das Parsing wird in zwei Schritten ausgeführt: Ein *NE-Parsing* und ein *Satz-Parsing* (engl. *Sentence-Parsing*). Hier interessiert das erstere. Ziel des NE-Parsings ist, wiederum NEs zu identifizieren, damit sie beim anschliessenden Satz-Parsing als Einheit behandelt werden. Für das NE-Parsing wird eine NE-Grammatik verwendet. Diese bildet ein Untermenge von NP-Regeln. Alle Regeln wurden von Hand produziert. In LaSIE 1.0 sind es 206 Regeln (94 für Organisationen, 54 für Personen, 11 für Orte, 18 für Daten/Zeiten, 29 für Währungs-/Prozentangaben). Bsp. für NE-Grammatik-Regeln:

(7)
```
NP --> ORGAN_NP
ORGAN_NP --> LIST_LOC_NP NAMES_NP CDG_NP
ORGAN_NP --> LIST_ORGAN_NP NAMES_NP CDG_NP
ORGAN_NP --> NAMES_NP "&" NAMES_NP
NAMES_NP --> NNP NAMES_NP
NAMES_NP --> NNP PUNC(_) NNP
NAMES_NP --> NNP
```

Die Nicht-Terminalen `LIST_LOC_NP`, `LIST_ORGAN_NP` und `CDG_NP`[26] sind Tags, die beim NE-Phrasen-Tagging zugewiesen wurden, das Nicht-Terminal `NNP` beim Wortarten-Tagging (vergleiche vorigen Abschnitt *1. Lexikalische Vorverarbeitung*). `NAMES_NP` - das Tag für einen Personennamen - wird erst in der NE-Grammatik eingeführt. Beispiel: Die Regel

(8)
```
ORGAN_NP --> NAMES_NP "&" NAMES_NP
```

kann bewirken, dass *Marks & Spencer* oder *American Telephone & Telegraph* als `ORGAN_NP` - also als Tag für einen Organisationsnamen - erkannt wird.
Etwa die Hälfte dieser Regeln dienen dem Erkennen von Organisationen-Namen, da in ihnen andere NPs - beispielsweise Personennamen, Ortsnamen, Nomen des allgemeinsprachlichen Lexikons - enthalten sind. Viele dienen auch dem Erkennen von Personennamen. Dazu werden beispielsweise die Titel verwendet (vergleiche Abschnitt 1. Lexikalische Vorverarbeitung). Aber auch lexikalische Wörter wie *Jr.* oder *de* können für Personennamen-Regeln eingesetzt werden. Der Erkennung von Ortsnamen dienen die Regeln kaum; Ortsnamen sind meist bereits beim NE-Phrasen-Tagging vollständig erkannt worden.

3. Diskurs-Interpretation

Bei der *Diskurs-Interpretation* (engl. *Discourse Interpretation*) werden keine neuen Eigennamen mehr erkannt, nur noch diejenigen klassifiziert, bei denen das bisher nicht gelungen ist. Zwei Methoden werden verwendet, die beide im Gegensatz zu den vorangegangenen *externe Evidenz*[27] ausnutzen:

[26] `CDG_NP` wird wahrscheinlich *Company Designator* (dt. *Firmenbezeichner*) bedeuten; in [Tak96] wird es nicht aufgelöst.

[27] Zum Begriff der *externen Evidenz* vgl. [McD93].

i. Koreferenzen auflösen (wird auch als *Name Matching* bezeichnet)

ii. Klassifizieren durch Inferieren (engl. *Semantic Type Inference*)

ad i. Koreferenzen auflösen (Name Matching): Ziel ist, alternative Formen von Eigennamen zu finden. Das ist besonders nützlich bei Namen von Organisationen, gelegentlich auch bei Personennamen. Ein Beispiel: Man möchte nebst einem als Organisationsnamen erkannten *Ford Motor Co.* auch das zwar als Eigenname erkannte, aber noch nicht klassifizierte *Ford* finden. Zu diesem Zweck werden heuristische Regeln verwendet, die folgendermassen aussehen können:

- Wenn der `Name2` aus einer Subfolge von Wörtern von `Name1` besteht, dann matchen `Name2` und `Name1`, und `Name2` wird gleich wie `Name1` klassifiziert. Bsp.: *American Airlines Co.* und *American Airlines*.

- Wenn `Name1` ein Personenname ist und `Name2` ist entweder der Vorname, der Nachname oder beides, dann matchen `Name2` und `Name1`, und `Name2` wird auch als Personenname klassifiziert. Bsp.: *John J. Major Jr.* und *John Major*. 31 solcher heuristischer Regeln dienen dem Matchen von Organisationsnamen, 11 dem von Personen- und 3 von Ortsnamen.

ad ii. Klassifizieren durch Inferieren: Steht ein noch nicht klassifizierter Eigenname in einer bestimmten Relation zu einem Wort mit bestimmter semantischer Information, können Rückschlüsse (Inferenzen) gezogen werden auf den Typ (Klasse) des Eigennamens. Beispiele:

- *Nomen-Nomen Qualifikation*: Wenn ein unklassifizierter Eigenname ein "organisations-verwandtes Ding"[28] qualifiziert, dann wird der Eigenname als Organisationsname klassifiziert. Beispiel: In *Ericsson stocks* ist *stock* semantisch als "organisationsverwandtes Ding" klassifiziert. Deshalb wird *Ericsson* als Organisationsname markiert.

- *Apposition*: Wenn einem unklassifizierten Eigennamen ein bekannter Ortsname als Apposition beigefügt ist, so wird der Eigenname ebenfalls als Ortsname klassifiziert. Beispiel: Weiss man bei *Fort Lauderdale, Fla.*, dass *Fla* ein Orstname ist, so wird *Fort Lauderdale* ebenfalls als Orstname klassifiziert.

[28] In [Tak96] ist von einem "organisation-related thing" die Rede.

b) Was ist neu bei LaSIE 2.1?

Ad 1. Lexikalische Vorverarbeitung: Was in LaSIE 1.0 als *NE-Phrasen-Tagging* bezeichnet wird, ist in LaSIE 2.1 ein *Nachschlagen in Listen* (engl. *Gazetteer Lookup*). Neu daran ist vor allem die enorme Erweiterung der Anzahl Listen (LaSIE 2.1 arbeitet mit 55 Listen mit total 23'000 Einträgen). Auch wurde in LaSIE 2.1 die Position dieses Moduls im Gesamtsystem verändert: Das Nachschlagen in Listen geschieht gleich nach der Tokenisierung, also vor dem Taggen. Da die Gründe für die Positionsänderung nicht mit dem NE-Task im Zusammenhang stehen, hat sie auf die NER auch keinen Einfluss.

Ad 2. NE-Parsing: [Hum98] gibt an, dass in LaSIE 2.1 die Grammatik-Entwicklungsumgebung erweitert und die Grammatik komplett neu geschrieben und vergrössert wurde, welche nun eine substanziell andere Philosophie zeige: "[...] and a completely rewritten and extended grammar which now reflects a substantially different philosophy."[29] Die neue Grammatik wurde in 17 Untergrammatiken unterteilt, wovon 10 NE-Grammatiken sind. Diese sind nach Klassen respektive Unterklassen aufgeteilt. Es gibt beispielsweise `general_ne_rules`, `person_ne_rules`, `location_ne_rules`, `space_ne_rules`, `organ_ne_rules`, `money_ne_rules`, `time_ne_rules`. Kurz darauf stellt sich jedoch heraus, dass sich grundlegend doch nicht so viel verändert hat, wenn es heisst: "While significantly rewritten since MUC-6 the basic philosophy here is the same [...]: patterns are detected in the texts and manually added to the grammar."[30]

Ad 3. Diskurs-Interpretation: Da in Bezug auf das *Auflösen von Koreferenzen* keine Unterschiede festgestellt werden konnten, ist anzunehmen, dass es in LaSIE 2.1 immer noch gleich ist. Betreffend *Klassifizieren durch Inferieren* schrieb die Forschergruppe in [Tak96], dass sie bis dahin diese Technik erst spärlich eingesetzt hätten und viel Raum für Erweiterung bestünde. Da in den Berichten über LaSIE 2.1 nichts mehr über Klassifizieren durch Inferieren geschrieben wurde, liegt die Vermutung nahe, dass es nicht mehr verwendet wird.

c) Resultate und Evaluation

Pro *Klasse* (Organisations-, Personen-, Ortsnamen und Zeitausdrücke) wurde je ca. 90 Prozent erkannt (*Ausbeute*, engl. *Recall*) mit einer *Präzision* (engl. *Precision*) von 89 bis 97 Prozent. Interessant ist die Untersuchung, welches Modul wieviel zur NER beigetragen hat. Auf alle Klassen angewandt ergibt sich Tabelle 5.2. Die Zahlen sind Prozentangaben, die als Summen zu verstehen sind: In einer Zeile ist jeweils die Summe abzulesen von der Leistung der bisher eingesetzten Module und der Leistung des aktuellen Moduls .

In [Tak96] wurde zudem *für jede Eigennamenklasse* untersucht, wieviel die einzelnen Module zur Namenserkennung beitragen. So konnte festgestellt werden, dass sich das

[29] [Hum98].
[30] [Hum98].

Modul	Ausbeute (A)	Präzision (P)	A und P
1. Lexikalische Vorverarbeitung	49	89	63
2. (NE-)Parsing	79	94	86
3(1). Auflösen von Koreferenzen	89	94	91
3(2). Klassifizieren durch Inferieren	91	93	92

Tabelle 5.2: LaSIE 1.0: Aufsummierte Leistung der Module

Ausnutzen von Externer Evidenz mittels Diskurs-Interpretation vor allem bei den Organisationennamen und, einiges weniger, bei den Personennamen lohnt. Da das Textkorpus eine Zeitung mit vielen News ist, wo naturgemäss viele Organisationsnamen vorkommen, lohnen sich also das Auflösen von Koreferenzen und das Klassifizieren durch Inferieren.

d) Bemerkungen

- LaSIE ist ein komplexes System, das alle fünf MUC-7-Tasks[31] erfüllt. Der Aufbau aus vielen Modulen, die meist nur Teilschritte ausführen, macht es schwierig, die Erfüllung eines Tasks - hier ist es der NE-Task - zu verfolgen. Dazu kommt die erschwerende Tatsache, dass LaSIE 1.0 und LaSIE 2.1 architektonisch verschieden sind und LaSIE 2.1 an verschiedenen Orten so verändert wurde, dass sich dies auf den NE-Task ausgewirkt hat.
- LaSIE 1.0 benutzt bereits umfangreiche Listen, LaSIE 2.1 noch viel mehr.
- Lexikalische Vorverarbeitung und (NE-)Parsing verwenden *Interne Evidenz*, Diskurs-Interpretation nutzt *Externe Evidenz* aus.
- Obwohl in den Berichten erwähnt, wird manchmal nicht ganz klar, welche Listen von Hand, welche halbautomatisch erstellt wurden. Zudem wird nicht deutlich, was halbautomatische Erstellung konkret bedeutet. Die Grammatik für das NE-Parsing und die Heuristiken zur Diskurs-Interpretation scheinen, da nichts anderes erwähnt wird, gänzlich von Hand entwickelt worden zu sein.

[31] Vgl. Unterkapitel 3.3.

5.1.4 LT TTT

Quellen	[Mik98]; [Mik99]
Erkannte NEs/Eigennamen	Gemäss MUC-NE-Task Definition (vergleiche [Chi97a])
Sprachen	Englisch

Die *Language Technology Group* der Universität Edinburgh hat für die Teilnahme an der MUC ihr *LT TTT*-System verwendet, ein Standard-Tool, das speziell für den NE-Task angepasst und erweitert wurde. LT steht dabei für *Language Technology* und TTT für *Text Tokenization Tool*. Das System ermöglicht nebst der klassischen Tokenisierung[32] auch die Kategorisierung der Tokens, beispielsweise in Klassen von Wortarten oder von Eigennamen.

Das LT TTT-System besteht aus verschiedenen Modulen, wiederum Tools genannt, die sehr flexibel eingesetzt werden können. Flexibel heisst, dass die mit einzubeziehenden Ressourcen (Grammatiken, Listen) je nach zu bearbeitendem Text variiert werden können. Auch ist die Reihenfolge, in der die Tools eingesetzt werden, je nach Verwendungszweck veränderbar. Im Folgenden wird erklärt, aus welchen Tools das LT TTT-System besteht und wie es zur NER eingesetzt wird.

a) Die LTG-Tools

Jedes Tool erhält als Input einen XML-formatierten Text, verändert bei der Verarbeitung das Markup und liefert den Output ebenfalls im XML-Format. Die Tools bilden so eine Kaskade. Beim Aufruf eines jeden Tools wird angegeben, auf welchen Inputtext - respektive sogar Textstelle - das Tool angewandt wird und welche Ressourcen zur Verarbeitung beigezogen werden.

Nachstehend werden die wichtigsten Tools des LT TTT-Systems kurz erläutert, wobei der Schwerpunkt auf das Kern-Tool *fsgmatch* gelegt wird.

1. Tool: lttok - ein Tokenisierungs-Tool

lttok ist ein Tokenisierungs-Tool, das mit Hilfe von Grammatikregeln arbeitet. Mehrwortige Ausdrücke werden allerdings noch nicht als ein Token erkannt und Satzendpunkte nicht von Abkürzungs- oder anderen Punkten unterschieden.

2. Tool: ltstop - löst Ambiguitäten bei Punkten auf

Zur Disambiguierung von Punkten verwendet *ltstop* ein Modell, das mit Maximaler Entro-

[32] Als *klassische Tokenisierung* wird hier die Aufteilung eines Textes in Tokens bezeichnet.

pie arbeitet (*Maximum-Entropy-Modell*)[33]. Im Spezialfall, dass ein Abkürzungspunkt gleichzeitig ein Satzendepunkt ist, wird ein "virtueller" Satzendpunkt eingefügt.

3. Tool: ltpos - Wortarten-Tagger

Der Wortarten-Tagger *ltpos* arbeitet mit statistischen Methoden (*Hidden-Markov-Modell*[34], *Trigramm-Maximum-Entropy-Modell*[35]). Speziell daran ist ein Modul, das mit Wörtern umgehen kann, die nicht im Lexikon sind. Da Eigennamen in gewöhnlichen Lexika kaum vorkommen, ist dieses Modul zur Namenserkennung äusserst nützlich.[36]

Für die Eigennamen-Erkennung wurde das *ltpos*-Standard-Tool erweitert. Beispielsweise wurden Funktionen eingefügt, die angeben, ob ein grossgeschriebenes Wort auch als kleingeschriebenes im Lexikon oder im Text vorkommt.[37] Neben den syntaktischen Tags werden gewissen zur Eigennamen-Erkennung nützlichen Tokens auch semantische Tags hinzugefügt, zum Beispiel Wörtern, die auf *-yst* und *-ist* enden (*analyst*, *geologist*). Solchermassen getaggte Tokens werden von den Grammatiken für *fsgmatch* zur Eigennamen-Erkennung genutzt.

4. Tool: fsgmatch - das Kern-Tool bei der Eigennamen-Erkennung

Das Tool *fsgmatch* ist ein SGML-Transduktor. Es nimmt gewisse Typen von SGML-Elementen und packt sie in grössere SGML-Elemente. Es ist dasjenige Tool, das beispielsweise eine Zahl mit einem Punkt dahinter, worauf ein Monatsname und eine vierstellige Zahl folgt, zum XML-Element <`timex`> des Typs `date` zusammennimmt und taggt.[38]

Bei *fsgmatch* kommt die Eigenschaft, dass je nach Verwendungszweck verschiedene Grammatiken aufgerufen werden können, besonders zum Tragen. Je nachdem, welche NE erkannt werden soll, wird ein anderes Regelset beigezogen.

[33]Was ein *Maximum-Entropy-Modell* ist und wie es funktioniert, ist in 5.2.3 ausführlich erklärt. An dieser Stelle sei lediglich erwähnt, dass das Modell auf von Hand getaggten Texten trainiert wurde.

[34]Die Funktionsweise eines *Hidden-Markov-Modells* wird in 5.2.2 erläutert.

[35]Vgl. Unterkapitel 5.2.3.

[36]Wie dieses Modul zur Bearbeitung unbekannter Wörter funktioniert, wird in [Mik98] und in [Mik99] nicht erklärt. Es wird lediglich ein Literaturhinweis auf [Mik97] gegeben. Wer sich eingehender für dieses Modul interessiert, sei auf diese letztere Literaturangabe verwiesen.

[37]Wozu diese Erweiterung dient, ist in [Mik98] und in [Mik99] nicht ausgeführt. Vermutlich sollen damit Tokens, die beispielsweise nur deshalb grossgeschrieben sind, weil sie am Satzanfang stehen, vor falscher Markierung als Eigennamen geschützt werden.

[38]Was `timex` und `date` und die im Folgenden vorkommenden Begriffe `numex` und `enamex` bedeuten, ist aus Anhang A ersichtlich.

b) 5-stufige Eigennamen-Erkennung

Um `numex`- und `timex`-Entitäten zu erkennen, reichen die bisher vorgestellten Tools aus. Die Erkennung von `enamex`-Entitäten, also von Eigennamen, ist komplizierter und verlangt eine komplexere Vorgehensweise: Sie wird in 5 Stufen durchgeführt, wobei sich die Anwendung des regelbasierten *fsgmatch* abwechselt mit statistisch basierten Methoden. Die Stufen heissen im Einzelnen:

1. Stufe: Zuverlässige Regeln (engl. *Sure-fire (transduction) Rules*)
2. Stufe: Partieller Match 1 (engl. *Partial Match 1*)
3. Stufe: Gelockerte Regeln (engl. *Rule Relaxation*)
4. Stufe: Partieller Match 2 (engl. *Partial Match 2*)
5. Stufe: Titel-Zuweisung (engl. *Titel Assignment*)

Jede Stufe nutzt diejenigen Informationen, die auf der vorangegangenen Stufe ermittelt wurden. In Stufe 1 und 3 wird *fsgmatch* angewandt, in den anderen wird statistisch gearbeitet. Im Folgenden sei das 5-stufige Vorgehen zur Eigennamen-Erkennung genauer erläutert.

Stufe 1: Zuverlässige Regeln

Auf der 1. Stufe wird fsgmatch mit zuverlässigen Transduktionsregeln angewandt. Das bedeutet, dass nur ganz eindeutige Eigennamen getaggt werden. Etwa 40 Prozent der vorkommenden Eigennamen können so gefunden werden. Zu beachten ist, dass die von den bisherigen Tools zugefügten Informationen rege benutzt werden, beispielsweise das Wortarten-Tag `JJ` für Adjektive, das vom Wortarten-Tagger zugefügt wurde. Beispiele von zuverlässigen Transduktions-Regeln:

Kontext-Regel	**Zuweisung**	**Beispiel**
`Xxxx+` is a`? JJ* PROF`	`PERS`	Yuri Gromov is a former director
`Xxxx+`, `DD+`,	`PERS`	White, 33,

`Xxxx+` ist eine Folge von grossgeschriebenen Worten, denen ein Tag zugeordnet werden soll; `DD+` ist eine Zahl; `PROF` ist eine Berufsbezeichnung wie etwa das beim Wortarten-Tagger erwähnte *analyst*; Fragezeichen `?` am Schluss eines Wortes bedeutet, dass das Wort optional ist; Stern `*` am Schluss eines Wortes bedeutet, dass es kein, ein oder mehrere Male vorkommen kann. `PERS` ist das durch die Regel zugewiesene Tag für Personennamen.

Auch werden auf dieser Stufe Bezeichnungen wie *Ltd.*, *Inc.* ausgenutzt, um Organisations-Namen zu ermitteln, oder Bezeichner wie *Mr.*, *Dr.*, *Sen.* zur Personennamenerkennung. Gefundene Eigennamen werden in dynamisch erzeugte und sich ändernde Listen

eingetragen.[39] Informationen aus diesen Listen werden als *wahrscheinlich*, nicht als *sicher* eingesetzt, eine sehr vorsichtige Taktik. Nur eindeutige Fälle werden getaggt, unsichere werden zurückgestellt für Stufe 3. Als unsichere Fälle gelten insbesondere Firmennamen mit einem *and* (Beispiel: *This was good news for China International Trust and Investment Corp*), Organisationen, die mit einem Wort beginnen, das auch wegen des Satzanfangs grossgeschrieben werden muss und ein Wort des allgemeinsprachlichen Lexikons ist (Beispiel: *Suspended* in *Suspended Ceiling Contractors Ltd denied the charge*) und Ortsnamen, die zwar in der benutzten Liste stehen, aber auch etwas anderes sein können, beispielsweise ein Personenname (Beispiel: *Washington*). Ortsnamen werden auf dieser Stufe nur getaggt, wenn auch der restliche Kontext dafür spricht.

Stufe 2: Partieller Match 1

Nun folgt ein Verarbeitungsschritt, der auf statistischer Basis funktioniert. Das Ziel ist, bei den in Stufe 1 gefundenen mehrwortigen Eigennamen zu untersuchen, ob Teilstrings davon ebenfalls als Eigennamen auftauchen.

Zuerst werden die bereits gefundenen Eigennamen in alle möglichen Teilstrings aufgeteilt und diejenigen Teilstrings, die im Text vorkommen, als *potenzielle* Eigennamen markiert. Dann wird ein vortrainiertes *Maximum-Entropy-Modell* angewandt, das Kontextinformationen berücksichtigt wie zum Beispiel die Position im Satz oder ob das Wort auch kleingeschrieben vorhanden ist, etc. Wenn das Modell eine positive Antwort liefert, dann wird der partielle Ausdruck als der korrespondierende Eigenname getaggt.

Nach Stufe 2 sind bereits 75 Prozent der Organisationsnamen, 80 Prozent der Personen- und 69 Prozent der Ortsnamen erkannt. Die Präzision liegt durchschnittlich etwa bei 97,5 Prozent.

Stufe 3: Gelockerte Regeln

Stufe 3 funktioniert ähnlich wie Stufe 1, also auch mit Transduktionsregeln (*fsgmatch*). Aber diese sind nun weniger strikt. Zusätzlich werden die in Stufe 1 und 2 dynamisch erstellten Eigennamenlisten in die Regeln miteinbezogen.

Beispielsweise wird nun angenommen, dass Namen von Organisationen und Personen nicht mehr verwechselt werden können[40], sondern dass solche, die wie Personennamen aussehen - eine in den beiden vorangegangen Stufen erstellte Liste mit Vornamen deutet darauf hin - auch solche sind. Wären sie Firmennamen, so wären sie bereits in Stufe 1 als solche erkannt worden. Auch Konjunktionsprobleme (vergleiche Stufe 1: *and* bei Organisationsnamen) können nun gelöst werden. Wenn ein Teil eines mehrwortigen Ausdrucks mit Konjunktion woanders alleine vorkommt, dann handelt es sich höchstwahrscheinlich um

[39] Für jeden Text wird eine eigene Liste erstellt, die anfangs immer leer ist.

[40] Ein prominentes Beispiel: *Philip Morris* kann ein Firmen- oder ein Personenname sein.

zwei Eigennamen, ansonsten um einen.

Auch das Satzanfangsproblem (vergleiche in Stufe 1 das Beispiel mit dem Bestimmungswort *Suspended*) wird nun auf ähnliche Art gelöst: Wird der Organisationsname woanders im Text mit dem grossgeschriebenen Bestimmungswort gefunden, so wird es in den Namen miteinbezogen, ansonsten nicht.

Erst jetzt am Schluss werden Listen mit Organisations- und Ortsnamen konsultiert und diejenigen, die im Text vorkommen, ohne Rücksicht auf den Kontext entsprechend markiert.

Nach Stufe 3 sind 83 Prozent der Organisations-, 90 Prozent der Personen- und 86 Prozent der Ortsnamen erkannt. Die Präzision ist etwas gesunken, sie liegt durchschnittlich etwa bei 95,5 Prozent.

Stufe 4: Partieller Match 2

Stufe 4 funktioniert wie Stufe 2, mit der Erweiterung, dass die in Stufe 3 gefundenen mehrwortigen Eigennamen auch ausgenutzt werden.

Nach Stufe 4 sind 85 Prozent der Organisations-, 93 Prozent der Personen- und 88 Prozent der Ortsnamen erkannt. Wieder ist die Präzision gesunken, allerdings sehr wenig, auf etwa 95 Prozent.

Stufe 5: Titel-Zuweisung

Das Textkorpus, das an der MUC verwendet wurde, hat die Besonderheit, dass Überschriften ausschliesslich in Grossbuchstaben geschrieben sind. Der sehr gute Indikator des grossen Anfangsbuchstabens bei potenziellen Eigennamen entfällt somit. Das Problem wird gelöst, indem jedes Wort mit im Text gefundenen Eigennamen (respektive Teilen davon) verglichen wird. Sodann wird wiederum mit einem Maximum-Entropy-Modell, das über Titel trainiert wurde, entschieden, ob es sich beim Wort aus der Überschrift um einen Eigennamen handeln kann oder nicht.

Nach der letzten Stufe sind 91 Prozent der Organisations-, 95 Prozent der Personen- und 95 Prozent der Ortsnamen erkannt - eine grosse Steigerung nochmals gegenüber Stufe 4. Die Präzision liegt immer noch bei etwa 95 Prozent.

c) Bemerkungen

Das Text Tokenization Tool der LTG hat am MUC-7-NE-Task am besten abgeschlossen! Vor allem in der Kategorie Organisationsnamen schneidet LT TTT viel besser als alle anderen am NE-Task teilnehmenden Systeme ab.[41] Gründe dafür sind wahrscheinlich in

[41] Vgl. Anhang A.

den folgenden, von der Forschergruppe hervorgehobenen, Punkten zu suchen:

- Das LT TTT- System benutzt wenig vorgefertigte Listen. Statt dessen wird der Kontext stark berücksichtigt, wodurch dynamische Listen erzeugt werden können. Dank solch spezifischer Listen kann präziser gesucht werden.
- Als Hauptunterschied zu den Systemen der anderen Teilnehmer wird ein konsequentes Verwenden des SGML- respektive XML-Paradigmas beschrieben. Dies ermögliche, dass jedes Tool auf eine spezifische Textstelle angewandt und zusätzlich dieser spezifischen Anwendung eine spezifische Ressource zugewiesen werden kann.[42].
- Die Möglichkeit der spezifischen Anwendung der Ressourcen wird von der Forschergruppe öfters hervorgehoben. Wahrscheinlich liegt darin die Hauptstärke des LT TTT-Systems, weil so eine präzisere Suche möglich ist, Fehler vermieden werden können.

Nebst dem Hervorheben dieser Eigenschaften, die LT TTT vor den anderen Systemen auszeichnet, räumt die Forschergruppe ein, dass das System auch Schwierigkeiten habe, nämlich dann, wenn Eigennamen ohne Kontext und nur ein Mal auftauchen.

5.1.5 NetOwl Extractor System

Quellen	[Kru98]
Erkannte NEs/Eigennamen	Gemäss MUC-NE-Task Definition (vergleiche [Chi97a])
Sprachen	Englisch

Die Firma *IsoQuest Inc.* hat am MUC-7-NE-Task mit ihrem kommerziellen System *NetOwl Extractor* teilgenommen. Dieses ist eine Weiterentwicklung des Systems *NameTag*, mit dem die Forschergruppe schon an der MUC-6 teilgenommen hatte und das 1995, also drei Jahre vor der MUC-7, zum ersten Mal herausgegeben wurde. Die an bereits 30 Kunden gelieferte Applikation wurde zur Teilnahme an der MUC-7 entsprechend angepasst.

In der Rangliste des MUC-7-NE-Wettbewerbs erscheint NetOwl Extractor zwei Mal - einmal auf Rang 2 und einmal auf Rang 10[43]. Die zwei verschiedenen Ränge rühren von zwei unterschiedlichen Durchläufen her: einem offiziellen Durchlauf (*Official Run*) und einem optionalen Durchlauf (*Optional Run*). Mit dem offiziellen Durchlauf wurde ein höherer

[42]Vgl.[Mik98]: "This allows the same tool to apply different strategies to different parts of the texts using different resources. The tools do not convert from SGML into an internal format and back, but operate at the SGML level."

[43]Vgl. *Anhang B: Rangliste NE-Task Englisch (MUC-7)*. Sieht man von den menschlichen Annotatoren ab, so wurden 14 Ränge verteilt.

F-Wert[44] erreicht, der mit 91.60 so hoch war, dass in der MUC-Gesamtwertung der 2. Rang erreicht wurde. Eine gute Leistung also. Allerdings wurde bei dieser Rangliste nur der F-Wert, nicht aber die Geschwindigkeit des Durchlaufs berücksichtigt. Die Forschergruppe von IsoQuest legte darauf aber grossen Wert, weshalb sie zusätzlich den optionalen Durchlauf mit nur 20 Prozent der Regeln vom offiziellen Durchlauf, dafür mit höherer Geschwindigkeit[45], tätigten.

a) Überblick System-Architektur

Das Bestreben um hohe Geschwindigkeit zeigt sich auch darin, dass in der Gesamtsystem-Architektur ein Compiler eingesetzt wird, der alle Ressourcen-Dateien in eine einzige Konfigurations-Datei umwandelt. Abbildung 5.2 illustriert dies und macht dabei deutlich, dass die *Engine* beim Durchlauf nur eine Datei, die *Konfigurations-Datei* (engl. *Configuration File*), konsultieren muss, und somit Zeit gespart werden kann. Diese Konfigurations-

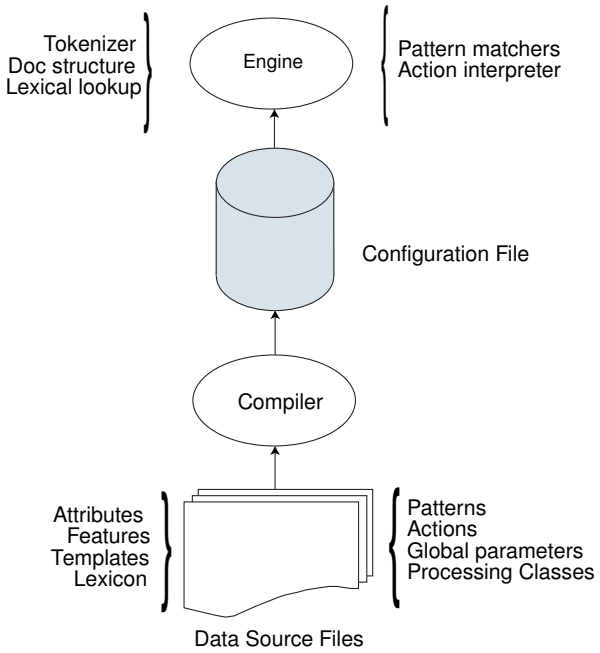

Abbildung 5.2: Architektur von NetOwl Extractor[46]

[44]Der F-Wert (engl. *F-Measure*) ist eine Masszahl, die erlaubt, die Leistung eines Systems zu messen. Er wird berechnet als Verhältnis von *Ausbeute* (engl. *Recall*) und *Präzision* (engl. *Precision*). Die Ausbeute gibt an, wieviele der im Antwortschlüssel getaggten Tokens durch das NER-System richtig getaggt wurden. Die Präzision gibt an, wieviele der durchs NER-System getaggten Tokens richtig getaggt wurden. Der (theoretisch) höchste F-Wert ist 100 (Prozent). Zur genauen Berechnung vgl. [Bor99], Seite 6.

[45]Die genauen Zahlen zur Geschwindigkeit sind [Kru98] zu entnehmen.

Datei steuert die Verarbeitung der Engine: Ein *Lexikon* und *Pattern-Regeln* legen fest, was die Engine erkennen soll, eine *Schablonen-Spezifikation* (engl. *Template*) und *Aktions-Definitionen* bestimmen, was die Engine extrahieren soll, und *Verarbeitungs-Klassen* (engl. *Processing classes*) definieren die verschiedenen Phasen der Verarbeitung durch die Engine.

b) Eigennamen-Erkennung

Die *Haupt-Verarbeitungsphase* (engl. *core processing*) der Konfigurations-Datei ist die Eigennamenerkennungsphase. Im Folgenden eine Wiedergabe der leider nur rudimentären Erläuterungen in [Kru98].

1. Linguistisches Wissen

Zur Erkennung von Eigennamen wird linguistisches Wissen über Struktur oder Komposition jedes Namen-Typs eingesetzt.[47] Beispielsweise wird das Wissen, dass Personennamen aus Vor- und Nachnamen mit optionalem Mittelnamen oder -initiale bestehen, verwendet, und zwar indem entsprechende Pattern-Regeln respektive *Schablonen* (*Templates*) geschrieben werden. Diese Regeln, abgelegt in *Daten-Quellen-Dateien* (engl. *Data Source Files*), werden dann wie erwähnt mittels Compiler in die Konfigurations-Datei eingefügt (vergleiche Abbildung 5.2).

2. Listen mit Eigennamen

Für Eigennamen, bei denen Pattern-Regeln oder Templates nicht oder nicht ausreichend greifen, werden *Listen von Eigennamen* eingesetzt. Dies ist zum Beispiel bei Ländernamen der Fall, in denen kein Kennzeichen wie *land* vorkommt, beispielsweise beim Ländernamen *France*. Auch sind in den umfangreichen Namenslisten *Akronyme*, die für Namen stehen (zum Beispiel *NYSE*), enthalten.

3. Identifikation von Aliassen

Mittels *Alias-Generierungs-Regeln* (engl. *Alias Generation Rules*) in der Konfigurations-Datei können Aliasse respektive verkürzte Variationen von Eigennamen erkannt werden. Dies ist zum Beispiel nützlich bei Akronymen für Organisationsnamen, Initialen bei Personennamen oder wenn bei Firmennamen Indikatoren wie *Corp.* weggelassen werden.

[46] Die Abbildung ist [Kru98] entnommen.

[47] Um die Terminologie von [McD93] zu benutzen: Interne und externe Evidenz wird ausgenutzt.

4. Mehrdeutigkeiten auflösen

Ein Name kann oft mehrere verschiedene Entitäten bezeichnen.[48] Um solche Mehrdeutigkeiten aufzulösen, wird jeder Regel[49] ein Gewicht zugewiesen.[50] In einer "*Regel-Wettkampf-Phase*" (engl. *rule competition phase*) werden die Gewichte verrechnet, so dass entschieden werden kann, welche Regel Anwendung findet.[51]

c) Anpassung von NetOwl an MUC-7-NE-Task

Da NetOwl ein kommerzielles Produkt ist und unabhängig von der MUC entwickelt wurde, entspricht es nicht in allen Punkten den von MUC aufgestellten Kriterien. Beispielsweise wählten die Entwickler von NetOwl eine andere Klassifizierung bei den Entitäten oder betrachteten Wortgruppen als zwei Entitäten, wohingegen diese für den MUC-NE-Task als eine Entität markiert werden mussten.

Um solche Unterschiede auszugleichen, musste die Konfiguration etwas angepasst werden. Dies wurde mit drei Mitteln bewerkstelligt: mittels *Tag-Abbildung*, durch zusätzliche *Patterns* und durch *Vergrösserung des Lexikons* um domänen-spezifische Eigennamen.

Tag-Abbildung: Das Tag-Abbildung (engl. *Tag Mapping*) ist eine mehr oder weniger direkte Abbildung der NetOwl-Tags in die von MUC vorgeschriebenen Tags. Die Forschergruppe bemängelt hierzu, dass die Abbildung deshalb nicht unmittelbar gemacht werden kann, weil die MUC-Spezifikation inkonsistent, ambig und unterspezifiziert sei. Ein Vorwurf, auf den hier nicht weiter eingegangen wird, da er nicht Thema dieser Arbeit ist.

Zusätzliche Patterns: Zusätzliche Patterns werden meist nach den herkömmlichen Patterns ausgeführt und bewirken unterschiedliche Dinge: Beispielsweise können mit einigen Patterns falsche Markierungen berichtigt werden. Andere Patterns sorgen für das Zusammenziehen von mehreren Tags zu einem.

d) Bemerkung

Neu an NetOwl gegenüber LTG, BSEE von FACILE / CONCERTO und LaSIE ist die Methode der Gewichtung (Gewichtung wurde beim BSEE zwar auch schon probiert, aber

[48] Wieder sei das Beispiel *Philip Morris* angeführt, vgl. 5.1.4, Seite 43.

[49] In [KRU98] wird nicht deutlich, wieviele und welche Arten von Regeln es insgesamt gibt. Wahrscheinlich handelt es sich nur um Pattern-Regeln und Schablonen, wobei aber wieder nicht klar ist, was diese beiden voneinander unterscheidet.

[50] Nach welchen Kriterien die Gewichtung vorgenommen wird, ist aus [KRU98] nicht zu erfahren.

[51] Auch wie die Verrechnung in der *Regel-Wettkampf-Phase* funktioniert, ist in [KRU98] unzureichend erklärt.

wieder fallengelassen). Leider wird nichts darüber ausgesagt, wie gut das Gewichten gelingt. Da NetOwl aber den 2. Rang erreichte, könnte es ein probates Mittel sein.

5.1.6 Oki Informations-Extraktions-System

Quellen	[Fuk98]
Erkannte NEs/Eigennamen	Gemäss MUC-NE-Task Definition (vergleiche [Chi97a])
Sprachen	Englisch

Mit ihrem *Informations-Extraktions-System* hat die japanische Firma *Oki* erstmals an der MUC-7 teilgenommen. Das Oki-System erfüllt mehrere MUC-Tasks: NE, CO, TE und TR.

Bisher baute Oki Übersetzungssysteme für Englisch-Japanisch und vice versa. Mit der Teilnahme an MUC-7 wollte Oki testen, wie gut sich Module, die für Übersetzungssysteme gebaut wurden, zur Informations-Extraktion einsetzen liessen.

a) Systemarchitektur und Funktion der Module

Abbildung 5.3 zeigt die Architektur des gesamten Oki-Systems, wie es für die MUC-7 gebaut wurde. Für die Eigennamen-Erkennung interessiert nur der Ablauf bis und mit *NE-Results*.

1. Oberflächen-Pattern-Erkennungs-Module

In den *Oberflächen-Pattern-Erkennungs-Modulen* (engl. *Surface Pattern Recognition Modules*) geschieht bezüglich der Eigennamen-Erkennung Folgendes: Zuerst werden alle (Sequenzen von) grossgeschriebenen Wörter(n) als NE-Kandidaten ermittelt. Sogleich wird anhand einer Wortliste mit Einträgen von "Nicht-Elementen"[53] geprüft, welche der NE-Kandidaten wieder zu streichen sind. Nun kommen Regeln zur Anwendung, mit denen erste Eigennamen gefunden werden können. Dafür werden *funktionale*[54] *Wörter* wie zum Beispiel *Bank* und Präpositionen ausgenutzt. Ein Beispiel: Findet sich die Verbindung des funktionalen Wortes *Bank* mit der Präposition *of*, so bilden diese zusammen mit den folgenden grossgeschriebenen Wörtern einen Eigennamen, zum Beispiel *Bank of Tokio*. Auch der Eigenname *University of Tokio* kann damit gefunden werden, da *University* als funk-

[52]Die Abbildung ist [Fuk98] entnommen.

[53]In [Fuk98] wird der Ausdruck "Non-element" verwendet. Es ist anzunehmen, dass damit Ausdrücke gemeint sind, die keine Eigennamen darstellen.

[54]Die Forschergruppe benutzt den Ausdruck "functional".

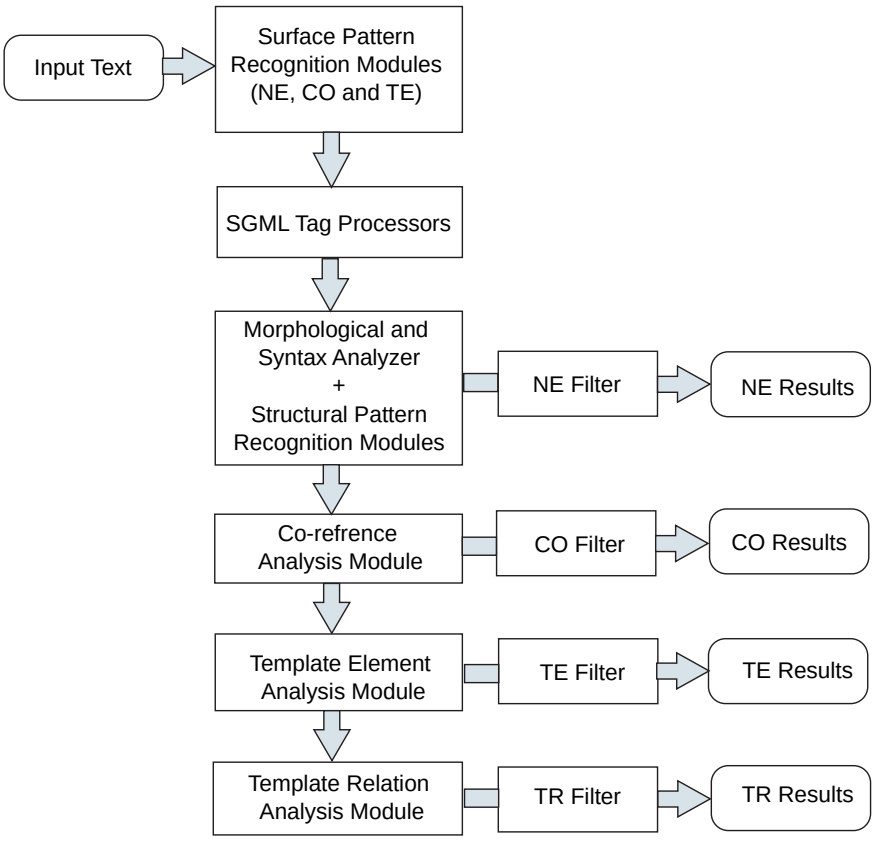

Abbildung 5.3: Architektur vom Oki Informations-Extraktions-System[52]

tionales Wort gilt. In einem nächsten Schritt werden die funktionalen Wörter bei NEs dazu ausgenutzt, sie zu kategorisieren. Beispielsweise deutet das funktionale Wort *Mr.* darauf hin, dass es sich bei dieser NE um einen Personennamen handelt, bei *Bank* um einen Organisationsnamen, bei *City* um einen Ortsnamen. Die restlichen NE-Kandidaten werden mit (von Hand erstellten) Namenslisten verglichen (pro Eigennamen-Klasse gibt es eine Liste). NE-Kandidaten, die in einer Namensliste vorhanden sind, werden entsprechend markiert.

Bevor das nächste Modul eingesetzt wird, werden die identifizierten Eigennamen dazu benutzt, koreferierende Eigennamen zu finden: Wurde beispielsweise der Personenname *Mr. John Doe*[55] gefunden, so werden Vorkommen von *John* respektive *Doe* ebenfalls als Personennamen markiert. Dasselbe gilt für Abkürzungen bereits gefundener Namen.

[55]Gemäss [Fuk98] wird *Mr.* seltsamerweise als Bestandteil des Personennamens betrachtet. Es könnte sich aber auch um einen Druckfehler handeln.

2. SGML-Tag-Prozessoren

Die *SGML-Tag-Prozessoren* (engl. *SGML Tag Processors*) dienen lediglich dazu, die Tags umzuformen, und zwar in Attribute der Wörter, denen sie zugehören. Dies ist nötig, weil das folgende Parsen ein solches *Attributformat* benötigt.

3. Morphologischer und syntaktischer Analysator

Beim folgenden Parsen wird eine morphologische und syntaktische Analyse durchgeführt. Als Resultat der syntaktischen Analyse ergeben sich Syntax-Bäume, wie die Resultate der morphologischen Analyse aussehen, wird aus [Fuk98] nicht klar.[56]

4. Struktur-Pattern-Erkennungs-Module

Die letzten zur Eigennamen-Erkennung nötigen Module sind die *Struktur-Pattern-Erkennungs-Module* (engl. *Structural Pattern Recognition Modules*). Sie werden auf die Syntax-Bäume angewandt, was im Vergleich zu allen anderen bisher vorgestellten regelbasierten Systemen ein Novum darstellt.

[Fuk98] gibt eine Auswahl von fünf strukturellen Patterns:

(i) Ist ein NE-Kandidat Subjekt gewisser Verben wie *say*, *die*, *play*, wird er als Personenname markiert.

(ii) Ist ein NE-Kandidat Nominalphrase vor dem Relativpronomen *who*, wird er als Personenname markiert.

(iii) Ist ein NE-Kandidat Nominalphrase gefolgt von *employee*, *spokesman* und ähnlichem, wird er als Organisationsname markiert.

(iv) Ist ein NE-Kandidat Nominalphrase, deren Apposition ein Personenname ist, wird er als Personenname markiert. Umgekehrt gilt das gleiche.[57]

(v) Ist ein NE-Kandidat Nominalphrase mit einer Präposition *in*, *at*, *near* oder *over*, deren Apposition ein Organisationsname ist, wird er als Organisationsname markiert.[58]

[56] Vgl. dazu auch den folgenden Abschnitt *b) Novum Parsen: Eine genauere Betrachtung.*

[57] Leider ist dieser Beschreibung kein Beispiel beigefügt. Wahrscheinlich handelt es sich um eine fehlerhafte Beschreibung. Es ist zu vermuten, dass es heissen sollte: "Ist ein NE-Kandidat Nominalphrase, deren Apposition ein *Titel* ist, wird er als Personenname markiert. Umgekehrt gilt das gleiche." Ein Beispiel dafür wäre dann der Fall *president of the USA, Bill Clinton*, wobei *president of the USA* der Titel ist.

[58] Wieder ist ein Anwendungsfall für dieses Pattern schwer vorzustellen. Vermutlich derselbe Fehler wie in (iv).

5. NE-Filter

Zum Schluss wird ein *NE-Filter* eingesetzt, der die immer noch im Attributformat vorliegenden NE-Informationen extrahiert und darauf den Eigennamen im Text die entsprechenden SGML-Tags beifügt.

b) Novum Parsen: Eine genauere Betrachtung

In [Fuk98] gibt es leider kein Beispiel, wie die Syntax-Bäume aussehen, die beim Parsen erstellt werden. Hier würde insbesondere interessieren, was die Resultate der morphologischen Analyse mit den Syntax-Bäumen zu tun haben. Braucht es die morphologische Analyse zur Erstellung eines Syntax-Baumes? Werden die Resultate der morphologischen Analyse in den Syntax-Baum eingebaut oder separat ausgewiesen?

Die Fragen sind wichtig, wenn man beurteilen möchte, wie nützlich morphologische und syntaktische Analyse sind. Es gibt Indizien, die darauf hindeuten, dass morphologische und syntaktische Analyse separate Resultate ergeben. Beispielsweise lautet der Kommentar in einem E-Mail eines Mitglieds der Forschungsgruppe: "But syntax analysis is not need for NE strongly, because this process is more useful rathaer [sic] CO, TE and TR than NE."[59] Etwas später dann: "To get higher precision, it is better to use morphorogical [sic] process to get part of speech and some information." Neben dem Schluss, dass die Ergebnisse morphologischer Analyse *nicht* in die Syntax-Bäume einfliessen, lassen die zitierten Worte weitere Schlüsse zu: Syntaktische Analyse scheint für die Eigennamen-Erkennung eher ungeeignet, morphologische Analyse hingegen gewinnbringend zu sein. Wie jedoch die Resultate der morphologischen Analyse aussehen und in welcher Art man sie verwenden kann, bleibt unklar.

Zur Vermutung über die geringe Nützlichkeit syntaktischer Analyse gelangt man auch beim Betrachten der strukturellen Patterns, die in [Fuk98] angeführt sind. Nur in einem der angegebenen Patterns wird Information verwendet, die mittels syntaktischer Analyse gefunden werden muss. Um für Pattern (i) das Subjekt eines Satzes zu ermitteln, ist syntaktische Analyse nötig. Allerdings könnte man statt dieser Regel auch eine etwas ungenauere Heuristik verwenden, etwa: *Folgt auf einen NE-Kandidaten eines der Verben 'say', 'die', 'play', dann ist er ein Personenname.* Für die Regeln (ii) und (iii) sieht man auf den ersten Blick, dass syntaktische Analyse nicht nötig ist. In beiden Regeln könnte man *Nominalphrase* einfach durch *ein grossgeschriebenes Wort respektive eine Folge grossgeschriebener Wörter* ersetzen, und diese Regeln wären auch ohne syntaktische Analyse anwendbar. Zu den Regeln (iv) und (v) kann leider kein Urteil abgegeben werden, da diese vermutlich falsch sind (vergleiche die Bemerkungen in den entsprechenden Fussnoten).

Zusammenfassend kann also festgehalten werden, dass syntaktisches Parsen zur Eigennamen-Erkennung eher ungeeignet ist. Wie nützlich morphologische Analyse ist, kann auf Grund der in [Fuk98] vorliegenden Informationen nicht festgestellt werden.

[59] E-Mail von Fumito Masui vom 29. Nov. 2001.

c) Bemerkungen

- Leider wird nur sehr rudimentär erklärt, wie in Oki die Eigennamen-Erkennung geschieht (wenig Beispiele, keine technischen Details). Es wird immer nur gesagt, *was* gemacht wird, aber kaum, *wie* es bewerkstelligt wird.
- Das Oki-System unterscheidet sich von den anderen bisher vorgestellten Systemen durch den Parser, der eine morphologische und syntaktische Analyse durchführt. Allerdings ist die syntaktische Analyse zur Eigennamen-Erkennung nicht erforderlich, sie wird erst für die weiteren Tasks benötigt.

5.1.7 LOLITA

Quellen	[Gar98]; [Mor95]
Erkannte NEs/Eigennamen	Gemäss MUC-NE-Task Definition (vergleiche [Chi97a])
Sprachen	Englisch

1986 startete die *Universität Durham* mit der Entwicklung von *LOLITA*, einem universellen NLP-System. Dies bedeutet, dass es sich um ein sehr umfangreiches *Kern-System* (engl. *core platform*) handelt, auf das ganz unterschiedliche Applikationen aufgesetzt werden können, wie zum Beispiel natürlichsprachliche Abfrage, Maschinelle Übersetzung, Informations-Extraktion, Tutoring beim Fremdsprachenlernen. Anders als bei ähnlichen NLP-Systemen handelt es sich nicht um ein Framework, das jeweils auf die spezifische Applikation zugeschnitten wird, sondern um ein System, bei dem jeweils nur die Wissensbasis um die spezifische Domäne einer Applikation erweitert wird.

An der MUC nahmen die Entwickler teil, um LOLITA zu evaluieren, indem die MUC-Tasks (NE, CO und TE) als Applikationen aufs Kern-System aufgesetzt wurden. Weder an der MUC-6 noch MUC-7 erzielte LOLITA im NE-Task gute Resultate: An der MUC-6 bewegt sich der F-Wert im Bereich von 60, an der MUC-7 im Bereich von 76 Prozent. Damit erreichte LOLITA im NE-Task den 13. und somit zweitletzten Rang.[60]

a) Systemarchitektur

LOLITAs *Kern-System* (in Abbildung 5.4 als *Lolita Core* bezeichnet) besteht aus drei Haupt-Komponenten: Wissensbasis, Analyse-Module, NL-Generator. Das Herz bildet die *Wissensbasis*, die als semantisches Netzwerk (engl. *Semantic Network*) organisiert ist. Erstellt wurde dieses zu 70 Prozent mit Hilfe des *WordNet*[61], der Rest ergibt sich unter

[60] Vgl. *Anhang B: Rangliste NE-Task Englisch (MUC-7).*

[61] WordNet ist eine Datenbasis, die lexikalische und semantische Informationen über englische Wortformen enthält. Weitere Informationen zu WordNet in [Mil90].

anderem aus der Analyse durch die dazu hintereinander geschalteten *Analyse-Module*. Die dritte wichtige Komponente ist der *NL-Generator*, der aus der Wissensbasis natürliche Sprache generieren kann. Da der NL-Generator für die Eigennamen-Erkennung keine Rolle spielt, werden im Folgenden nur die beiden anderen Komponenten kurz erläutert werden.

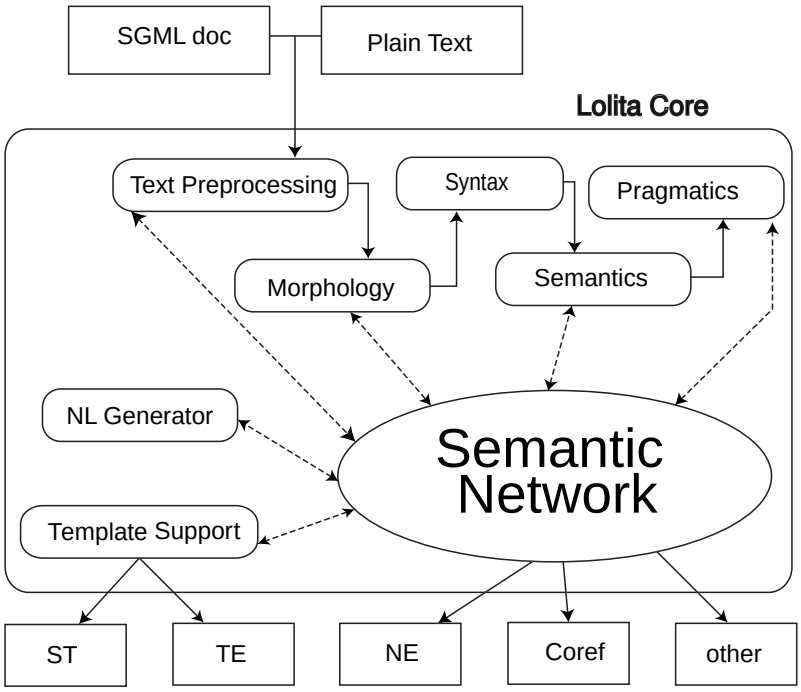

Abbildung 5.4: Architektur von LOLITA mit einigen Applikationen[62]

1. Wissensbasis als semantisches Netzwerk

LOLITA organisiert seine Wissensbasis als *semantisches Netzwerk* (engl. *Semantic Network*), das vereinfacht gesagt aus Knoten und Verbindungen derselben besteht. Knoten stehen für Entitäten oder Ereignisse. Den Knoten können *Attribute* (engl. *controls*) beigefügt werden, die unter anderem darüber Auskunft geben, wann es sich beim Knoten um eine NE handelt. Die Applikation der Eigennamen-Erkennung braucht nur noch auf diese Attribute in der Wissensbasis zu achten, um so die entsprechenden Tokens als Eigennamen markieren zu können.

[62]Die Abbildung ist [Gar98] entnommen.

2. Analyse-Module

Die Analyse-Module *Text-Vorverarbeitung, Morphologie, Syntax, Semantik* und *Pragmatik*[63] sind - in der genannten Reihenfolge - in einer Pipeline angeordnet. Dem ersten Modul dient der zu analysierende Text als Input. Wie in der Abbildung 5.4 dargestellt, ist jedes Modul mit dem semantischen Netz durch einen Doppelpfeil verbunden, was bedeutet, dass sowohl aus der Wissensbasis gelesen werden kann, als auch relevanter Output des jeweiligen Moduls darin eingetragen werden kann.

Durch die Analyse mit den Analyse-Modulen wird die Wissensbasis mit Daten aus dem Inputtext vergrössert: Neue Knoten mit Attributen und neue Verbindungen werden hinzugefügt, die dann als Basis für die NER-Applikation dienen.

Ohne die Analyse-Module im Detail zu erklären[64], soll dennoch das Syntax-Modul hervorgehoben werden, das ein Parsing durchführt. Erstaunlich ist dies deshalb, weil man erwarten würde, dass ein System erst *nach* der NER eine Syntax-Analyse durchführt, da die Identifikation der NEs das Parsen erleichtert. Dass die unübliche Reihenfolge der Abarbeitung Grund für die relativ schlechte NER-Leistung von LOLITA ist, ist gut möglich.[65]

b) NER-Komponenten

Seit MUC-6 ist LOLITA bezüglich NER weitgehend verändert worden. Im MUC-6-Beschrieb wird gesagt, dass der NER-Algorithmus nach der semantischen Analyse ansetze, indem er die ins semantische Netz eingetragenen Knoten (die für Ereignisse oder Entitäten stehen) untersuche. Knoten, denen bestimmte Attribute beigefügt wurden, könnten so als NEs identifiziert werden.

Zusätzlich wurde in MUC-6 versucht, Namenslisten zu integrieren. In [Mor95] schreiben die Autoren aber, dass sich bei Firmen- und Ortsnamen keine merkliche Verbesserung ergeben hätte und man wieder davon abgesehen hätte. Jedoch sei schon eine kleine Menge von Informationen über Personennamen im semantischen Netzwerk verfügbar gewesen - wahrscheinlich von WordNet herrührend.

Die schlechte Leistung bei der NER von MUC-6-LOLITA führen die Autoren unter anderem auf ungeschicktes Parsing zurück. Wenn dieses fehlschlage, dann könne das nachfolgende Semantik-Modul - das zur Hauptsache für die NER verantwortlich ist, weil es die dazu nötigen Informationen bereitstellt - nicht mehr arbeiten, da ihm der Input fehle. Dies wurde in MUC-7-LOLITA dann durch Vergrösserung und Verbesserung der Grammatik

[63] In Abbildung 5.4 entsprechend als Text *Preprocessing*, *Morphology*, *Syntax*, *Semantics* und *Pragmatics* bezeichnet.

[64] Die Analyse-Module im Einzelnen zu erläutern wird hier deshalb als nicht sinnvoll erachtet, weil es sehr aufwändig ist und der Zusammenhang mit der Eigennamen-Erkennung daraus kaum hervorgeht (vgl. dazu auch den folgenden Abschnitt *b) NER-Komponenten*). Wer dennoch genauere Informationen zu den Modulen wünscht, sei auf [Gar98] und [Mor95] verwiesen.

[65] Vgl. dazu die Bemerkungen im folgenden Abschnitt *b) NER-Komponenten*.

und durch verschiedene Back-up-Strategien beim Parsing verbessert.[66]

Eine weitere Schwachstelle von MUC-6-LOLITA sei auch, dass Information, die durch vorangegangene Module ermittelt wurde, verloren gehe, wenn das Semantik-Modul fehlschlage. So könne es beispielsweise vorkommen, dass der Parser (Syntax-Modul) einen Eigennamen als solchen erkennt, diese Information nicht in die NER einfliesse, wenn das Semantik-Modul aus irgendeinem Grund seinen Dienst versage.

Will man aus [Gar98] erfahren, wie die NER in MUC-7-LOLITA bewerkstelligt wurde, wird man Ungereimtheiten feststellen. Zwar ist einerseits beschrieben, dass die Parsing-Strategie zugunsten der NER erweitert wurde, wie in [Mor95] angekündigt war. Doch andererseits ist plötzlich von NER-Komponenten die Rede, was darauf hinweist, dass NEs in *mehreren* Modulen erkannt werden, obwohl in [Mor95] nur von einem NER-Algorithmus die Rede war, der nach dem semantischen Modul einsetze. In welchen Modulen und auf welche Art NER durchgeführt wird, geht aus [Gar98] aber nicht klar hervor. Anhand von Hinweisen in den Kapiteln "Ausgeführte Verbesserungen seit MUC-6" und "Resultate [für den NE-Task]"[67] wurde hier versucht, ein konsistentes Bild der NER in MUC-7-LOLITA zu erstellen, mangels genauem Beschrieb der NER müssen aber viele Fragen offen bleiben.[68]

1. Eigennamen-Erkennung im Parser

Da in der Textstelle "The components responsible for named entity recognition have been revised and many new rules have been added."[69] von Regeln die Rede ist, ist zu vermuten, dass damit die Grammatikregeln im Parser gemeint sind. Das würde auch Sinn machen, wurde doch schon in [Mor95] erwähnt, dass der Parser NEs erkenne. Interessieren würde hier natürlich, wie diese Regeln aussehen. Zwei Textstellen in [Gar98] enthalten Hinweise dazu:

- "Grammar rules have been added to deal with constructions common in the training texts, such as references to aircraft and flights (*Boeing 747*, or *Paris-bound Boeing 747*, *the TWA flight 800*, etc)." Mit solchen Grammatik-Regeln wird die innere Struktur (*internal evidence*, vergleiche [McD93]) von Eigennamen verwendet, um sie zu finden. Jedes der genannten Beispiele enthält als letzten Bestandteil eine Zahl - dieser Umstand wird wahrscheinlich mit Regeln in Form von regulären Ausdrücken ausgenützt werden.

[66] Folgende Back-up-Strategien, die in [Gar98] angeführt werden, seien hier kurz erwähnt: Zeitliche Limitierung des Parsing-Prozesses, Neuerungen beim Parsing von Überschriften (spezifische Grammatik, Analyse erst am Ende des Textes), Einführung von *Insel-Parsing* (engl. *Island Parsing*) und Einsatz von Brill-Tagger und reduzierter Grammatik für Sätze, deren Parsing fehlschlug.

[67] Vgl. [Gar98]: Original heissen diese Kapitel dort "Improvements carried out since MUC-6" und "Results: The Named Entity Task".

[68] Folgendes Zitat aus [Gar98] soll als kleine Kostprobe der Quellensituation dienen: "Finally, in cases where all three parsing passes fail, a way of recovering most named entities [...] has been devised." Doch wie dieser Weg aussieht, wird nicht erklärt!

[69] [Gar98].

- "A change in the treatment of unknown proper names that appear without clear designators (i.e., without *Corp, Ltd, Mrs., etc*) has been introduced. In the MUC-6 system a decision as to what type of entity an unknown name stood for was made early [...]" Hier lässt sich schliessen, dass wie schon in den meisten bisher vorgestellten Systemen Wörter als Indikatoren - hier "designators", also *Designatoren*, genannt - für Eigennamen-Erkennungs-Regeln verwendet werden. Offen bleibt allerdings, wie das System die zu klassifizierenden "unknown proper names" ermittelt hat. Es ist anzunehmen, dass damit einfach grossgeschriebene Wörter respektive Wortfolgen gemeint sind.

Die Neuheit, auf die im obigen Zitat angesprochen wird, ist eine *Verzögerung beim Kategorisieren* unbekannter Eigennamen. Erst wenn Desambiguierungs-Information vorliegt, wird entschieden, um welche Art von Eigenname es sich handelt. Ein Beispiel:

(9) *Shortly after Fossett's launching Monday his competitors sent him telegrams of congratulation.*

Das System kann beim Einlesen von *Fossett* noch nicht entscheiden, welcher Kategorie dieser Eigenname angehört, es könnte sowohl ein Personenname als auch ein Firmenname sein. Erst wenn das Pronomen *his* eingelesen wird, kann entschieden werden, dass *Fossett* ein Personenname sein muss.

2.Eigennamen-Erkennung dank Morphologie-Modul

Auch das *Morphologie-Modul* trägt seinen Teil zur Eigennamen-Erkennung bei. Neu hat es in MUC-7-LOLITA Zugriff auf vom Semantik- und Pragmatik-Modul ermittelte Eigennamen des vorangegangenen Textteils (der Text wird satzweise abgearbeitet). Was aber das Morphologie-Modul mit diesen Daten anfängt, ist nicht beschrieben.

3. Eigennamen-Erkennung mit Listen

Neben Listen mit Designatoren (vergleiche Abschnitt *1. Eigennamen-Erkennung im Parser*) werden für MUC-7-LOLITA auch Namenslisten verwendet, obwohl die Idee für MUC-6-LOLITA verworfen worden war.[70] Namenslisten von Personen (Vornamen), Orte, Organisationen und Namen aus dem Fluggewerbe (Fluggesellschaften, Flughäfen und sonstige) wurden ins semantische Netz eingefügt.[71]

[70] Vgl. Einleitung von Abschnitt *b) NER-Komponenten*.

[71] Auf Namen aus dem Fluggewerbe wurde deshalb Wert gelegt, weil die Artikel, die an der MUC-7 zum Testen ausgeteilt wurden, hauptsächlich vom Fluggewerbe handeln.

4. Eigennamen-Erkennung mit Akronym-Matching-Algorithmus

Der Klammersatz "([...] Unfortunately, the rules in the acronym matching algorithm didn't handle this case correctly.)"[72] weist auf einen Algorithmus zur Erkennung von Akronymen hin. Da wieder von Regeln die Rede ist, ist anzunehmen, dass dieser Algorithmus im Parser eingebaut ist. Genaueres ist aber nicht zu erfahren.

c) Bemerkungen

- Das Durchführen des Parsings *vor* der NER ist wahrscheinlich ungeschickt, denn NER soll ja das Parsen erleichtern.
- Die schlechte Leistung von LOLITA ist wahrscheinlich auch auf die Anordnung der Module in einer Pipeline zurückzuführen, weil der Output des einen Moduls Input des nächsten ist. Da das Parsing gerne fehlschlägt, bricht die Analyse ab. Mit einigen Parsing-Back-up-Strategien wurde bei MUC-7-LOLITA versucht, dieses Abbrechen zu vermeiden. Inwieweit das gelungen ist, lässt sich schlecht beurteilen. Die Gesamtleistung von MUC-7-LOLITA bezüglich NER ist gestiegen, worauf sie zurückzuführen ist, ist nicht festzustellen.
- Das semantische Netz ist sehr kompliziert, LOLITAs Core ist ein riesiges Unterfangen. Es ist für zu viele Bereiche verwendbar, als dass es jeden Task mit guter bis sehr guter Leistung erfüllen könnte.

5.2 Statistisch basierte Systeme

5.2.1 PIE-System mit Erweiterung durch Kollokations-Statistik

Quellen	[Lin98]; [Lin95]
Erkannte NEs/Eigennamen	Gemäss MUC-NE-Task Definition (vergleiche [Chi97a])
Sprachen	Englisch

Die *Universität Manitoba* ist im Gebiet der IE tätig und hat an der MUC mehrmals teilgenommen. Mit dem *PIE*-System[73] nahm sie an der MUC-6 teil, und zwar an den folgenden Tasks: NE, CO, TE und ST. PIE arbeitet *regelbasiert*. An der MUC-7 nahm die Universität

[72] [Gar98].

[73] PIE steht für *Principar-driven Information Extraction*, wobei *Principar* wiederum der Name des eingesetzten Parsers ist.

Manitoba mit einem System teil, das als Erweiterung der Eigennamen-Erkennung von PIE fungierte. Dabei wurden *statistische* Methoden verwendet: Mit Hilfe von Kollokationen wurde versucht, einerseits weitere Eigennamen-Erkennungs-Regeln automatisch zu extrahieren, andererseits unbekannte Eigennamen zu klassifizieren.

a) MUC-6: PIE-System

Bezüglich Eigennamen-Erkennung ist beim PIE-System nichts Neues zu erfahren. NEs werden wie bei den in Kapitel 5.1 beschriebenen Systemen mit Hilfe von NE-Regeln, die auf dem Prinzip des Mustervergleichs (engl. *Pattern Matching*) basieren, erkannt. Zwar wird in PIE ein Parser eingesetzt. Dieser dient jedoch anderen Zwecken[74], die NEs werden vor der Parser-Analyse erkannt.

Auch Listen mit Eigennamen und Firmennamen-Bezeichnern (*Co.*, *Ltd.* etc.) werden insofern verwendet, als sie direkt im Lexikon, das beim Mustervergleich benutzt wird, integriert sind. Neben den üblichen syntaktischen Informationen, die jeder im Lexikon eingetragene Begriff enthält, enthalten Eigennamen auch semantische Informationen, das heisst, welcher Art der Eigenname ist (Personen-, Organisations- oder Ortsname).

Regeln zur Erkennung von Eigennamen bestehen immer aus den beiden Teilen *Muster* und *Aktion*. Matcht eine Sequenz grossgeschriebener Wörter[75] des Input-Textes mit dem Muster einer Eigennamen-Regel, wird - als Aktion - diese Wortfolge ins Lexikon aufgenommen, und zwar inklusive der durch die Regel gewonnenen semantischen Information, um welche Art von Eigennamen es sich handelt. Passt keine der Eigennamen-Regeln auf eine zu bestimmende Sequenz grossgeschriebener Wörter des Input-Textes, so wird geprüft, ob diese Sequenz ein *Substring* einer bereits als Eigenname erkannten Wortfolge ist. Ist dies der Fall, erhält die Sequenz dieselben semantischen Informationen wie der bereits erkannte Eigenname und wird ebenfalls ins Lexikon aufgenommen.

Diese neuen Lexikon-Einträge dienen später als Grundlage für die entsprechende Eigennamen-Markierung im Input-Text.

b) MUC-7: Erweiterung der Eigennamen-Erkennung durch Kollokations-Statistik

Um die Leistung der Eigennamen-Erkennung zu steigern, entwickelte die Universität Manitoba für die MUC-7 ein System, das die Eigennamen-Erkennung von PIE *erweiterte*. Diese Erweiterung kommt mittels zwei Techniken zu Stande:

- Automatisches Extrahieren weiterer Eigennamen-Erkennungs-Regeln
- Klassifikation unbekannter Eigennamen mittels eines Naiven Bayes-Klassifikators

[74] Vom Parser profitieren die CO-, TE- und ST-Tasks.

[75] Wenn jetzt und im Folgenden von *Sequenz* respektive *Folge (grossgeschriebener Wörter)* die Rede ist, so ist damit auch immer der Fall eines einzelnen grossgeschriebenen Wortes mitgemeint.

Nr.	Hauptwort	Relation	Relationswort	Relation zwischen
1	`have`	`V:subj:N`	`I`	einem Verb und seinem Subjekt
2	`have`	`V:comp1:N`	`dog`	einem Verb und seinem nominalen Objekt
3	`dog`	`N:jnab:A`	`brown`	einem Nomen und seinem adjektivischen Modifizierer
4	`dog`	`N:det:D`	`a`	einem Nomen und seinem Determinator
5	`I`	`N:r-subj:V`	`have`	[Umkehrung von 1]
6	`dog`	`N:r-comp1:V`	`have`	[Umkehrung von 2]
7	`brown`	`A:r-jnab:N`	`dog`	[Umkehrung von 3]
8	`a`	`D:r-det:N`	`dog`	[Umkehrung von 4]

Tabelle 5.3: Kollokationen des Satzes *I have a brown dog*. `V` steht für Verb, `N` für Nomen, `D` für Determinator, `A` für Adjektiv. Die Bezeichnungen `subj, comp1, jnab, det` etc. sind Abkürzungen für die Relationsarten.

Beide Techniken stützen sich auf das Ausnutzen von Kollokationen. Solche werden aus umfangreichen Textkorpora gewonnen und in einer Kollokations-Datenbasis abgelegt.

1. Erstellen der Kollokations-Datenbasis

Zur Erstellung der Datenbasis wurden Sätze aus Textkorpora mit insgesamt 100 Millionen Wörtern geparst. Um Fehlern vorzubeugen, wurden nur solche Sätze geparst, die weniger als 26 Wörter enthielten, und nur vollständig geparste Sätze wurden weiterverwendet. So verringerte sich die Anzahl der Wörter in den Parse-Bäumen auf 22 Millionen.

Aus den Parse-Bäumen werden nun Kollokationen ermittelt. Eine solche besteht aus drei Teilen: *Hauptwort, Relation, Relationswort*. Nicht jedes Wort steht mit jedem in Beziehung, nur bestimmte Relationen interessieren. Informationen, welche Wörter wie miteinander in Beziehung stehen, werden den Parse-Bäumen entnommen. Welche Bedingungen jedoch bei der Auswahl der Beziehungen gelten, geht aus [Lin98] nicht deutlich hervor. Welche Kollokationen von Interesse sind, soll Tabelle 5.3 anhand des Beispielsatzes *I have a brown dog* verdeutlichen.

Für jedes Hauptwort wird in der Kollokations-Datenbasis ein Eintrag gemacht, wobei jede Relation mit Relationswort und der Häufigkeit des Vorkommens aufgeführt wird.

2. Automatisches Extrahieren weiterer Eigennamen-Erkennungs-Regeln

Aus der Kollokations-Datenbasis können nun weitere Eigennamen-Erkennungs-Regeln gewonnen werden. Eigentlich sollte von *Klassifikations-Regeln* gesprochen werden, denn immer geht es darum, dass eine Sequenz grossgeschriebener Wörter bereits als unbekannter Eigenname erkannt wurde und dieser folglich nur noch klassifiziert werden soll.

Folgendes Beispiel soll der Veranschaulichung dienen, wie eine Eigennamen-Erkennungs-Regel respektive eine Klassifikations-Regel aus der Kollokations-Datenbasis automatisch extrahiert werden kann: In dem Korpus mit 22 Millionen Wörtern taucht 33 Mal[76] ein Eigenname als pränominaler Modifizierer von *managing director*[77] auf (das heisst in der Form von *Microsoft managing director*). In 26 der 33 Fälle konnte der Eigenname bereits als Organisationsname erkannt werden (beispielsweise mit Hilfe des Lexikons), in 7 Fällen jedoch kann der Eigenname nicht klassifiziert werden. Da die Wahrscheinlichkeit gross[78] ist, dass es sich auch bei den 7 unbekannten Eigennamen um Organisationsnamen handelt, kann deshalb eine Eigennamen-Erkennungs-Regel folgender Form erstellt werden: "Ist eine Sequenz grossgeschriebener Wörter pränominaler Modifizierer von *managing director*, so handelt es sich bei der Sequenz um einen Organisationsnamen." Mittels dieser Methode wurden 3623 Kollokationen gefunden, wovon viele zuvor bereits manuell ermittelt wurden.

3. Klassifikation unbekannter Eigennamen mittels eines Naiven Bayes-Klassifikators

Meist sind die Zahlen aber nicht so eindeutig wie im obigen *managing director*-Beispiel. Es könnte beispielsweise vorkommen, dass ein Eigenname[79] in einer bestimmten Relation mit einem bestimmten Hauptwort in 75 Prozent der Fälle ein Organisationsname ist, in 25 Prozent ein Personenname. Zwar ist die Wahrscheinlichkeit, dass der Eigenname dann ein Organisationsname ist, drei Mal höher als ein Personenname, jedoch ist die Wahrscheinlichkeit, dass es sich dennoch um einen Personennamen handelt, nicht so klein, dass man sie einfach vernachlässigen dürfte. In solchen Fällen bietet sich ein anderes Verfahren an: Mittels *Naivem Bayes-Klassifikator*, der alle Kollokationen untersucht, in denen ein bestimmter Eigenname vorkommt, und daraus dessen wahrscheinlichste Klasse errechnet, kann der Eigenname klassifiziert werden.

Wieder sei die Funktionsweise anhand eines Beispiels erklärt: In einem zu markierenden Text kommt das Wort *Xichang* vor. Im Lexikon kann es nicht gefunden werden. Da es grossgeschrieben ist, muss es sich somit um einen unbekannten Eigennamen handeln. In Tabelle 5.4 sind diejenigen Kollokations-Kontexte aufgeführt, die der Parser im Text findet.

Sind die Kollokations-Kontexte ermittelt, wird für jedes der Hauptwörter in der Kollokations-Datenbasis geprüft, wie oft es in der gefundenen Relation mit einem Eigennamen als Partnerwort vorkommt. Wichtig ist dabei, dass unterschieden wird, um welche Klasse von

[76] Es kann sich dabei um 33 verschiedene Eigennamen handeln.

[77] *managing director* wird als Einheit, also wie ein einzelnes Wort, betrachtet.

[78] Was "grosse Wahrscheinlichkeit" heisst, bestimmt ein willkürlich festgelegter Schwellwert.

[79] Wieder muss es nicht ein bestimmter Eigenname sein, sondern irgendeiner.

i	Kollokations-Kontext	Hauptwort	Relation	Beschreibung
1	*the Xichang base*	`base`	`N:nn:N`	Xichang ist pränominaler Modifizierer von *base*
2	*the Xichang site*	`site`	`N:nn:N`	Xichang ist pränominaler Modifizierer von *site*
3	*the site in Xichang*	`in`	`P:pcomp:N`	Xichang ist Objekt der Präposition (P) *in*

Tabelle 5.4: Kollokations-Kontexte von *Xichang*

Eigennamen es sich jeweils handelt. Bei drei Klassen von Eigennamen - Ortsnamen (LOC), Organisationsnamen (ORG) und Personennamen (PER) - sind also pro Hauptwort und entsprechender Relation drei Werte zu ermitteln. Tabelle 5.5 zeigt, wie die Werte aussehen könnten.

i	Hauptwort	Relation	Häufigkeitszahlen		
			LOC	*ORG*	*PER*
1	`base`	`N:nn:N`	77	19	0
2	`site`	`N:nn:N`	26	16	34
3	`in`	`P:pcomp:N`	35641	15630	0

Tabelle 5.5: Häufigkeiten der Kollokations-Kontexte, in denen *Xichang* anzutreffen war. Die Werte stammen aus der mittels Trainingskorpus (22 Mio Wörter) erstellten Kollokations-Datenbasis.

77 Mal kommt `base` mit einem Ortsnamen als pränominalem Modifizierer vor, 19 Mal mit einem Organisationsnamen als pränominalem Modifizierer und nie mit einem Personennamen. In der gleichen Art sind die Zeilen 2 und 3 zu lesen. Anhand dieser Zahlen lässt sich nun berechnen, wie wahrscheinlich es ist, dass ein Eigenname, der in diesen Relationen vorkommt, der Klasse LOC, ORG oder PER angehört.

C soll für die Eigennamen-Klasse stehen (also für LOC, ORG oder PER). F_i ist ein so genanntes *Kollokations-Feature*: ein Hauptwort in Verbindung mit seiner Relation, also zum Beispiel F_1 = `(base ; N:nn:N)`[80]. Die F_i werden von F_1 bis F_k indexiert.[81] Es muss

[80] Vgl. Tabellen 5.4 und 5.5.

[81] In Tabellen 5.4 und 5.5 ist $k = 3$.

nun dasjenige C bestimmt werden, das die Formel

$$\prod_{i=1}^{k} P(C \mid F_i) \tag{5.1}$$

maximiert. Mit Hilfe des Theorems von Bayes

$$P(C \mid F_i) = \frac{P(F_i \mid C)P(C)}{P(F_i)} \tag{5.2}$$

kann Formel (5.1) umgewandelt werden in

$$\prod_{i=1}^{k} P(F_i \mid C)P(C) \tag{5.3}$$

Da es sich bei $P(F_i)$ um eine Konstante handelt und dasjenige C gesucht wird, durch das die Formel (5.3) maximiert wird, darf die Konstante im Nenner weggelassen werden.

$P(C)$ ist eine geschätzte Wahrscheinlichkeit, die dadurch entsteht, dass die Kardinalität aller im 22 Millionen-Wörter-Korpus vorkommenden Eigennamen der jeweiligen Klasse C geteilt wird durch die 22 Millionen Wörter. $P(F_i \mid C)$ ist die Wahrscheinlichkeit eines Kollokations-Features F_i in Verbindung mit dem Eigennamen C, also zum Beispiel `(base ; N:nn:N ; LOC)`. Dieser Teil der Formel wird folgendermassen geschätzt:

$$P_m(F_i \mid C) = \frac{\parallel F_i, C \parallel + \frac{1}{|CF|}}{\sum_{f \in CF} \parallel f, C \parallel + 1} \tag{5.4}$$

Der Index m weist auf eine so genannte *m-Schätzung* hin, die dazu dient, im Trainingskorpus nicht angetroffenen Kombinationen von Kollokations-Features und Klassen eine geringe Wahrscheinlichkeit zuzuordnen. Mathematisch gesehen wird dies dadurch erreicht, dass im Zähler $\frac{1}{|CF|}$ und im Nenner 1 hinzuaddiert wird. CF ist die Menge aller Kollokations-Features. $\parallel f, C \parallel$ bezeichnet die Anzahl der Eigennamen, die C angehören im Kollokations-Feature, das durch die Variable f repräsentiert wird.

Beispiel: In Tabelle 5.5 ist F_1 `(base ; N:nn:N)`. C sei LOC. Die Werte der Tabelle 5.5 in die Formel (5.4) eingesetzt - die zweiten Summanden in Zähler und Nenner dabei unberücksichtigt - ergibt:

$$P(F_1 \mid LOC) = \frac{\parallel F_1 \mid C \parallel}{\parallel F_1 \mid C \parallel + \parallel F_2 \mid C \parallel + \parallel F_3 \mid C \parallel} = \frac{77}{77 + 26 + 35641} \tag{5.5}$$

Auf die selbe Art werden noch $P(F_2 \mid LOC)$ und $P(F_3 \mid LOC)$ berechnet, dann kann in Formel (5.3) eingesetzt werden. Der gewonnene Wert wird gespeichert, die Berechnungen werden in gleicher Weise für $C = ORG$ und $C = PER$ durchgeführt. Dasjenige C, bei

dem die Wahrscheinlichkeit P am grössten wird, ist die Eigennamenklasse, der *Xichang* zugeordnet wird. In vorliegenden Beispiel wird es *LOC* sein.

c) Bemerkungen

- Von den bisher beschriebenen Systemen ist PIE das erste, das versucht, Eigennamen-Erkennungs-Regeln *automatisch* zu ermitteln. Und das gelingt nicht mal schlecht.
- Die Kombination aus regelbasierten und statistisch arbeitenden Systemen scheint vielversprechend.

5.2.2 IdentiFinder

Quellen	[Mil98]; [Bik98]; [Bik99]
Erkannte NEs/Eigennamen	Gemäss MUC-NE-Task Definition (vergleiche [Chi97a])
Sprachen	Englisch und Spanisch

An der MUC-7 hat die US-amerikanische Firma *BBN* an den drei Tasks NE, TE und TR teilgenommen. Das für den NE-Task entwickelte NER-System *IdentiFinder* arbeitet mit einer Variante eines Hidden Markov Modells.

a) Basis Hidden Markov Modell

Ein *Hidden Markov Modell (HMM)* wird meist dazu verwendet, den Wörtern in einem Text die richtigen Wortarten respektive Tags zuzuweisen. Dabei werden die Wortarten als die verborgenen (engl. *hidden*) *Zustände* betrachtet, die Wörter als die sichtbaren *Symbole*, die von den Zuständen ausgegeben werden.

1. Trainings-Phase

Bevor ein HMM auf einen zu taggenden Text angewandt werden kann, muss es entsprechend trainiert werden. Das heisst, dass die *Übergangswahrscheinlichkeiten* zwischen jeweils zwei Zuständen und die *Ausgabewahrscheinlichkeiten* eines Zustandes für ein Symbol berechnet werden müssen.

Die Abbildung 5.5 zeigt ein Beispiel eines reduzierten HMM-Diagramms für die Nominalphrase *die auf der Bank sitzende Frau*. Die Zahlen beziffern die errechneten Übergangs-

und Ausgabewahrscheinlichkeiten (für genauere Angaben vergleiche Bildkommentar). Reduziert ist das Diagramm deshalb, weil viele Zustände nicht abbgebildet sind. Für ein vollständiges HMM müssten pro Wort der Nominalphrase alle möglichen Zustände, also alle vorkommenden Wortarten-Tags, aufgeführt werden. Weggelassen wurden diejenigen Knoten, deren Ausgabewahrscheinlichkeit für das entsprechende Wort null beträgt.[82] Da sich die *Gesamtwahrscheinlichkeit* eines Pfads durch das Diagramm - und diese ist es, die interessiert, wenn man wissen will, wie gross die Wahrscheinlichkeit einer Wortartenfolge für eine bestimmte Wortfolge ist - mittels Multiplikation der auf dem Pfad liegenden Werte (sowohl der Übergangs- als auch der Ausgabewahrscheinlichkeiten) berechnen lässt, wäre die Wahrscheinlichkeit für einen Pfad, der einen Knoten mit Ausgabewahrscheinlichkeit null passiert, ebenfalls null und somit irrelevant. Um der Übersichtlichkeit willen wurden solche Fälle in Abbildung 5.5 nicht dargestellt.

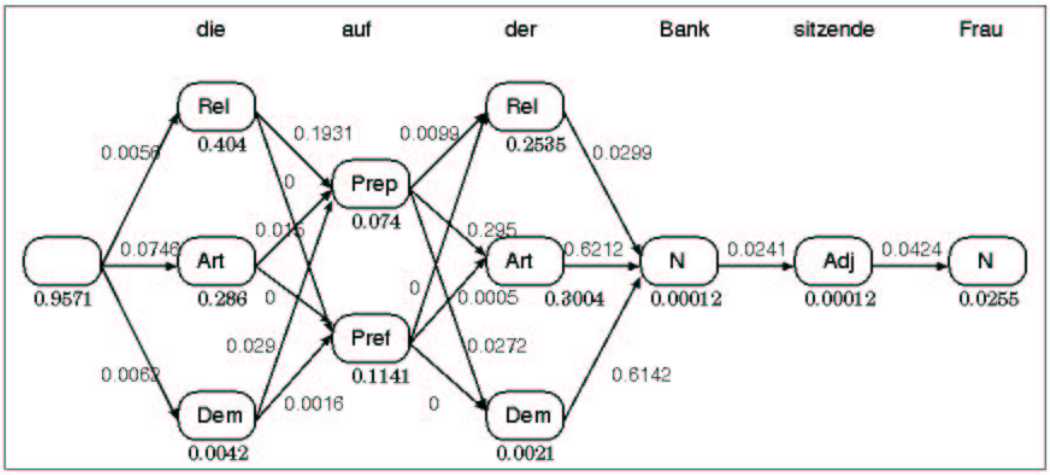

Abbildung 5.5: Reduziertes HMM für die Nominalphrase *die auf der Bank sitzende Frau*. Die Zustände sind als Knoten dargestellt und mit Kürzeln für Wortarten beschriftet (z. B. `N` für Nomen, `Adj` für Adjektiv). Die unter den Knoten stehenden Zahlen beziffern die Ausgabewahrscheinlichkeit des entsprechenden Zustandes für das jeweilige Wort. Beispielsweise beträgt die Ausgabewahrscheinlichkeit für das Wort *die* im Zustand `Rel`(Relativpronomen) 0.404. Die den Pfeilen zugeordneten Zahlen beziffern die Übergangswahrscheinlichkeit zwischen zwei Zuständen.[84]

Liegt ein genügend grosses, korrekt getaggtes Trainingskorpus vor, so werden die Wahrscheinlichkeiten als *Maximum-Likelihood-Schätzwert* ermittelt.

[82]Beispielsweise ist die Ausgabewahrscheinlichkeit des Zustandes `Rel` (Relativpronomen) für das Wort *Bank* null, da *Bank* nie ein Relativpronomen sein kann.

[84]Die Abbildung ist [Sch00] entnommen.

Zur Schätzung der Übergangswahrscheinlichkeit von einem Zustand t_{i-1} nach t_i wird der Maximum-Likelihood-Schätzwert ermittelt, indem die Anzahl der im Trainingskorpus vorkommenden Zustandsfolgen (t_{i-1}, t_i) durch die Anzahl Vorkommnisse von t_i geteilt werden. Formal:

$$P(t_i \mid t_{i-1}) = \frac{C(t_{i-1}, t_i)}{C(t_i)} \tag{5.6}$$

$C(x)$ steht für die Anzahl Vorkommnisse von x.

Der Maximum-Likelihood-Schätzwert für die Ausgabewahrscheinlichkeit eines Wortes w_i durch den Zustand t_i wird analog errechnet: Man zählt, wie häufig das Wort w_i vom Zustand t_i generiert wird und teilt durch die Anzahl der Zustände t_i. Formal:

$$P(w_i \mid t_i) = \frac{C(w_i, t_i)}{C(t_i)} \tag{5.7}$$

Wenn kein getaggtes Korpus, respektive ein zu kleines vorliegt, müssen die Parameter, das heisst Ausgabealphabet (Menge der Symbole), Tag-Set (Menge der Zustände respektive Wortarten und deren Abgrenzung), Start-[85], Ausgabe- und Übergangswahrscheinlichkeiten, anderweitig gewonnen werden.

Dazu wird ein zweiteiliges Verfahren angewandt. In einem ersten Schritt werden allen Parametern initiale Werte zugeordnet. Dies sind keine Zufallswerte, sondern sie werden gestützt auf ein Wörterbuch ermittelt.[86] Im zweiten Schritt wird das initialisierte Modell mittels des so genannten *Forward-Backward-Algorithmus*[87] trainiert, das heisst, die Parameter so lange verändert, bis die Werte optimal sind.

Mit oder ohne getaggtes Trainingskorpus erhält man also in beiden Fällen ein Hidden Markov Modell. Dieses kann dann auf neue, ähnliche Texte angewandt werden, um sie zu taggen.

2. Anwendungs-Phase

Für die Anwendung eines trainierten und somit optimierten HMMs auf einen zu taggenden Text wird folgende Formel[88] verwendet:

$$\hat{t}_{1,n} = argmax_{\ t_{1,n}}\ P(t_{1,n} \mid w_{1,n}) = argmax_{\ t_{1,n}} \prod_{i=1}^{n} P(w_i \mid t_i) P(t_i \mid t_{i-1}) \tag{5.8}$$

$\hat{t}_{1,n}$ steht dabei für die Tagfolge (also Wortartenfolge) für eine Sequenz von n Wörtern. Formel (5.8) sagt aus, dass $\hat{t}_{1,n}$ gleich demjenigen $t_{1,n}$ ist, das die Wahrscheinlichkeit

[85] Vgl. [CAR01], S. 119.

[86] Genauere Angaben über das Verfahren der Initialisierung können beispielsweise in [MAN99], S. 357ff, nachgelesen werden.

[87] Auch *Baum-Welch-Algorithmus* genannt. Für genauere Erläuterungen siehe beispielsweise in [CAR01], S. 129ff, oder [MAN99], Kapitel 9.

[88] Übernommen aus [MAN99], Formel (10.7), S. 347. Dort ist auch eine vollständige Herleitung aufgeführt.

$P(t_{1,n} \mid w_{1,n})$ maximiert. Wie bereits erwähnt lässt sich diese Wahrscheinlichkeit schätzen als Produkt aller Ausgabewahrscheinlichkeiten $P(w_i \mid t_i)$ und aller Übergangswahrscheinlichkeiten $P(t_i \mid t_{i-1})$. In einem trainierten HMM sind diese Wahrscheinlichkeiten vorhanden und können jetzt direkt eingesetzt werden.

Würde Formel (5.8) direkt implementiert werden, käme man auf sehr lange Rechenzeiten, da es eine Unmenge von zu berechnenden Pfaden gäbe. Üblicherweise wendet man deshalb bei der Implementierung den *Viterbi-Algorithmus*[89] an, mit dessen Hilfe die Rechenzeit erheblich verkürzt werden kann.

b) IdentiFinder: Variante eines Hidden Markov Modells

1. Konzeptuelles Modell

Statt Wortarten werden in IdentiFinder Named Entities[90] getaggt. Acht NE-Klassen (wobei die grösste Klasse NOT-A-NAME, also diejenige für alle Wörter, die keine NEs sind, miteingerechnet ist) stellen acht Zustände dar. Jedem Wort wird also eines der acht Tags zugeordnet; das Tag NOT-A-NAME wird dabei als "neutrales" Tag verstanden, so dass diejenigen Wörter, die damit getaggt werden, als ungetaggt gelten. Abbildung 5.6 zeigt diesen konzeptuellen Aufbau von IdentiFinder. Weitere Ausführungen zur Bildunterschrift finden sich in den anschliessenden Abschnitten.

2. Mikrostruktur: Bigramm-Sprachmodelle

Abweichend von einem Standard-HMM existieren zusätzlich zum übergeordneten HMM[93] wiederum "kleinere" HMMs, pro NE-Klasse eines. Eine Besonderheit dieser acht kleineren HMMs - sie werden *Bigramm-Sprachmodelle* (engl. *bigram language models*) genannt - ist, dass jedes gleich viele Zustände enthält, wie Wörter im spezifischen Vokabular vorhanden sind. Beispielsweise enthält das Bigramm-Sprachmodell für Personennamen so viele Zustände wie die Gesamtzahl der darin enthaltenen Vor-, Mittel- und Nachnamen[94]. Sinnigerweise heisst ein Zustand deshalb auch immer gleich wie das Wort, das er ausgibt. Aus dieser "eins-zu-eins"-Zuordnung folgt, dass jeder Zustand eine Ausgabewahrscheinlichkeit von genau eins hat. Somit sind zur Berechnung der Wahrscheinlichkeit einer Wortfolge w_1 bis w_n innerhalb einer NE-Klasse - wobei w_1 das erste Wort dieser NE-Klasse meint und

[89] Zur Erklärung des Viterbi-Algorithmus' siehe die Ausführungen in [MAN99], S. 99f.

[90] NEs wird hier im weiteren Sinne verstanden, nämlich gemäss MUC-7 Task Definition: Als NEs gelten die sieben Klassen Personen, Orte, Organisationen, Geldbeträge, Zeitpunkte, Datums- und Prozentangaben.

[92] Die Abbildung ist [BIK99] entnommen.

[93] Vgl. Abschnitt *3. Übergeordnetes HMM als Makrostruktur.*

[94] Auch durch Buchstaben abgekürzte Vor-, Mittel- und Nachnamen gelten als selbständige Wörter.

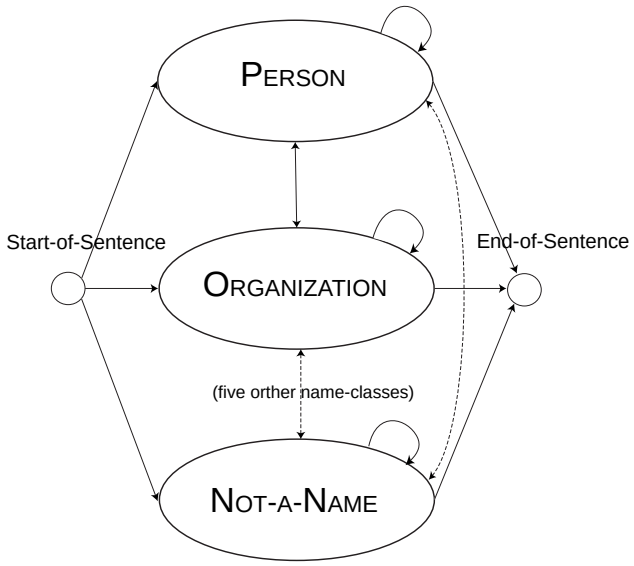

Abbildung 5.6: Konzeptuelles Modell von IdentiFinder. In der Mikrostruktur steht jede Ellipse für ein HMM für die entsprechende NE-Klasse respektive für die NOT-A-NAME-Klasse. In der Makrostruktur stellen die Ellipsen Zustände eines übergeordneten HMMs dar, wobei ein Zustand nicht nur ein einzelnes Wort, sondern auch NEs, die aus mehreren Wörtern bestehen, generieren kann. Aus Platzgründen wurden fünf NE-Klassen weggelassen und durch die gestrichelten Pfeile angedeutet.[92]

w_n das letzte - nur die Übergangswahrscheinlichkeiten von Bedeutung. Diese gehen jeweils von w_{i-1} nach w_i, wobei $w_1 \leq w_{i-1} < w_n$ und $w_1 < w_i \leq w_n$. Die Wahrscheinlichkeit für eine Wortfolge w_1 bis w_n innerhalb einer NE-Klasse kann somit nach Formel (5.9) geschätzt werden:

$$p\left(w_1, w_n\right) = \prod_{i=1}^{n} p\left(w_i \mid w_{i-1}\right) \tag{5.9}$$

$(w_i \mid w_{i-1})$ kann durch Einsetzen in Formel (5.6) geschätzt werden.[95] Zur Berechnung von $p\left(w_1\right)$, also der Übergangswahrscheinlichkeit zum ersten Wort einer NE, wird ein spezielles *Beginn-Wort*[96] verwendet.

[95]Dazu ist t durch w zu ersetzen. Diese Äquivalenz von Zustand t und Wort w kommt darum zu Stande, weil, wie bereits erwähnt, jedem Wort ein Zustand entspricht - eine Besonderheit dieser Bigramm-Sprachmodelle.

[96]Mehr, als dass dieses Beginn-Wort *+begin+ word* genannt wird, ist in den Berichten über IdentiFinder nicht vermerkt.

3. Übergeordnetes HMM als Makrostruktur

Um nun *im ganzen Text* Bezeichner für NEs zu finden und diese von den anderen Wörtern[97] zu unterscheiden, muss mit einem weiteren HMM, einem übergeordneten, gearbeitet werden. Dieses übergeordnete HMM hat acht Zustände, die den acht NE-Klassen entsprechen. Übergangs- und Ausgabewahrscheinlichkeiten sind etwas komplexer als im Standard-HMM.

Die Übergangswahrscheinlichkeit zwischen den NE-Klassen NC_{-1} und NC hängt nicht nur von den beiden NE-Klassen selbst ab, sondern zusätzlich vom (letzten)[98] ausgegebenen Wort von NC_{-1}, also von w_{-1}.[99] Formal:

$$P\left(NC \mid NC_{-1}, w_{-1}\right) \tag{5.10}$$

Grund für diese Erweiterung ist die Annahme, dass ein vorangehendes Wort ein Indikator für eine bestimmte NE-Klasse sein kann. Bekanntes Beispiel ist das Wort *Mr.* (w_{-1}), das indiziert, dass ein Wort w der NE-Klasse PERSONENNAME folgen wird.

Die Berechnung der Ausgabewahrscheinlichkeit beschränkt sich in der Makrostruktur auf diejenige des *ersten* Wortes in einer bestimmten NE-Klasse - danach befindet man sich (bis zum Übergang in die nächste NE-Klasse) in einem Bigramm-Sprachmodell (Mikrostruktur). In IdentiFinder ist das Besondere an der Ausgabewahrscheinlichkeit von w_{first}, des ersten Wortes einer NE-Klasse, dass sie nicht nur von der ausgebenden NE-Klasse NC, sondern auch von der vorangegangenen NE-Klasse NC_{-1} abhängig angenommen wird. Formal:

$$P\left(w_{first} \mid NC, NC_{-1}\right) \tag{5.11}$$

Auch hier liegt der Grund für die Erweiterung in der Annahme, dass die vorangehende NE-Klasse Einfluss auf die Ausgabe eines Wortes hat.

Formel (5.11) ist jedoch noch nicht vollständig. IdentiFinder teilt jedem Wort ein Merkmal f (*Feature*) zu, beispielsweise `allCaps`, wenn das Wort nur aus Grossbuchstaben besteht, oder `initCap`, wenn das Wort mit einem Grossbuchstaben beginnt. Die vollständige Formel zur Berechnung der Ausgabewahrscheinlichkeit des ersten Wortes einer bestimmten NE-Klasse lautet demnach:

$$P\left(\langle w, f\rangle_{first} \mid NC, NC_{-1}\right) \tag{5.12}$$

Mit dem Erweitern um Merkmale werden Indikatoren ausgenutzt, die sprachspezifisch sind. Deshalb ist die Zuteilung der Merkmale ebenfalls sprachenabhängig.

[97] Wörter, die keine NEs bezeichnen, werden wie bereits erwähnt der Klasse NOT-A-NAME zugordnet.

[98] Pro NE-Klasse können wie dargelegt mehrere Wörter ausgegeben werden.

[99] Es wird von der Vorstellung ausgegangen, dass das aktuelle Wort, also das erste Wort von NC, als w bezeichnet wird.

4. Zusammenarbeit der Mikro- und Makrostruktur

Wie im Basis-HMM wird die wahrscheinlichste NE-Klassen-Folge ($\boldsymbol{NC}$) für eine gegebene Wortfolge ($\boldsymbol{W}$) gefunden, indem sie maximiert wird. Gesucht ist also:

$$max\ P\ (\boldsymbol{NC} \mid\ \boldsymbol{W}) \tag{5.13}$$

Im Folgenden wird von einem generativen Modell ausgegangen, das heisst von der Annahme, dass das HMM die Wort- und NE-Klassen-Folge generiert. Gemäss Bayes gilt:

$$P\ (\boldsymbol{NC} \mid\ \boldsymbol{W}) = \frac{P(\boldsymbol{W},\ \boldsymbol{NC})}{P(\boldsymbol{W})} \tag{5.14}$$

Da $P(\boldsymbol{W})$ konstant ist, ist nur noch die Maximierung von $P(\boldsymbol{W},\ \boldsymbol{NC})$ nötig. $P(\boldsymbol{W},\ \boldsymbol{NC})$ meint lediglich die gemeinsame Wahrscheinlichkeit einer Wortfolge und einer NE-Klassen-Folge. Im IdentiFinder setzt sich die Berechnung von $P(\boldsymbol{W},\ \boldsymbol{NC})$ aus drei Schritten zusammen, wobei in jedem Schritt Wahrscheinlichkeiten erzeugt werden, die am Schluss miteinander zu multiplizieren sind. Wie die Werte, die für die einzelnen Wahrscheinlichkeiten eingesetzt werden müssen, ermittelt werden, wird in Abschnitt *5. Training: Geschätzte Wahrscheinlichkeiten* erläutert.

(i) Zuerst wird eine NE-Klasse NC gewählt, deren Übergangswahrscheinlichkeit wie in Formel (5.10) gezeigt, abhängig ist von der vorangehenden NE-Klasse NC_{-1} und vom vorangehenden Wort w_{-1}.

(ii) Darauf wird das erste Wort w_{first} innerhalb der gewählten NE-Klasse NC generiert. Die Ausgabewahrscheinlichkeit dieses Wortes ist bedingt durch die aktuelle NE-Klasse NC und die vorangegangene NC_{-1}. Die zur Berechnung nötige Formel (5.12) berücksichtigt diese Bedingtheiten.

(iii) Sich nun innerhalb der aktuellen NE-Klasse NC befindend, werden nun so lange die folgenden Wörter der Wortfolge ($\boldsymbol{W}$) generiert, bis man in ($\boldsymbol{W}$) auf ein Wort stösst, das einer anderen NE-Klasse angehört. Wie in Formel (5.9) gezeigt, ist jedes Wort w bedingt durch das vorangehende Wort w_{-1}. Für IdentiFinder muss Formel (5.9) angepasst und erweitert werden, nämlich um das Merkmal f und die aktuelle NE-Klasse NC. Die Formel für die Übergangswahrscheinlichkeiten lautet demnach:

$$P\ (\langle w, f\rangle \mid \langle w, f\rangle_{-1}, NC) \tag{5.15}$$

Um die Wahrscheinlichkeit zu berechnen, ob ein Wort das letzte Wort innerhalb einer NE-Klasse ist, benötigt man ein spezielles *+end+*-Wort, das durch eine weitere Formel eingeführt wird:

$$P\ (\langle +end+, \texttt{other}\rangle \mid \langle w, f\rangle_{final}, NC) \tag{5.16}$$

Das Merkmal `other` in Formel (5.16) wird Satzzeichen und allen Wörtern zugeordnet, denen kein anderes der vierzehn Merkmale zugeordnet werden kann.[100]

[100] Für eine vollständige Auflistung aller Merkmale siehe z.B. [Bik99].

Um in einem Satz die richtigen NE-Klassen zu finden, müssen also Mikro- und Makrostruktur ineinander greifen, das heisst, das übergeordnete HMM - Schritt (i) und (ii) - und die Bigramm-Sprachmodelle - Schritt (iii) - arbeiten zusammen. Die drei Schritte werden so lange wiederholt, bis die ganze Wortfolge ($\boldsymbol{W}$) generiert ist. Wie im Standard-HMM wird mittels Viterbi-Algorithmus diejenige NE-Klassen-Folge ($\boldsymbol{NC}$) gesucht, die $P(\boldsymbol{W, NC})$ maximiert.

Am besten lässt sich die Schätzung der Wahrscheinlichkeit einer Wort-NE-Klassen-Folge an einem Beispiel illustrieren.[101] Angenommen IdentiFinder trifft folgenden Satz an:

(10) *Mr. Jones eats.*

Jones gehört der NE-Klasse PERSONENNAMEN an, die anderen Wörter (inklusive des Punkts, der in diesem Fall auch als Wort angesehen wird) der NE-Klasse NOT-A-NAME.

Unter der Voraussetzung, dass IdentiFinder ausreichend trainiert wurde, würde dieser Wort-NE-Klassen-Folge die Wahrscheinlichkeit folgender Berechnung (Multiplikation mehrerer Faktoren, angedeutet durch Asterix *) zugewiesen werden, da diese Berechnung bei ausreichendem Tranining die höchste Wahrscheinlichkeit $P(\boldsymbol{W, NC})$ ergeben muss:

Schritt (i)	$P(\text{NOT-A-NAME} \mid \text{START-OF-SENTENCE}, +end+)$ *
Schritt (ii)	$P(Mr. \mid \text{NOT-A-NAME}, \text{START-OF-SENTENCE})$ *
Schritt (iii)	$P(+end+ \mid Mr., \text{NOT-A-NAME})$ *
Schritt (i)	$P(\text{PERSON} \mid \text{NOT-A-NAME}, Mr.)$ *
Schritt (ii)	$P(Jones \mid \text{PERSON}, \text{NOT-A-NAME})$ *
Schritt (iii)	$P(+end+ \mid Jones, \text{PERSON})$ *
Schritt (i)	$P(\text{NOT-A-NAME} \mid \text{PERSON}, Jones)$ *
Schritt (ii)	$P(eats \mid \text{NOT-A-NAME}, \text{PERSON})$ *
Schritt (iii)	$P(. \mid eats, \text{NOT-A-NAME})$ *
Schritt (iii)	$P(+end+ \mid ., \text{NOT-A-NAME})$ *
Schritt (i)	$P(\text{END-OF-SENTENCE} \mid \text{NOT-A-NAME}, .)$

5. Training: Geschätzte Wahrscheinlichkeiten

Wie im Standard-HMM werden die Wahrscheinlichkeiten mittels Maximum-Likelihood-Schätzwert ermittelt. Für die Formeln (5.10), (5.12)und (5.15) lauten die Gleichungen zur Berechnung der Wahrscheinlichkeiten demnach:

$$P\left(NC \mid NC_{-1}, w_{-1}\right) = \frac{c(NC, NC_{-1}, w_{-1})}{c(NC_{-1}, w_{-1})} \tag{5.17}$$

[101] Das Beispiel ist [BIK99] entnommen. Wie im Original wurden auch hier die Merkmale der Wörter nicht aufgeführt. Hinzugefügt wurde die Angabe, von welchem der drei Schritte ein Faktor für die Multiplikation herrührt.

$$P\left(\langle w, f\rangle_{first} \mid NC, NC_{-1}\right) = \frac{c(\langle w, f\rangle_{first}, NC, NC_{-1})}{c(NC, NC_{-1})} \tag{5.18}$$

$$P\left(\langle w, f\rangle \mid \langle w, f\rangle_{-1}, NC\right) = \frac{c(\langle w, f\rangle, \langle w, f\rangle_{-1}, NC)}{c(\langle w, f\rangle_{-1}, NC)} \tag{5.19}$$

Wobei $c(x)$ beziffert, wie häufig das Muster x im Trainingskorpus auftaucht.

6. Problem der ungenügenden Trainingsdaten

Trotz Training über grossen Korpora tauchen zwei Probleme häufig auf:

Unbekannte Wörter: Im zu taggenden Korpus kommen Wörter vor, die im Trainingskorpus nicht vorhanden waren.

Unbekannte Bigramme: Im zu taggenden Korpus kommen Zwei-Wortfolgen (*Bigramme*) vor, die im Trainingskorpus nicht vorhanden waren.

Diesen Problemen begegnet man mit so genannten *Back-off-Modellen* und *Smoothing*. In der Grundidee geht es darum, die Bedingungen zu lockern. Das kann heissen, dass beispielsweise in den Formeln die Merkmale nicht berücksichtigt werden müssen oder die Bedingungen bei den Wahrscheinlichkeiten weggelassen werden können.[102]

5.2.3 MENE

Quellen	[Bor98]; [Bor99]; [Rat98]; [Rat97]
Erkannte NEs/Eigennamen	Gemäss MUC-NE-Task Definition (vergleiche [Chi97a])
Sprachen	Englisch

Eine Forschungsgruppe an der *New York University* hat zur Teilnahme am NE-Task an der MUC-7 das *MENE*-System entwickelt. MENE steht für *Maximum Entropy Named Entity*. Wie der Name sagt, basiert das System auf der Verwendung von Maximaler Entropie.

a) Vorzüge der Maximalen Entropie

Das Ausnutzen von *Maximaler Entropie (ME)* ist eine statistische Methode, die den Vorzug hat, dass mit ihr eine *heterogene* Menge kontextueller Evidenz ausgenutzt werden kann,

[102] Genauere Informationen zu diesen Methoden finden sich in [Bik99], Seite 9ff.

um beliebige Klassifikationen vorzunehmen - in diesem Fall sollen Tokens[103] entweder als bestimmte Eigennamen (respektive als Teile von Eigennamen, die aus mehreren Tokens bestehen) oder als Nicht-Eigennamen klassifiziert werden: "Maximum entropy probability models offer a clean way to combine diverse pieces of contextual evidence in order to estimate the probability of a certain linguistic class occuring with a certain linguistic context."[104] Sogar sich überschneidende Evidenz kann problemlos nebeneinander ausgenutzt werden, sie brauchen nicht voneinander unabhängig zu sein.[105] Beispielsweise kann festgelegt werden, dass *"ein einzelner grossgeschriebener Buchstabe mit anschliessendem Punkt und darauffolgendem Familiennamen"* eine Evidenz für einen *Vornamen (Personennamen)* sein soll. Gleichzeitig kann auch *"etwas Grossgeschriebenes mit anschliessendem Familiennamen"* als weitere Evidenz für einen *Vornamen* definiert werden - diese Evidenz wird im Gegensatz zur ersteren auch für ganze Wörter gelten, schliesst aber einen einzelnen grossgeschriebenen Buchstaben mit Punkt nicht aus. Trifft man dann beim Klassifizieren eines Textes auf einen grossgeschriebenen Buchstaben mit anschliessendem Punkt und darauffolgendem Familiennamen, so trifft darauf sowohl die eine als auch die andere Evidenz zu, was insgesamt eine grössere Evidenz bedeutet, als wenn nur eine Evidenz zutreffen würde.[106]

Auch ist mit ME möglich, Evidenz zu verwenden, die sich widerspricht. Möglich wird dies dank einer *Gewichtung* der Evidenz in der Trainingsphase. So darf zum Beispiel die Evidenz *"'Mr.' vor grossgeschriebenem Wort deutet darauf hin, dass es sich beim grossgeschriebenen Wort um einen Personennamen handelt"* neben der Evidenz *"der Ausdruck 'Mr. Pickwick Pub'*[107] *ist ein Firmenname"* stehen. Trifft man beim Klassifizieren eines Textes auf den Ausdruck *Mr. Pickwick Pub*, auf den sowohl die eine als auch die andere Evidenz zutrifft, die aber widersprüchlich klassifizieren würden, kann dank der Gewichtung (und allenfalls dank weiterer auf den Ausdruck zutreffender Evidenz) entschieden werden, welche der verschiedenen Klassen die wahrscheinlichste ist. Da in der Trainingsphase der Ausdruck *Mr. Pickwick Pub* immer als Firmenname klassifiziert angetroffen wurde (Trefferquote von 100 Prozent), wird diese Evidenz das höhere Gewicht erhalten als diejenige, die für den Personennamen spricht, da diese keine Trefferquote von 100 Prozent aufweisen kann, weil sie für den *Mr. Pickwick Pub*-Fall nicht zutrifft.

[103] Der Tokenizer in MENE erkennt ein Token als eine Zeichenfolge zwischen Leerschlägen. Ein Token ist also ein einzelnes Wort, eine Zahl, eine Kombination der beiden ohne Leerschlag, evtl. mit anschliessendem Punkt, Bindestrich oder ähnlichem.

[104] [Rat98], S. 6.

[105] Vgl. [Rat98], S. 16: "The features in a maximum entropy model need not be statistically independent; all probability models in this thesis fully exploit this advantage by using overlapping and interdependent features".

[106] Wie verschiedene Evidenz, die im ME-Modell als *Features* dargestellt wird, gewichtet und miteinander verrechnet wird, wird später noch genau erklärt werden.

[107] *Mr. Pickwick Pub* ist der Name einer Bar-Kette im britischen Stil - daher der Namensteil *Pub* - die in der Schweiz in zwölf Städten mit ihren Pubs ansässig ist.

b) Trainingsphase

Das eingangs erwähnte Ziel der Klassifikation versucht ein ME-System mit der Methode zu erreichen, die Wahrscheinlichkeit zu berechnen, dass ein Token der *Klasse* f zugeordnet wird, wenn es im *Kontext* h auftaucht, also $p(f|h)$. Wie bei statistischen Methoden üblich, benötigt man zur Berechnung der Wahrscheinlichkeit ein annotiertes Trainingskorpus[108].

1. Klassen und Kontext

Die MENE-*Klassen* stehen in direktem Zusammenhang mit den Eigennamenklassen, die es im MUC-NE-Task zu finden gilt. Allerdings wird in MENE feiner unterteilt: Während im MUC-NE-Task ein Name sowohl aus einem als auch aus mehreren Tokens bestehen kann, wird in MENE jedes Token einzeln klassifiziert gemäss seiner Position im Eigennamen. MENE kennt vier Unterklassifizierungen: `x_start`, `x_continue`, `x_end`, `x_unique`, wobei `x` für die MUC-Eigennamenklasse steht. Da in MENE jedes Token klassifiziert werden soll, gibt es die zusätzliche Klasse `other`, um anzuzeigen, dass es sich beim so klassifizierten Token um keinen Teil eines Eigennamens handelt. Zum Beispiel würde der Satz *Jerry Lee Lewis flew to Paris* folgendermassen klassifiziert werden: `person_start`, `person_continue`, `person_end`, `other`, `other`, `location_unique`. Da am MUC-NE-Task insgesamt sieben NE-Klassen vorgegeben waren, wird in MENE mit vier mal sieben plus einer[109], also mit 29 Klassen gearbeitet.

Als *Kontext* werden in der ME - im weiteren Sinn - alle Informationen bezeichnet, die Einfluss auf die Klassifizierung eines Tokens haben. Anders ausgedrückt: Alle (nützlichen) Informationen, die bezüglich eines klassifizierten Tokens aus dem Trainingskorpus gewonnen werden können, bilden den Kontext dieses Tokens. Im engeren Sinn besteht ein Kontext normalerweise aus Wörtern; je nach Aufgabenstellung besteht er aus bloss einem Wort, oder aber aus mehreren Wörtern und vielleicht zusätzlich aus den zugeordneten Klassen. In MENE besteht ein Kontext jeweils nur aus dem klassifizierten Token mit seinem Index, welcher ermöglicht, bei Bedarf auch auf die davor und die danach stehendenden Tokens zuzugreifen.

2. Modellerstellung: Features, ME-Abschätzungsprozess und Gewichte

Die Berechnung der Wahrscheinlichkeit $p(f|h)$ in der ME ist abhängig von einer Menge von binär-wertigen Funktionen, so genannten *Features*. Features dienen der Abbildung der eingangs erwähnten Evidenz. "Binär-wertig" heisst, dass entweder auf 0 oder auf 1 abgebildet wird (Wertebereich). Wird auf 1 abgebildet, so ist das Feature *aktiv*, ansonsten

[108] Um ein annotiertes Trainingskorpus zu erhalten, kann man alle Tokens von Hand klassifizieren oder aber halbautomatisch - also mit einem Tagger und anschliessender manueller Kontrolle - arbeiten.

[109] Vier Unterklassen, sieben MUC-Klassen und eine `other`-Klasse.

ist es *nicht aktiv*. Im Definitionsbereich benötigt das Feature Paare von je einer Klasse f und einem Kontext h.

Folgende Beispiele von MENE-Features mögen der Veranschaulichung dienen:

$$g_1(h,f) = \left\{ \begin{array}{ll} 1: & \text{if current\_token\_capitalized}(h) = \text{true} \\ & \text{and } f = \texttt{location\_start} \\ 0: & \text{else} \end{array} \right\} \tag{5.20}$$

In Worten: Das Feature g_1 ist für den Kontext h (wobei h hier identisch mit dem klassifizierten Token ist) und die Klasse f dann aktiv, wenn im Trainingskorpus das klassifizierte Token grossgeschrieben ist (current_token_capitalized (h)) und als Beginn eines Ortsnamens klassifiziert wurde (f = `location_start`).

Ein anderes Feature g_2 für denselben Kontext h und dieselbe Klasse f könnte beispielsweise lauten:

$$g_2(h,f) = \left\{ \begin{array}{ll} 1: & \text{if Lexical\_View}(\text{token}_{-1}(h)) = \text{`}to\text{'} \\ & \text{and } f = \texttt{location\_start} \\ 0: & \text{else} \end{array} \right\} \tag{5.21}$$

In Worten: Das Feature g_2 ist für den Kontext h und die Klasse f dann aktiv, wenn im Trainingskorpus unmittelbar vor dem klassifizierten Token ($\text{token}_{-1}(h)$) das Wort '*to*' steht und das Token selbst (wie in Feature g_1) als Beginn eines Ortsnamens klassifiziert wurde.

Angenommen im Trainingskorpus stehe der folgende Satz:

(11) *She went to New York.*

Für das Token '*New*' (das dem Kontext h entspricht) wird sowohl das Feature g_1 als auch das Feature g_2 aktiv. Mittels dem *ME-Abschätzungsprozess* (engl. *ME estimation process*[110]) wird nun für jedes Feature g_i ein *Gewicht* α_i berechnet. Dank der Gewichtung der Features können auch Features herangezogen werden, die nur sehr schwache Evidenz für eine bestimmte Klassifizierung darstellen. Beispielsweise ist '*to*' vor dem zu klassifizierenden Token ein schwacher Hinweis darauf, dass dieses Token ein Ortsname ist. In einem regelbasierten NER-System würde man es wohl als zu riskant betrachten, eine Regel zu erstellen, die besagt, nach '*to*' folge ein Ortsname, weil es in zu vielen Fällen nicht zutrifft. Im Unterschied zu regelbasierten Methoden, wo eine einzige Regel entscheidet, wie ein Token klassifiziert werden soll, geben in der ME gewichtete Features nur (mehr oder weniger starke[111]) *Hinweise* darauf, dass ein Token einer bestimmten Klasse zugeordnet werden könnte. Erst durch die Verbindung mit anderen gewichteten Features wird klar, welcher Klasse ein Token dann tatsächlich zugeordnet werden kann. Meistens - jedoch, wie in Abschnitt *c) Anwendung auf neue Texte* erklärt werden wird, nicht immer - wird es diejenige Klasse sein, bei der die Verrechnung der Gewichte am grössten wird.

[110] Dem ME-Abschätzungsprozess verdankt die ME-Methode ihren Namen. Mehr dazu ist im Abschnitt *d) ME-Abschätzungsprozess* nachzulesen.

[111] Je höher das Gewicht eines Features ist, desto stärker der Hinweis auf die Klassifizierung.

Indem anhand eines Trainingskorpus das Gewicht jedes Features berechnet wurde, hat man ein *Sprachmodell* erstellt. Wie das Modell nun dazu verwendet werden kann, das Ziel, die Wahrscheinlichkeiten $p(f|h)$ zu berechnen, wird im folgenden Abschnitt erläutert werden.

c) Anwendung auf neue Texte

Dank der Gewichte α_i eines jeden Features g_i kann bei der Anwendung des Sprachmodells auf einen neuen Text berechnet werden, mit welcher Wahrscheinlichkeit p ein Token h der Klasse f zugeordnet werden kann. Die Formeln hierzu lauten:

$$p\left(f \mid h\right) = \frac{\prod_i \alpha_i^{g_i(h,f)}}{Z_\alpha(h)} \tag{5.22}$$

$$Z_\alpha\left(h\right) = \sum_f \prod_i \alpha_i^{g_i(h,f)} \tag{5.23}$$

Die Wahrscheinlichkeit $p(f|h)$ in Formel (5.22) ist das Produkt aus den Gewichten α_i aller aktiven Features $g_i(h, f)$ geteilt durch die Summe aller Produkte aus den α-Werten der aktiven Features für den Kontext h. In anderen Worten: Im Nenner wird eine Summe aus 29 Summanden[112] gebildet, wobei jeder Summand selbst ein Produkt der α-Werte aus den aktiven Features für eine bestimmte Klasse f gegeben den Kontext h ist.

Dass nur die Gewichte der aktiven Features miteinander multipliziert werden, wird dadurch erreicht, dass $g_i(h, f)$ als Exponent der Potenz eingesetzt wird. Ist das Feature aktiv, also der Exponent gleich 1, so ergibt α hoch den Exponenten wieder α. Ist das Feature nicht aktiv (Exponent gleich 0), so hat α hoch den Exponenten, was 1 ergibt, keinen Einfluss auf das Produkt.

Zur Veranschaulichung wieder ein Beispiel mit folgendem angenommenen Satz im zu taggenden Text:

(12) *He lives in New York.*

Im Gegensatz zu Beispielsatz (11) wäre im Satz (12) für das Token '*New*' von den Beispiel-Features g_1 und g_2 nur g_1 aktiv. Weitere Features könnten vorhanden sein:

$$g_3(h,f) = \left\{ \begin{array}{ll} 1: & \text{if Lexical\_View}(\text{token}_{-1}(h)) = \text{'}in\text{'} \\ & \text{and } f = \texttt{location\_start} \\ 0: & \text{else} \end{array} \right\} \tag{5.24}$$

$$g_4(h,f) = \left\{ \begin{array}{ll} 1: & \text{if Lexical\_View}(\text{token}(h)) = \text{'}New\text{'} \\ & \text{and } f = \texttt{location\_start} \\ 0: & \text{else} \end{array} \right\} \tag{5.25}$$

[112] Zur Erinnerung, woher die Zahl 29 stammt, siehe Abschnitt *1. Klassen und Kontext*, Seite 74.

$$g_5(h,f) = \left\{ \begin{array}{ll} 1: & \text{if current\_token\_all\_capitalized}(h) = \text{true} \\ & \text{and } f = \texttt{organization\_start} \\ 0: & \text{else} \end{array} \right\} \tag{5.26}$$

$$g_6(h,f) = \left\{ \begin{array}{ll} 1: & \text{if Lexical\_View}(\text{token}_{-1}(h)) = \text{`}in\text{'} \\ & \text{and } f = \texttt{organization\_start} \\ 0: & \text{else} \end{array} \right\} \tag{5.27}$$

$$g_7(h,f) = \left\{ \begin{array}{ll} 1: & \text{if Lexical\_View}(\text{token}(h)) = \text{`}New\text{'} \\ & \text{and } f = \texttt{organization\_start} \\ 0: & \text{else} \end{array} \right\} \tag{5.28}$$

Setzt man dies in Formel (5.22) ein, wird die Wahrscheinlichkeit, dass '*New*' zur Klasse `location_start` gehört, folgendermassen berechnet:

$$p\left(\texttt{location\_start} \mid \text{`}New\text{'}\right) = \frac{\alpha_1^1 * \alpha_2^0 * \alpha_3^1}{Z_\alpha(h)} = \frac{\alpha_1 * \alpha_3}{Z_\alpha(h)} \tag{5.29}$$

(5.29) veranschaulicht erst den Zähler von Formel (5.22). Der Nenner ist sehr viel umfangreicher als der Zähler. Bezogen auf die Features g_1 bis g_7 würde der Nenner wie in Formel (5.30) beginnen:

$$p\left(\texttt{location\_start} \mid \text{`}New\text{'}\right) = \frac{\alpha_1 * \alpha_3}{\alpha_1 * \alpha_3 + \alpha_4 * \alpha_6 * \alpha_7 + ... + ...} \tag{5.30}$$

Das Feature g_5 ist nicht aktiv, weil h nicht aus lauter Grossbuchstaben besteht, wie in g_5 verlangt wird. Ansonsten sind alle auf $f =$ `organization_start` bezogenen Features aktiv, so dass ihre entsprechenden α-Werte miteinander multipliziert werden und so im Nenner den zweiten Summand bilden können.

Da in diesem Minibeispiel viel zu wenige Features und somit zu wenige Gewichte α_i vorhanden sind, kann in (5.30) nur der Beginn eines Nenners gezeigt werden. Da in Wirklichkeit viel mehr Features vorhanden sind, wird der Nenner um ein Vielfaches länger werden, und zwar maximal[113] 29 Summanden lang, welche wiederum Produkte von Gewichten α_i sind.

Sind für jedes Token alle Wahrscheinlichkeiten berechnet, so wird von den 29 möglichen Wahrscheinlichkeitswerten, die pro Token ermittelt wurden, *nicht* einfach jedes Mal der grösste ausgewählt. Dies würde sonst zu Klassen-Abfolgen führen, die nicht möglich wären. Eine Zuweisung von zwei aufeinander folgenden Klassen ist oft Bedingungen unterworfen, wie zum Beispiel, dass auf eine `x_start`-Klasse keine `y_start`-Klasse folgen darf. Dieses Problem wird durch eine *Viterbi-Suche* gelöst, die durch das Gitterwerk der Wahrscheinlichkeiten denjenigen *Pfad* sucht, der die höchste aufsummierte Wahrscheinlichkeit hat und der die Bedingungen, wie Klassen aufeinander folgen dürfen, einhält.

[113] Es kann durchaus vorkommen, dass für einen Kontext h keines der Features einer bestimmten Klasse f aktiv ist. Dann trägt dieses Feature keinen Summand für den Nenner bei, und der Nenner besteht sodann aus weniger als 29 Summanden.

d) ME-Abschätzungsprozess

Entropie, ein Begriff, der auch in anderen Wissenschaftsgebieten wie zum Beispiel der Physik verwendet wird, hat mehrere Bedeutungen. Im vorliegenden Zusammenhang ist damit das Mass für den *durchschnittlichen Informationsgehalt einer Zufallsvariablen respektive einer Wahrscheinlichkeitsverteilung* gemeint. Dieser durchschnittliche Informationsgehalt (Entropie) ist dann am grössten (maximal), wenn alle Ereignisse einer Zufallsvariablen die gleich grosse Information enthalten, was genau dann der Fall ist, wenn die Ereignisse alle gleich wahrscheinlich sind (Gleichverteilung). Maximale Entropie bedeutet also Gleichverteilung, die Zuordnung gleich grosser Wahrscheinlichkeiten dort, wo kein Wissen über die Wahrscheinlichkeitsverteilung vorliegt.

Das zu erstellende Sprachmodell soll die bekannten Wahrscheinlichkeiten abbilden. Wo jedoch keine Anhaltspunkte darüber vorhanden sind, sollen keine Annahmen gemacht werden respektive angenommen werden, dass gleich verteilt ist.

Dieser Ansatz scheint umso plausibler, als er der moralischen Forderung nach Vorurteilslosigkeit nachkommt: Wenn einem über eine Sache keine Informationen vorliegen, soll man auch keine erdichten und schon gar keine (voreiligen) Urteile fällen, sondern offen und neutral an die Sache herangehen. Diese Verwandtschaft von Maximaler Entropie in der Naturwissenschaft und Vorurteilslosigkeit im Menschsein drückt [Jay91] sehr schön aus, wenn er über dieses Prinzip der ME schreibt:

> [...] it agrees with everything that ist known, but carefully avoids assuming anything that is not known. It is a transcription into mathematics of an ancient principle of wisdom.[114]

Annahme von Gleichverteilung, wenn keine Informationen vorhanden sind, ist das *Grundprinzip* beim ME-Abschätzungsprozess. Aufgrund seiner Komplexität wird bei der Beschreibung von ME-Systemen der *Algorithmus* selbst entweder nicht beschrieben, oder es wird der gutgemeinte Ratschlag erteilt, dass man sich mehr auf das Suchen von guten Features konzentrieren solle als auf das Verständnis eines komplizierten "Ab-der-Stange-Werkzeugs". Borthwick formuliert diesen Ratschlag wie folgt:

> One thing to bear in mind while reading the chapter is that although the mathematics behind M.E. can be quite complex, in the end the computation of the value of the parameters ai by the M.E. estimation routine ca be treated as a 'black box' (we used an off-the-shelf toolkit [...]). This allows the modeler to concentrate on selecting the features which best characterize the problem while letting the M.E. estimator worry about assigning the features their relative weights.[115]

[114] [Jay91], Seite 1.
[115] [Bor99], Seite 19.

Auch in dieser Arbeit soll der ME-Algorithmus aus oben genannten Gründen nicht näher beschrieben werden, sondern lediglich wiederholt werden, wozu er dient (Output) und welche Voraussetzungen dazu nötig sind (Input).[116]

Der ME-Algorithmus dient der Berechnung der Gewichte α_i aller Features g_i, womit ein Sprachmodell erstellt werden kann, das sodann auf neue Texte zur Klassifizierung angewandt werden kann. Als Input wird ein von Hand getaggtes Trainingskorpus benötigt: Für jedes Token muss ein Kontext h bestimmt werden und das Token muss einer der vorgängig festgelegten Klassen zugeordnet sein. In MENE wird mit 29 Klassen und 320'000 Kontexten gearbeitet (zur Erinnerung: in MENE ist ein Kontext eines klassifizierten Tokens mit dem Token identisch). Nebst dem getaggten Trainingskorpus braucht der ME-Algorithmus natürlich die zu gewichtenden Features. Für MENE wurden insgesamt 22'000 Features verwendet. Mehr zu den Features siehe im folgenden Abschnitt *e) Features: Typen und Gewinnung*.

e) Features: Typen und Gewinnung

Entscheidend für den Erfolg eines ME-Systems ist die Wahl guter Features respektive die Wahl guter *Typen*[117] von Features. Dieser Umstand kann verglichen werden mit der Wichtigkeit guter Regeln in einem regelbasierten System.

Für MENE wurden acht Feature-Typen verwendet, die fünf wichtigsten seien kurz erläutert. Daran anschliessend eine Beschreibung, wie Features gewonnen werden.

1. Binäre Features

Während alle Features einen binären Output (0 oder 1) erzeugen, zeichnen sich *binäre Features* (engl. *Binary Features*) zusätzlich dadurch aus, dass in ihnen Eigenschaften für den Kontext h (und somit für das aktuelle Token) formuliert sind, die entweder gelten oder nicht gelten. Beispiele für solche Eigenschaften (die "von Hand" ermittelt werden müssen): "*Das Token besteht ausschliesslich aus grossgeschriebenen Buchstaben*" oder "*Das Token besteht nur aus Ziffern*" und so weiter. Beispiele für binäre Features sind g_1 und g_5.

2. Lexikalische Features

MENEs *lexikalische Features* (engl. *Lexical Features*) sind die wichtigsten aller acht Typen.

[116] Wer dennoch genau wissen möchte, wie der ME-Abschätzungsprozess funktioniert, dem sei der vielzitierte Aufsatz *A Simple Introduction to Maximum Entropy Models for Natural Language Processing*, [Rat97], zur Lektüre empfohlen.

[117] In [Bor98] und [Bor99] wird von "Feature *classes*" gesprochen. Da der Begriff *classes* gleichzeitig aber auch für NE-Einteilung (NE-Klassifizierung) verwendet wird, wird hier zur Vermeidung von Verwechslungen der Ausdruck *Feature-Typen* verwendet.

Lexikalische Features betrachten die umgebenden Tokens eines getaggten Tokens, und zwar jeweils die beiden davor und danach stehenden Tokens, ein Fünfer-Fenster demnach. g_8 ist ein Beispiel eines lexikalischen Features:

$$g_8(h,f) = \left\{ \begin{array}{ll} 1: & \text{if Lexical\_View}(\text{token}_{-1}(h)) = \text{`}Mr\text{'} \\ & \text{and } f = \texttt{person\_unique} \\ 0: & \text{else} \end{array} \right\} \tag{5.31}$$

g_8 würde beispielsweise für *Mr. Borthwick* aktiv sein. Auch g_2, g_3, g_4, g_6 und g_7 sind Beispiele für lexikalische Features.

3. Wörterbuch-Features

Auch *Wörterbuch-Features* (engl. *Dictionary Features*) werden in MENE verwendet. Ein Eintrag in einem Wörterbuch kann sowohl aus einem als auch aus mehreren Tokens bestehen. In einem Vorverarbeitungsschritt werden das Trainingskorpus respektive die zu taggenden Texte gemäss den Einträgen in den Wörterbüchern mit folgenden Merkmalen bezeichnet: `start`, `continue`, `end`, `unique`, `other`. Wäre beispielsweise *British Airways* ein Eintrag in einem Wörterbuch, so würde in den Texten die Phrase *on British Airways Flight 962* folgendermassen markiert: `other`, `start`, `end`, `other`, `other`. Diese vorgängigen Markierungen werden dann in den Wörterbuch-Features ausgenutzt, wie das folgende Feature-Beispiel zeigt:

$$g_9(h,f) = \left\{ \begin{array}{ll} 1: & \text{if First\_Name\_Dictionary}(\text{token}_0(h)) = \texttt{unique} \\ & \text{and } f = \texttt{person\_start} \\ 0: & \text{else} \end{array} \right\} \tag{5.32}$$

Angenommen im "First_Name_Dictionary" (ein Wörterbuch mit Vornamen, daneben gibt es weitere Wörterbücher zur NE-Erkennung, vergleiche Tabelle 5.6) wäre der Vorname *Richard* eingetragen, so wäre g_9 für *Richard M. Nixon* im Trainingskorpus aktiv.

4. Features für externe Systeme

Mit *Features für externe Systeme* (engl. *External System Features*) wird die Möglichkeit geschaffen, MENE mit den Ergebnissen anderer NER-Systeme, auch regelbasierter, zu kombinieren und so die Leistung zu steigern.[118] Für den MUC-7-Wettbewerb wurde dies auch getan: MENE benutzte als Input den Output des regelbasierten Proteus-Systems, mit dem die New York University an MUC-6 teilgenommen hatte.[119] Wie der Output externer

[118] Genauere Angaben zur Leistungssteigerung mittels Features für externe Systeme finden sich im Kapitel 6.

[119] Unterdessen war Proteus noch erweitert und verbessert worden.

Wörterbuch	**Anzahl Einträge**	**Daten-Quelle**	**Beispiele**
Vornamen	1245	www.babyname.com	John; Julie; April
Organisationsnamen	10300	www.marketguide.com	Exxon Corporation
Organisationsnamen ohne Suffixe	10300	"Organisationsnamen" bearbeitet durch Perl-Skript	Exxon
Colleges und Universitäten	1225	www.utexas.edu/world/univ/alpha/	New York University; Oberlin College
Suffixe von Organisationsnamen	244	Informanten-Quelle	Inc.; Incorporated; AG
Abkürzungen US-Staaten	50	www.usps.gov	NY, CA
Welt-Regionen	14	www.yahoo.com	Africa; Pacific Rim

Tabelle 5.6: In MENE verwendete Wörterbücher, um Wörterbuch-Features zu erstellen

Systeme in die Feature-Form von MENE eingebaut werden kann, zeigt das Feature g_{10}:

$$g_{10}(h,f) = \left\{ \begin{array}{ll} 1: & \text{if Proteus\_System\_Tag}(\text{token}_0(h)) = \texttt{person\_start} \\ & \text{and } f = \texttt{person\_start} \\ 0: & \text{else} \end{array} \right\} \tag{5.33}$$

g_{10} wäre beispielsweise in *Richard M. Nixon* für den Vornamen *Richard* aktiv, falls Proteus *Richard* korrekt als `person_start` getaggt hat. Wichtig ist hier noch anzumerken, dass MENE auch ein anderes Tag setzen kann als dasjenige, das vom externen System vergeben wurde. MENE hat die Möglichkeit, stereotype Fehler des externen Systems zu erkennen und durch Vergabe des richtigen Tags (an f) den Fehler zu korrigieren.

5. Referenz-Auflösungs-Features

Das *Referenz-Auflösungs-Feature* (engl. *Reference Resolution Feature*) ist ein spezielles Feature, das erst in einem zweiten Durchlauf (engl. *Run*) zum Zuge kommt, da es sich auf bereits gefundene Namen abstützt. Prinzip und Ziel sind dieselben wie bei Koreferenz-Regeln in regelbasierten Systemen[120]: Hat man mittels anderer Features einen Eigennamen gefunden, der in ähnlicher Form auch woanders im Text auftaucht, möchte man den bereits gefundenen dazu nutzen, den ähnlichen auch zu finden. Dies geschieht, indem das Referenz-Auflösungs-Feature prüft, ob ein Wort oder eine Wortfolge eine geordnete Untermenge eines bereits gefundenen Namens darstellt. Wäre beispielsweise die Wortfolge *Georg Herbert*

[120]Vgl. bspw. in Unterkapitel 5.1.2, Abschnitt *b) Koreferenz-Auflösung*, (Seite 32).

Walker Bush schon als Personenname erkannt worden, so würde das Referenz-Auflösungs-Feature auch bei *Georg Bush*, *Georg*, *Georg Herbert Bush*, aber nicht bei *Herbert Georg* feuern. Auf dieselbe Art können Abkürzungen gefunden werden.[121]

6. Gewinnung der Features

Sind die Typen von Features einmal bestimmt - wobei dies den kreativen und schwierigen Teil der Arbeit darstellt[122] - so können die einzelnen Features pro Typ einfach gefunden werden. Pro Feature-Typ wird ein *Feature-Selektions-Algorithmus* eingesetzt, der zuerst für jedes Token alle Features dieses Typs bestimmt. Für die lexikalischen Features beispielsweise wird für jedes Token das erwähnte Fünfer-Fenster geöffnet und dann ein Feature für jede vorhandene Kombination erstellt (das heisst pro Token werden $29 * 5$ mögliche Features erstellt). Von all den so erstellten möglichen Features werden nur diejenigen beibehalten, die im Trainingskorpus mindestens drei Mal aktiv sind. Auch dieses Auswahlverfahren ist im Feature-Selektions-Algorithmus eingebaut. Handarbeit bleibt beispielsweise bei den Wörterbuch-Features die Erstellung der Wörterbücher oder bei den binären Features das Finden der nützlichen Eigenschaften eines Tokens (beispielsweise muss man herausfinden, dass die Information, ob ein Wort gross- oder kleingeschrieben ist, eine nützliche ist).

5.2.4 Entscheidungs-Bäume

Quellen	[Qui86]; [Qui93]; [Mit97]; [Sek98a]; [Sek98b]; [Ben97]; [Gal96]
Erkannte NEs/Eigennamen	Gemäss MUC-NE-Task Definition (vergleiche [Chi97a])
Sprachen	Englisch und Japanisch

Systeme, die im Bereich der NER mit *Entscheidungs-Bäumen* (engl. *Decision Trees*) arbeiten, beruhen auf einem Algorithmus namens *ID3* respektive auf dessen Nachfolger *C4.5*, beide von Ross J. Quinlan entwickelt[123]. Im Folgenden soll deshalb zuerst der Basis-Algorithmus von Quinlan erläutert werden, um das Grundprinzip von Entscheidungs-Bäumen aufzuzeigen. Danach wird eine Anwendung von Entscheidungs-Bäumen für NER erklärt werden.

[121] Für genauere Erläuterungen, insbesondere für ein Beispiel eines Referenz-Auflösungs-Features, siehe [Bor99], Seite 44ff.

[122] Warum die Leistung des Systems in erster Linie von der Intuition des Linguisten abhängt, kann in 6.2.3 nachgelesen werden.

[123] Vgl. [Qui86] und [Qui93].

a) Basis-Algorithmus von Quinlan

Ziel eines Entscheidungs-Baums ist es, *Instanzen*, die durch *Attribute* und *Werte* beschrieben werden können, zu klassifizieren.[124] Dem dabei wichtigen Umstand, dass bei der Klassifizierung einer Instanz deren Attribute in der richtigen Reihenfolge geprüft werden müssen, trägt die Architektur des Entscheidungs-Baums Rechnung.[125]

1. Instanzen als Mengen von Attribut-Wert-Paaren

Ein *Attribut*, respektive das Prüfen desselben, wird im Entscheidungs-Baum als *Knoten* dargestellt, die einem Attribut entsprechenden *Werte* als von seinem Knoten nach unten führende *Kanten* und die *Klassen*, in welche die Instanzen eingeteilt werden sollen, stehen in den *Blättern*. Jede Instanz wird als Menge von Attribut-Wert-Paaren aufgefasst. Klassifiziert wird nun eine Instanz, indem im Entscheidungsbaum derjenige Pfad abgeschritten wird, der ihr entspricht, bis man in einem Blatt anlangt, das sodann die gesuchte Klasse angibt. Damit ein Pfad einer Instanz entspricht, ist es nicht nötig, dass jedes Attribut der Instanz geprüft wird. Entscheidend ist jedoch, das alle Attribut-Wert-Paare des Pfades eine (unechte) Teilmenge der Menge der Attribut-Wert-Paare der Instanz bilden.

Abbildung 5.7 zeigt einen Entscheidungs-Baum, der auf Grund der "Wetter-Attribute" *Ausblick*, *Feuchtigkeit* und *Wind* und deren Werte entscheidet, ob Tennis gespielt werden soll (Klasse *ja*) oder nicht (Klasse *nein*). Eine Instanz kann beispielsweise folgendermassen aussehen:

(13) *Ausblick = sonnig ; Temperatur = heiss ; Feuchtigkeit = hoch ; Wind = stark*

Instanz (13) wird im Entscheidungsbaum in Abbildung 5.7 als *nein* klassifiziert (das heisst, bei dieser Wetterlage wird nicht Tennis gespielt), weil auf Grund ihrer Attribut-Wert-Paare *Ausblick = sonnig ; Feuchtigkeit = hoch* im Entscheidungsbaum der Pfad ganz links abgeschritten wird. Hier werden drei Dinge kenntlich:

- Das Attribut *Temperatur* ist im Entscheidungsbaum nicht vorhanden. Dies kommt daher, dass sich bei der Erstellung des Entscheidungsbaums gezeigt hat, dass es keinen Einfluss auf die Klassifizierung hat.
- Das Attribut *Wind* ist im Entscheidungsbaum zwar vorhanden, spielt aber bei der Klassifizierung einer Instanz keine Rolle, wenn diese als Wert für das Attribut *Ausblick* nicht *regnerisch* hat.

[124] Im Falle der Eigennamen-Erkennung sind die Instanzen Tokens, die aufgrund von Attributen wie zum Beispiel Wortart oder Schreibweise (Werte: gross oder klein) in Klassen von Eigennamen eingeteilt werden - wobei auch eine Klasse *Kein Eigenname* existiert).)

[125] Weshalb die Reihenfolge von Bedeutung ist, wird zu Beginn von Abschnitt *2. Erstellen eines Entscheidungsbaums* erläutert werden.

[127] Die Abbildung wurde [Mit97] entnommen und auf Deutsch übersetzt.

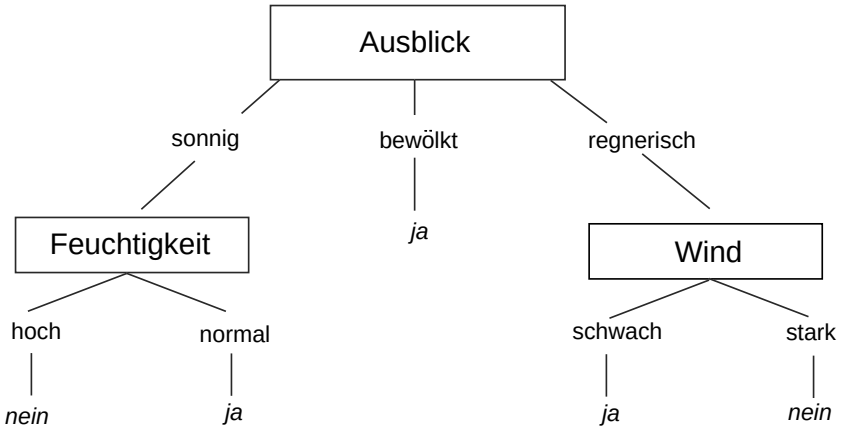

Abbildung 5.7: Entscheidungs-Baum fürs Tennisspiel: Dieser Baum klassifiziert Instanzen - zum Beispiel Samstage - danach, ob sie sich zum Tennisspiel eignen oder nicht. Ein Samstag wird durch (die in den Knoten vorkommenden) "Wetter-Attribute" und deren Werte charakterisiert. Klassifiert wird ein Samstag, indem der Pfad, der seinen Attribut-Wert-Paaren entspricht, abgeschritten wird; die Klasse, der ein Samstag dann angehört, steht im entsprechenden Blatt (in diesem Fall gibt es eine *ja*- und eine *nein*-Klasse.)[127]

- Hier liegen zwar nur zwei Klassen vor - *ja* und *nein* - jedoch sind die Pfade, die zur Klassifizierung führen, zahlreicher. So kann beispielsweise der Pfad (also die Menge von Attribut-Wert-Paaren) *Ausblick = sonnig ; Feuchtigkeit = hoch*[128] gleich klassifiziert werden, wie der Pfad *Ausblick = regnerisch ; Wind = stark*[129].

Letzterer Punkt wird nicht erstaunen, ist es doch auch der Zweck von Klassifizierung, in einer Menge von Elementen nicht nur die gleichen, sondern auch die *gleichartigen* Elemente in Untermengen (Klassen) zusammenzufassen. Im Falle des "Tennis-Entscheidungs-Baums" sind all diejenigen Wetterlagen gleichartig, die fürs Tennisspiel günstig sind (*ja*)- respektive alle die, die sich nicht zum Tennisspielen eignen (*nein*). Analog können auch in der NER Tokens mit verschiedenen Ketten von Attribut-Wert-Paaren gleich klassifiziert werden. So ist es beispielsweise denkbar, dass ein Token mit der Attribut-Wert-Paar-Kette (14) ebenso als Nachname klassifiziert wird wie das Token mit den Attribut-Wert-Paaren (15).[130]

[128] In Abbildung 5.7 der Pfad ganz links.
[129] In Abbildung 5.7 der Pfad ganz rechts.
[130] In realen Systemen sind die Attribut-Wert-Paar-Ketten meist sehr viel länger; da es hier nur darum geht, gleiche Klassifikation bei verschiedenen Attribut-Wert-Paar-Ketten deutlich zu machen, reicht ein verkürztes Beispiel.

(14) *Schreibweise aktuelles Token = gross ; Schreibweise vorgängiges Token = gross ; Typ vorgängiges Token = Vorname*[131]*; Schreibweise nachfolgendes Token = klein*

(15) *Schreibweise aktuelles Token = gross ; Schreibweise vorgängiges Token = klein ; Typ vorgängiges Token = Berufsbezeichnung ; Typ nachfolgendes Token = Satzendpunkt*

Pfad (14) könnte zum Beispiel in *Barbara Partee writes [...]* das Token *Partee* als Nachnamen erkennen, Pfad (15) dagegen in *[...] writes the researcher Partee.*

2. Erstellen eines Entscheidungs-Baums

Eingangs wurde erwähnt, dass die Reihenfolge, in der die Attribute geprüft werden, wichtig sei und sich in der Architektur des Baums niederschlage. Weshalb die Reihenfolge wichtig ist, lässt sich anschaulich erklären anhand eines Vergleichs mit einem Berufs-Rate-Spiel, in dem nur Ja-Nein-Fragen gestellt werden dürfen. Will man herausfinden, welchen Beruf das Gegenüber hat, so wird man kaum mit einer Frage beginnen wie beispielsweise derjenigen, ob es sich um den Beruf Tramchauffeur handle. Viel eher wird man vorerst allgemeinere Fragen stellen wie zum Beispiel, ob das Gegenüber selbständig sei. Lautet die Antwort *Nein*, so sind alle Berufe selbständiger Art im weiteren Frageverlauf bereits ausgeschlossen. Die Antwort *Nein* auf die Tramchauffeur-Frage hingegen schliesst nur einen einzigen Beruf für den weiteren Frage-Verlauf aus, was bei einer Fortsetzung derartiger Fragen dazu führen würde, dass das Spiel sehr viel länger dauerte oder im Extremfall zu gar keinem Ende fände, weil Anhaltspunkte, die zur Lösung führen könnten, fehlten.

Auch wenn das Gegenüber die Selbständigkeits-Frage mit *Ja* beantwortet, wird dies dem Fragenden einiges nützen, da die Menge an Berufen selbständiger Art in etwa gleich gross ist wie diejenige an unselbständiger Art.[132] Ideal ist also eine Ja-Nein-Frage, die bei beiden Antworten die Menge der in Frage kommenden Berufe halbiert, weil damit ein maximaler *Informationsgewinn* (engl. *information gain*) erreicht wird. Genereller (und nun in Bezug zu den Entscheidungs-Bäumen) ausgedrückt: Den grössten Informationsgewinn bringt ein Attribut, das bei Zuweisung eines jeden Werts die Menge der in Frage kommenden Klassen in gleich grosse Mengen teilt.[133] Für die Erstellung eines Entscheidungs-Baums bedeutet dies also, dass in jedem Knoten dasjenige Attribut stehen muss, das an dieser Stelle den grössten Informationsgewinn erbringt.

[131]Da in Entscheidungs-Baum-Systemen häufig auch mit Listen (zum Beispiel Listen mit Vornamen) gearbeitet wird, kann bereits an dieser Stelle bestimmt werden, dass ein bestimmtes Token ein Vorname ist.

[132]Natürlich wäre der ratenden Person auch und vor allem bei einer *Ja*-Antwort auf die Tramchauffeur-Frage geholfen - da somit das Ziel bereits erreicht ist - nur ist die Wahrscheinlichkeit, dass diese Frage mit *Ja* beantwortet wird, extrem klein.

[133]Hiermit sollte der Zusammenhang zwischen der spezifischen Terminologie fürs Berufs-Rate-Spiel und der allgemeinen klar werden: Fragen entsprechen Attributen (und stehen in den Knoten), Antworten entsprechen Werten (Kanten) und Berufe sind die Klassen (Blätter).

Informationsgewinn berechnet man, indem man bei der Erstellung eines Knotens für jedes in Frage kommende Attribut misst, wie gut, das heisst wie genau, es *alleine* die Instanzen in ihre Klassen einzuteilen vermag. Es wird also gemessen, wie genau sich die Werte eines Attributs mit den (End-)Klassen in den Blättern decken. Das beste Attribut ist dasjenige, bei welchem die Übereinstimmung möglichst gross ist, das heisst, bei welchem die Aufteilung durch die Werte der Aufteilung in die Klassen am ähnlichsten ist. Dieses Attribut wird dann für den bei der Erstellung aktuellen Knoten gewählt und danach die möglichen, nach unten gerichteten Kanten (welche die Werte des soeben eingefügten Attributs darstellen) gezogen und für den nächsten Knoten an deren unterem Ende diese Berechnung mit den übriggebliebenen Attributen wiederholt. Ein Baum wird also von oben nach unten (engl. *top down*) erstellt, und zwar so, dass ein Attribut nach dem anderen "aufgebraucht" wird. Wie die eben beschriebene Berechnung des Informationsgewinns formalisiert wird, soll im Folgenden dargelegt werden.

Wichtigstes Mass zur Berechnung des Informationsgewinns ist die *Entropie*[134]. Diese wird nach folgender Formel berechnet:

$$Entropie(S) \equiv -\sum_{i=1}^{c}(p_i \; log_2 p_i) \tag{5.34}$$

wobei S eine Sammlung von Instanzen ist, die in c verschiedene Klassen eingeteilt werden. p_i steht für relative Häufigkeit und meint die Anzahl Instanzen, die der Klasse i angehören, geteilt durch die Gesamtzahl aller Instanzen.

Zur Veranschaulichung ein kleines Beispiel: S sei eine Sammlung von 14 Instanzen, die in 2 Klassen eingeteilt werden (vergleiche in Abbildung 5.7 das Beispiel des Tennis-Entscheidungs-Baums, wo in die zwei Klassen *ja* und *nein* eingeteilt wird). Angenommen 9 Instanzen werden in die Ja-Klasse eingeteilt, 5 in die Nein-Klasse (im Folgenden notiert als [9+, 5-]). Die Entropie dieser Verteilung wird dann folgendermassen berechnet:

$$\begin{aligned} Entropie([9+, 5-]) &= -\ (9/14)\ \log_2\ (9/14) - (5/14)\ \log_2\ (5/14) \\ &= 0.940 \end{aligned} \tag{5.35}$$

Ausgehend von der Entropie lässt sich nun der gesuchte Informationsgewinn berechnen. Dieses Mass für die Effektivität eines Attributs, Instanzen zu klassifizieren, entspricht der *erwarteten Reduktion an Entropie*, wenn die Instanzen gemäss dem in Frage stehenden Attribut in Klassen aufgeteilt werden. Formaler: Der Informationsgewinn $Gewinn(S, A)$ eines Attributs A relativ zu einer Sammlung S von Instanzen wird definiert als:

$$Gewinn(S, A) \equiv Entropie(S) - \sum_{w \in Werte(A)} \frac{|S_w|}{|S|}\, Entropie(S_w) \tag{5.36}$$

$Werte(A)$ ist die Menge aller möglichen Werte für das Attribut A, und S_w ist die Untermenge von S, für die das Attribut A den Wert w hat. Formel (5.36) lässt sich folgendermassen interpretieren: Der erste Term ist die Entropie der ursprünglichen Instanzen-Sammlung

[134]Zum Begriff *Entropie* siehe die Erklärungen in 5.2.3, Abschnitt *d) ME-Abschätzungsprozess.*

S (wie in Beispielberechnung (5.35) vorgeführt), der zweite Term die erwartete Entropie, nachdem die Instanzen der Sammlung S gemäss Attribut A klassifiziert wurden.

Auch hier wieder ein Beispiel: Will man für die Erstellung des Tennis-Entscheidungs-Baums berechnen, welches Attribut den grössten Informationsgewinn ergibt und somit in der Wurzel stehen muss, so ist $Gewinn(S, A)$ für jedes Attribut zu berechnen, um darauf dasjenige mit dem grössten Gewinn zu wählen. Die folgende Berechnung für das Attribut *Wind* soll stellvertretend für die Berechnung aller Attribute stehen (zur Erinnerung: im Falle des Tennis-Entscheidungs-Baums sind es die drei Wetter-Attribute *Ausblick, Feuchtigkeit, Wind*). Wie in der vorigen Beispielberechnung (Formel (5.35)) wird von 14 Instanzen ausgegangen, 9 positiv, 5 negativ klassifiziert. Weiter wird angenommen, dass 6 der positiven Instanzen und 2 der negativen Instanzen für das Attribut *Wind* den Wert *schwach* aufweisen ($Wind = schwach$), die restlichen $Wind = stark$. Die Berechnung des Informationsgewinns für die 14 Instanzen bezüglich des Attributs *Wind* lautet demnach:

$$
\begin{aligned}
Werte(Wind) &= schwach, stark \\
S &= [9+, 5-] \\
S_{schwach} &= [6+, 2-] \\
S_{stark} &= [3+, 3-]
\end{aligned}
$$

$$
\begin{aligned}
Gewinn(S, Wind) &= Entropie(S) - \textstyle\sum_{w \in \{schwach, stark\}} \frac{|S_w|}{|S|} Entropie(S_w) \\
&= Entropie(S) - (8/14) Entropie(S_{schwach}) \\
&\quad - (6/14) Entropie(S_{stark}) \\
&= 0.940 - (8/14) 0.811 - (6/14) 1.00 \\
&= 0.048
\end{aligned}
\tag{5.37}
$$

Wie die Entropie für die Gesamtheit der Instanzen ($Entropie(S)$) werden auch die Werte für $Entropie(S_{schwach})$ respektive $Entropie(S_{stark})$ mit Hilfe der Formel (5.34) berechnet.

Nachdem im Tennis-Entscheidungs-Baum auf dieselbe Art die Informationsgewinne für die Attribute *Ausblick* und *Feuchtigkeit* berechnet worden sind, hat sich ergeben, dass das Attribut *Ausblick* den grössten Gewinn ergibt und ist somit zur Wurzel des Baums geworden.[135]

b) Anwendung von Entscheidungs-Bäumen auf NER

Das Grundprinzip von Entscheidungs-Bäumen, das heisst der Basis-Algorithmus, kann nun auf verschiedene Arten variiert werden. Entscheidungs-Baum-Algorithmen unterscheiden sich nicht nur von Anwendungsart zu Anwendungsart, sondern auch innerhalb einer bestimmten Anwendungsart. In der Anwendungsart NER wurden für die vorliegende Arbeit mehrere Entscheidungs-Baum-Systeme ermittelt, die in ihrer Funktionsweise in einigen Punkten voneinander abweichen. Im Folgenden soll ein System genauer vorgestellt werden,

[135] Wer genauere Angaben zu Zahlen wünscht, findet in [Mit97], Seite 59f, ein schönes Berechnungsbeispiel (*3.4.2 An Illustrative Example*).

für ein weiteres soll, stellvertretend für alle anderen, angegeben werden, worin es sich vom ersten unterscheidet.

Mit einem NER-System fürs Japanische nahm an der MUC-7 der japanische Forscher Satoshi Sekine von der *New York University* teil.[136] Bisher wurden in vorliegender Arbeit vor allem Systeme fürs Englische vorgestellt. Im Folgenden wird ein System zur NER im Japanischen erläutert.

1. Klassifizierung

Sekines NER-System arbeitet mit denselben 29 Klassen wie MENE.[137] Allerdings stehen in den Blättern nicht einfach einzelne Klassen, sondern die Wahrscheinlichkeiten für jede der 29 Klassen, falls diese grösser als Null ist. Die Klassifizierung ist also nicht deterministisch, in den Blättern stehen Wahrscheinlichkeitsverteilungen. Damit kann denjenigen Tokens Rechnung getragen werden, die zwar durch die gleichen Attribut-Wert-Paare beschrieben werden (und eventuell sogar gleich aussehen), aber verschieden getaggt werden. So kann zum Beispiel das Token *Matsushita* mal als Organisationsname, mal als Personenname und mal als Ortsname getaggt sein. Die Wahrscheinlichkeiten entsprechen den relativen Häufigkeiten im Trainingskorpus. Der Grund, weshalb nicht einfach jeweils dasjenige Tag mit der grössten Wahrscheinlichkeit im Blatt steht, ist derselbe wie bei MENE beschrieben[138]: Es könnten inkonsistente Klassen-Abfolgen zu Stande kommen (zum Beispiel darf auf eine `other`-Klasse[139] keine `end`-Klasse folgen). Wie schon in MENE gesehen, ist es ein Viterbi-Algorithmus, der für die definitive Klassifizierung sorgt.[140]

2. Attribute

Drei Arten von Attributen werden verwendet:

- *Schriftzeichen-Typen* (engl. *character type information*)
  Hier werden folgende Schriftzeichen-Typen unterschieden: Kanji, Hiragana, Katakana, (lateinisches) Alphabet, Ziffer, Symbol usw.

- *Wortarten* (engl. *part-of-speech)*
  Wortarten wie Nomen, Verb, Zahlwort, aber auch Komma, Nomen-Suffix[141] und an-

[136] An der MUC-7 wurde auch ein Extra-Task für asiatische Sprachen durchgeführt. Dieser wurde *Multilingual Entity Task* genannt und fand bereits zum zweiten Mal statt (*MET-2*).

[137] Vergleiche Unterkapitel 5.2.3, Abschnitt *b) Trainingsphase*, *1. Klassen und Kontext* (Seite 74).

[138] Vgl. Unterkapitel 5.2.3, Abschnitt *c) Anwendung auf neue Texte.*

[139] Die Terminologie der Klassenbezeichnung ist hier von MENE übernommen.

[140] Wie eine solche Klassifizierung vor sich geht und warum sie sinnvoll ist, wird in Abschnitt *3. Ein Beispiel* anschaulich dargestellt.

[141] In [Sek98a] wird nicht erläutert, was ein Nomen-Suffix (dargestellt als `N-suf`) ist. Es scheint eine spezielle Wortart fürs Japanische zu sein.

Token	ISURAERU	KEISATSU	NI	YORU	TO	,	ERUSAREMU
Wortart	PN-loc	N	postpos	V	postpos	comma	PN-loc
Schrift.typ	Kata	Kanji	Hira	Hira	Hira	Komma	Kata
Spez. Wörtb.	loc	org-S	-	-	-	-	loc
NE-Klasse	org_start	org_end	-	-	-	-	loc_start

Token	SHI	HOKUBU	DE	26	NICHI	GOGO	,
Wortart	N-suf	N	postpos	number	N-suf	N	comma
Schrift.typ	Kanji	Kanji	Hira	Num	Kanji	Kanji	Komma
Spez. Wörtb.	loc-S	-	-	-	date-S	time(-P)	-
NE-Klasse	loc_start	-	-	date_start	date_end	time_uniq.	-

Abbildung 5.8: Japanischer Beispiel-Satz mit drei Attribut-Arten *Schriftzeichen-Typ*, *Wortart* und *Spezielle Wörterbücher* und der *NE-Klasse* in der jeweils letzten Zeile.

dere Wortarten, die Kennern von ausschliesslich lateinischen und germanischen Sprachen nicht bekannt sind.

- *Spezielle Wörterbücher* (engl. *special dictionaries*)
  Mit Wörterbüchern sind diverse Listen mit Eigennamen und anderen NEs[142] gemeint.

Der Entscheidungs-Baum ist so aufgebaut, dass von jedem Knoten immer nur zwei Kanten nach unten führen. In einem Knoten steht also beispielsweise nicht einfach *Wortart* und dann führen davon eine Anzahl von Kanten weg, die der Anzahl der Wortarten entsprechen. Stattdessen wird im Knoten ein konkretes Attribut abgefragt, also zum Beispiel ob das Token ein Nomen sei. Die wegführenden Kanten eines Knotens lauten somit immer *ja* oder *nein*, unabhängig davon ob im Knoten auf eine bestimmte Wortart, auf einen Schriftzeichen-Typen oder ein Wörterbuch getestet wird. Zudem gibt es auch Knoten, die die Attribute für Tokens vor respektive nach dem zu bestimmenden (aktuellen) Token prüfen.

Neben der bereits erwähnten Klassifizierung als Wahrscheinlichkeitsverteilung in den Blättern zeigt das Beispiel im folgenden Unterkapitel auch, wie Attribute in den Knoten und die Werte in den Kanten konkret aussehen können.

3. Ein Beispiel

Das folgende Beispiel ist aus [Sek98a] übernommen.[143] Abbildung 5.8 zeigt einen japani-

[142] Da die Vorgaben für MET-2 gleich lauteten wie diejenigen für den NE-Task an der MUC-7 - nämlich dass als NEs neben Eigennamen auch Zeit-, Währungs- und Prozentangaben zu erkennen seien - werden auch Wörterbücher mit Zeit-, Währungs- und Prozentangaben verwendet.

[143] Die Terminologie entspricht nur teilweise dem Original-Beispiel in [Sek98a]. Vor allem im Bereich der Klassen-Bezeichnung wurden andere Begriffe gewählt, nämlich diejenigen, die bereits von MENE her

```
ISURAERU (erstes Token)
falls aktuelles Token:  location     -> ja
falls nächstes Token:   loc-suffix   -> nein
falls nächstes Token:   pers-suffix  -> nein
falls nächstes Token:   org-suffix   -> ja
falls vorangeg. Token:  location     -> nein
DANN other = 0.67, org_start = 0.33
```

```
KEISATSU (zweites Token)
falls aktuelles Token:  location     -> nein
falls aktuelles Token:  organization -> nein
falls aktuelles Token:  time         -> nein
falls aktuelles Token:  loc-suffix   -> nein
falls nächstes Token:   time-suffix  -> nein
falls aktuelles Token:  time-suffix  -> nein
falls nächstes Token:   date-suffix  -> nein
falls aktuelles Token:  date-suffix  -> nein
falls aktuelles Token:  date         -> nein
falls nächstes Token:   location     -> nein
falls aktuelles Token:  org-suffix   -> ja
falls vorangeg. Token:  location     -> ja
DANN other = 0.14, org_end = 0.86
```

Abbildung 5.9: Beispiel für Entscheidungs-Baum-Pfade (Japanisch)

schen Beispielsatz mit den drei Attribut-Arten *Schriftzeichen-Typ*, *Wortart* und *Spezielle Wörterbücher* und mit der *Klassifizierung* in der jeweils letzten Zeile (Zeile *NE-Klasse*). Abbildung 5.9 zeigt zwei Beispielpfade für die ersten beiden Tokens aus dem Beispielsatz (`ISUAERU` = *Israel* und `KEISATSU` = *Polizei*). Jede Zeile im Pfad entspricht einem Attribut und seinem Wert (*ja* oder *nein*). Die jeweils letzte Zeile eines Pfades gibt alle Wahrscheinlichkeiten an, mit der das Token einer der 29 Klassen entspricht, sofern diese Wahrscheinlichkeiten jeweils grösser als Null sind. Hier kann nun abgelesen werden, was der Viterbi-Algorithmus bewirkt und wozu dies dient. Die Wahrscheinlichkeit `other` (das heisst *keine NE*) fürs Token `ISUAERU` ist mit 0.67 höher als diejenige, mit der das Token `org_start` getaggt wird (0.33). Im folgenden Token `KEISATSU` hingegen ist die Wahrscheinlichkeit, dass es sich um ein `org_end` handelt, sehr viel grösser, als dass es keine NE (`other`) ist (0.84 zu 0.14). Der Viterbi-Algorithmus verrechnet nun die jeweils zusammengehörigen Werte miteinander durch Multiplikation. Dies ergibt für die Klassen mit `org` einen Wert von 0.29, für `other` 0.09. Somit wird der Ausdruck `ISURAERU KEISATSU` richtigerweise als Organisationsname getaggt. Wären einfach die jeweils grössten Wahrscheinlichkeiten genommen worden, so hätte dies zu einer widersprüchlichen und somit unmöglichen Klassifikation geführt.

4. Vergleich mit verwandten Systemen

Wie bereits erwähnt gibt es einige NER-Systeme, die mit Entscheidungs-Bäumen arbeiten, zum Beispiel [Ben97] oder [Gal96]. Im Folgenden soll das System *RoboTag* von Scott W.

bekannt sind. Die Entsprechungen lauten: `x-OP-CL` in [Sek98a] für `x_unique` in MENE, `x-OP-CN` für `x_start`, `x-CN-CN` für `x_continue`, `x-CN-CL` für `x_end` und `none` für `other`, wobei `x` für `organization, location` usw. steht.

Bennett et al.[144] mit dem soeben beschriebenen verglichen werden.

Als Attribut-Arten verwendet RoboTag sehr ähnliche wie Sekines System: Wortart, Token-Typ[145] und Listen mit Eigennamen. Auch in RoboTag gehen von einem Knoten jeweils nur zwei Kanten nach unten weg, eine Ja- und eine Nein-Kante. Sowohl die Attribute als auch die Baumarchitektur sehen denjenigen von Sekine also sehr ähnlich.

Der Hauptunterschied zu Sekines NER-System ist jedoch, dass RoboTag mit mehr als einem Baum arbeitet, nämlich mit einem *Start-Token-Baum* und einem *End-Token-Baum* pro Klasse. Ein Start-Token-Baum dient dazu, diejenigen Tokens als `x_start` zu markieren, die den Beginn einer NE darstellen, ein End-Token-Baum zur Markierung der entsprechenden `x_end`-Tokens.

Diese Eigenheit von RoboTag hat gravierende Auswirkungen. Mehrere Bäume bedeuten auch mehrere Klassifikationen pro Token, nämlich pro Baum eine. So kann es vorkommen, dass ein Token von verschiedenen Bäumen verschieden klassifiziert wird. Diese Inkonsistenzen müssen verhindert werden. Die Methoden, die in RoboTag dazu angewendet werden, erscheinen kompliziert und nicht unbedingt einsichtig. Gerne schliesst man sich dem Urteil von Sekine an:

> They [Bennett et al.] solved the problem by introducing two somewhat idiosyncratic methods. One of them is the distance score, which is used to find an opening and closing pair for each named entity mainly based on distance information. The other is the tag priority scheme, which chooses a named entity among different types of overlapping candidates based on the priority order of named entities. These methods require parameters which must be adjusted when they are applied to a new domain. In contrast, our system does not require such methods, as the multiple possibilities are resolved by the probabilistic method. This is a strong advantage, because we don't need manual adjustments.[146]

Was auf den ersten Blick als Vorteil erscheint, kann sich bei näherem Hinschauen aber auch wieder als Nachteil herausstellen: Wenn bei einem Baum etwa 50 Attribute[147] für drei verschiedene Tokens[148] getestet werden können, so kann dieser Baum sehr gross werden. Dies kann zur *Überbestimmung* (engl. *overfitting*)[149] führen, so dass unter Umständen damit keine Klassifikation mehr durchgeführt werden kann. Eine mögliche Lösung ist es dann,

[144] Vgl. [Ben97].

[145] Token-Typ entspricht in etwa dem Schriftzeichen-Typ von Sekines System. Da RoboTag sowohl fürs Englische als auch fürs Japanische entwickelt wurde, finden sich unter diesen Attribut-Arten auch Attribute wie `hiragana`, `kanji` usw.

[146] [Sek98a].

[147] In [Sek98a] werden keine genaue Angaben über die Anzahl Attribute gemacht. Der Schätzwert von 50 beruht auf Beobachtung der Beispiele und Paraphrasierungen über die Attribut-Arten in [Sek98a]. Es kann gut sein, dass der richtige Wert doppelt so gross ist, vor allem weil über die Anzahl Wortarten-Attribute mit dem Satz "We define the set of our categories based on its major category and minor category" eine vage Beschreibung vorliegt. Da mit der Schätzung über die Anzahl an Attributen aber nur gezeigt werden soll, wie gross der so erstellte Entscheidungs-Baum werden kann, genügt auch eine ungefähre Angabe.

[148] Das aktuell zu bestimmende Token, das davor und das danach stehende.

[149] Vgl. dazu [Mit97], Kapitel *3.7.1 Avoiding Overfitting the Data*.

den Baum unten abzuschneiden (engl. *pruning*). Ob Sekines Entscheidungs-Baum-System solchen Problemen unterliegt, ist aus [Sek98a] nicht zu entnehmen.

Kapitel 6

Vergleich, Beurteilung und Fazit

In diesem Kapitel werden die in Kapitel 5 vorgestellten Methoden miteinander verglichen, Gemeinsamkeiten gesucht, Vor- und Nachteile aufgeführt, Leistungen beurteilt. Zuerst soll den ersten beiden in Kapitel 1 formulierten Fragen nachgegangen werden: Welches sind in der Computerlinguistik die heute üblichen Ansätze oder Methoden, um in Texten Eigennamen automatisch zu erkennen? Welche Methoden eignen sich am besten?

Dazu werden in 6.1 zunächst die Methoden regelbasierter Systeme miteinander verglichen, daraus die wichtigsten ermittelt, um danach zu beurteilen, wie gut sich regelbasierte Systeme zur Eigennamen-Erkennung eignen. Unterkapitel 6.2 ist etwas anders aufgebaut: Die statistischen Systeme werden zuerst alle einzeln beurteilt, dann miteinander verglichen, um hernach wieder ein Urteil abzugeben, wie gut sie sich zur Eigennamen-Erkennung eignen.

In 6.3 wird sodann versucht, die letzte der in Kapitel 1 gestellten Fragen zu beantworten: Wie eignen sich die derzeitigen Methoden zur Erkennung von Personen-, Orts- und Organisationsnamen, um Produktenamen zu erkennen?

Bereits an dieser Stelle kann vorweggenommen werden, dass es keine endgültigen Antworten auf die Eignungs-Fragen geben wird: Sowohl die Frage, welches System am geeignetsten zur Eigennamen-Erkennung sei, als auch diejenige, welches System sich am besten zur Produktenamen-Erkennung adaptieren lässt, kann nur tendenziell, aber längst nicht abgeschlossen, beurteilt werden. Zu einem solchen und weiteren Schlüssen gelangt man in 6.4, wo die Erkenntnisse der vorangegangenen Unterkapitel zusammengefasst und wieder erwogen werden.

6.1 Methoden regelbasierter Systeme im Vergleich

6.1.1 Wichtigste Methoden

McDonalds Kernidee vom Ausnutzen interner und externer Evidenz[1] setzen alle regelbasierten Systeme um. In all diesen Systemen wird mit *Regeln* gearbeitet, welche *Indikatoren*[2] enthalten, die entweder selbst Bestandteile eines Eigennamens sind (interne Evidenz: zum Beispiel *Inc.* bei einem Organisationsnamen) oder die in der Nähe eines solchen stehen (externe Evidenz: zum Beispiel *Mr.* vor Personennamen). Mit wenigen Ausnahmen (PNF und LT TTT) werden *vorgefertigte Eigennamenlisten* eingesetzt.[3] Einige arbeiten auch mit *Koreferenz-Regeln*, um Namen zu erkennen, die an anderer Stelle bereits erkannt wurden, jetzt aber in anderer Ausprägung auftreten - das eine Mal voll ausgeschrieben, das andere Mal entweder als Abkürzung oder als Teilstück des vollen Namens.[4] Allen regelbasierten Systemen gemeinsam ist, dass sie sich als erstes auf Grossschreibung abstützen[5]: Als NE-Kandidaten gelten (alleinstehende oder Sequenzen von) grossgeschriebenen Tokens, allenfalls können dabei auch Ziffern vorkommen, manchmal auch (kleingeschriebene) Präpositionen in Kombination mit bestimmten Indikatoren (zum Beispiel in *Bank of Canada* ist *Bank* dieser bestimmte Indikator).

Dies sind die wichtigsten Methoden zur Eigennamen-Erkennung bei regelbasierten Systemen, und alle werden von - nahezu - allen regelbasierten Systemen eingesetzt. Es sind also Methoden, die sich weitgehend eignen, NEs zu erkennen.

6.1.2 Vereinzelt angewandte Methoden

Um die Leistung der Systeme noch zu erhöhen, arbeiten einige regelbasierte Systeme mit Methoden, die in anderen nicht vorkommen. Solche weniger häufig anzutreffende Methoden werden im Folgenden angeführt.

Zwei regelbasierte Systeme setzen *Filter* ein (PNF und Oki), um Markierungen von Tokens, die keine Eigennamen sind, jedoch fälschlicherweise als solche erkannt wurden,

[1] Vgl. Seite 26.

[2] Die Indikatoren werden je nach System verschieden benannt. Im BSEE von FACILE / CONCERTO zum Beispiel werden sie *Clue Words* (vgl. Seite 29) genannt, in Oki heissen sie *funktionale Wörter* (vgl. Seite 49) und in LOLITA *Designatoren* (vgl. Seite 57). Trotz unterschiedlicher Bezeichnungen lässt sich erkennen, dass diese Wörter, die hier unter der Bezeichnung *Indikatoren* zusammengefasst werden, dieselbe - oben beschriebene - Funktion ausüben.

[3] "Vorgefertigt" besagt, dass die Listen bereits *vor* dem Anwenden auf einen zu taggenden Text vorhanden sind. In LT TTT wird zwar auch mit Namenslisten gearbeitet, diese werden aber dynamisch während des Anwendens auf einen Text erstellt und auch nur für diesen einen Text verwendet.

[4] Einige der Systeme sind explizit für beide Fälle ausgelegt - also fürs Erkennen von Teilnamen und Abkürzungen (vgl. zum Beispiel Oki), andere nur für Abkürzungen (zum Beispiel LOLITA) und dritte nur für Teilnamen (zum Beispiel LT TTT). Aus den Berichten geht nicht immer klar hervor, welche Art von koreferierenden Namen mit den Regeln erkannt werden, was aber auch nicht problematisch ist. Von Interesse ist vor allem, dass mit Koreferenz-Regeln gearbeitet wird.

[5] Oft wird dieses Abstützen auf Grossschreibung aber nicht explizit erwähnt.

wieder zu entfernen. Wie sich Filterung auswirkt, wie gut sie ist, wird weder bei PNF noch bei Oki beschrieben - weshalb sie hier nicht näher beurteilt werden kann.

Mit *Gewichtung* von Regeln versuchen zwei regelbasierte Systeme (NetOwl und BSEE von FACILE / CONCERTO) bei konkurierenden Regeln die jeweils richtige zum Einsatz kommen zu lassen. Wie wirksam Regel-Gewichtung ist, ist schwierig zu beurteilen. Genaue Ausführungen zur Funktionsweise (wie die Gewichte gefunden werden, wie sie gegeneinander abgewogen werden) und zur Effektivität fehlen in beiden Systembeschreibungen. Die Beobachtungen, dass das BSEE-System von nach anfänglichem Einsatz von Gewichtung wieder davon abgesehen hat, NetOwl dagegen im MUC-7-Wettbewerb den zweitbesten Rang erreicht hat, lassen auch keine schlüssigen Aussagen zu. Allerdings kann vermutet werden, dass auch bei NetOwl mittels Gewichtung kein grosser Erfolg erzielt werden konnte, ansonsten dies im Beschrieb sicherlich erwähnt worden wäre.

Wiederum nur zwei regelbasierte Systeme führen vor der NER *Syntax-Analyse* durch (Oki und LOLITA). Bei Oki scheint dieses Parsen allerdings nicht sehr fruchtbringend zu sein.[6] Bei LOLITA ist der Parser einem Semantik-Modul vorgeschaltet, das für die Eigennamen-Erkennung zur Hauptsache verantwortlich ist. Da der Parser von MUC-6-LOLITA nur ungenügende Leistung erbrachte, verarbeitete das Semantik-Modul fehler- und lückenhaften Input, was sogar zur Verschlechterung der NER führte. Erst umfangreiche Änderungen (Vergrösserung und Verbesserung der Grammatik, Back-up-Strategien[7]) im MUC-7-LOLITA erlaubten einen Einsatz von Parsing, der nicht kontraproduktiv war. Ob das Parsing überhaupt - auch nach den verbessernden Änderungen - eine sinnvolle Methode zur NER ist, scheint zweifelhaft, zumal die Leistungen von LOLITA im NE-Task nicht überzeugen (zweitletzter Rang)[8].

Auch anderweitig zeichnet sich LOLITA durch umfangreiche Abweichung von den anderen regelbasierten Systemen aus. Auffällig ist vor allem die *Wissensbasis*, die einzig bei LOLITA als *semantisches Netz* organisiert ist. Viele Analyse-Module (wie zum Beispiel die oben erwähnten Syntax- und Semantik-Analyse-Module) reichern die Wissensbasis mit Informationen an. Eigennamen-Erkennung wird dann als Applikation auf das semantische Netz aufgesetzt. Doch LOLITA kann mehr als nur NEs zu erkennen, zahlreiche andere Applikationen[9] können aufgesetzt werden. Diese Multifunktionalität bewirkt aber Einbussen bei der Leistung der einzelnen Applikationen. Die Konzipierung als multifunktionales System[10] und wohl auch die Organisation der Wissensbasis als semantisches Netz scheinen demnach keine probaten Mittel für effektive Eigennamen-Erkennung zu sein.

Dagegen scheint sich die *Kombination von Regeln und statistischen Methoden* zu bewähren. LT TTT, das System, das im MUC-7-Wettbewerb den ersten Rang erlangt hat, wendet

[6] Vgl. Unterkapitel *b) Novum Parsen: Eine genauere Betrachtung*, Seite 52.

[7] Vgl. Unterkapitel *b) NER-Komponenten*, Seite 55.

[8] Vgl. *Rangliste NE-Task Englisch* im *Anhang B.*

[9] Zum Beispiel weitere MUC-Tasks wie TE und ST, aber auch natürlichsprachliche Abfrage, Maschinelle Übersetzung, Informations-Extraktion, Tutoring beim Fremdsprachenlernen.

[10] Mit "Konzipierung als multifunktionales System" soll auch auf die durch umfangreiche Analyse erstellte, informationsreiche, als semantisches Netz organisierte Wissensbasis angesprochen werden, die sodann für unterschiedlichste Anwendungen benutzt wird.

an mehreren Stellen ein ME-Modell an, an einer auch ein HMM. Auch andere regelbasierte Systeme setzen zusätzliche statistische Methoden ein: McDonald (PNF) spricht von statistischen Heuristiken, und in LaSIE wird ein Brill-Tagger[11] eingesetzt. Da, wie bereits in 5.1.1[12] erläutert, McDonald nicht ausführt, wie diese statistische Heuristiken aussehen und da sich sonstiges statistisches Verfahren bei anderen regelbasierten Systemen immer auf den Tagger beschränkt (der lediglich einen Vorverarbeitungsschritt ausführt), lohnt eine nähere Untersuchung statistischer Methoden in diesen Systemen nicht. Lediglich die Anwendung statistischer Methoden, wie es in LT TTT praktiziert wird, gibt einen starken Hinweis darauf, dass die Kombination statistischer und regelbasierter Verfahrensweisen erfolgversprechend sein könnte, und es sich lohnt, diese im Auge zu behalten.[13]

Zum Schluss dieses Unterkapitels noch drei Methoden, die jede nur einmal, und zwar je in einem anderen regelbasierten System, angewandt wurden.

- In PNF wird neben einem *semantischen Modell* auch ein *Diskursmodell* erstellt. Diese Zweiteilung soll zur Disambiguierung von Namen (im semantischen Modell als "Namensobjekte" bezeichnet) beitragen, die mehrere Entitäten (im Diskursmodell mit "konkrete Individuen" benannt) bezeichnen können. Die Macher von PNF vertreten den Standpunkt, Eigennamen-Erkennung für sich sei nutzlos, denn sie müsse immer den Zweck haben, zum gründlichen Verstehen eines Textes beizutragen. Diesem Textverstehen versuchen sie mit dem Aufbau eines Diskursmodells, welches die Welt abbilden soll, näher zu kommen.

  Obwohl [McD93] eine wegweisende Arbeit ist, wurde die Idee von Diskurs- und semantischem Modell von anderen Forschergruppen nicht übernommen.[14] Wahrscheinlich ist einerseits der Anspruch des Verstehens von Text sehr hoch - in [McD93] wird von "thorough understandig of extended unrestricted texts" gesprochen - und andererseits kann Disambiguierung auch anderweitig durchgeführt werden (vergleiche beispielsweise die folgenden beiden Methoden).

- In LaSIE werden bestimmte Beziehungen von nicht klassifizierten Eigennamen zu Wörtern mit bestimmter semantischer Information ausgenutzt, um Rückschlüsse auf die Klasse der Namen zu ziehen (*inferieren*). Ob sich diese Methode bewährt, scheint jedoch zweifelhaft, zumal sie in einer späteren Version von LaSIE (LaSIE 2.1) nicht mehr angewandt wird.

- In MUC-7-LOLITA wird Disambiguierung ausgeführt, indem das *Kategorisieren verzögert* durchgeführt wird. Das heisst, dass ein noch nicht kategorisierter Name erst dann getaggt wird, wenn ein klarer Hinweis[15] auftaucht, um welche Art von Eigennamen es sich handelt. Wieder kann aber keine Aussage über die Effizienz dieser

[11] Der Brill-Tagger ist ein sehr bekannter und häufig verwendeter Tagger, der regelbasiertes Verfahren mit statistischem kombiniert.

[12] Vgl. Seite 27.

[13] In Unterkapitel 6.4 wird auf kombinierte - hybride - Ansätze zurückgekommen.

[14] Wegweisend in [McD93] ist vielmehr die Unterscheidung zwischen *interner* und *externer Evidenz*.

[15] So ein Hinweis kann beispielsweise ein Personalpronomen sein (vgl. Beispiel (9), Seite 57).

Disambiguierungs-Methode gemacht werden, da sich in [Gar98] keine Angaben dazu finden.

6.1.3 Fazit für regelbasierte Systeme

Für regelbasierte Systeme können also als wichtigste Methoden zur Eigennamen-Erkennung die folgenden festgehalten werden: Regeln, Indikatoren, (vorgefertigte) Eigennamenlisten und Ausnutzen von Koreferenz und Grossschreibung.

Die Leistungen dieser Methoden sind allerdings stark von *Anzahl* und *Qualität* der Regeln, Indikatoren etc. abhängig. Zu diesem Schluss kommt man, wenn man die beiden regelbasierten Systeme BSEE von FACILE / CONCERTO und NetOwl miteinander vergleicht, die von aussen betrachtet sehr ähnlich arbeiten. Beide beruhen auf Anwenden von (gewichteten) Regeln, Beiziehen von Namenslisten und Indikatoren, nutzen Koreferenz und Grossschreibung aus. Trotzdem sind die Unterschiede in der Leistung der beiden Systeme enorm: NetOwl erreicht in MUC-7 einen F-Wert von 91.6 und somit gesamthaft den zweiten Rang, unter den nur regelbasierten[16] sogar den ersten Rang. BSEE hingegen erreicht mit dem F-Wert von 81.91 gesamthaft nur den elften Platz, was unter den regelbasierten Systemen der letzte Rang ist. Diese Inkonsistenz zwischen gleichen Methoden, aber unterschiedlichen Leistungen ist vor allem darauf zurückzuführen, dass in NetOwl viel mehr Arbeitszeit investiert wurde als in BSEE.[17] Mit viel Fleiss, Geschick, Zeitaufwand und - da diese Arbeit von gut ausgebildeten ComputerlinguistInnen ausgeführt werden muss - umfangreichen finanziellen Mitteln kann also durchaus ein gutes regelbasiertes Eigennamen-Erkennungs-System gebaut werden kann, bei Fehlen dieser entscheidenden Faktoren werden die Leistungen eines solchen Systems jedoch nicht zufriedenstellend sein. Anders gesagt: Wie leistungsfähig ein regelbasiertes System arbeitet, ist von den Ressourcen abhängig. Grosse Ressourcen an Zeit und Geld bedeuten grosse Ressourcen an guten Indikatoren[18] und Regeln - die Basis für grosse Leistungsfähigkeit.

Nicht zu vergessen ist der Nachteil regelbasierter Systeme, dass das Portieren in eine andere Domäne oder Sprache wiederum grosser Arbeit bedarf, was erstens neue Kosten verursacht und zweitens die bereits ausgerichtete Arbeit grösstenteils unnütz werden lässt.[19]

[16] Den ersten Rang in der Gesamtwertung erreicht LT TTT, das jetzt nicht zu den rein regelbasierten gezählt wird, weil es, wie schon öfters erläutert, auch mit vielen statistischen Methoden arbeitet.

[17] Zum Zeitpunkt der MUC-7 war in das bereits an 30 Kunden verkaufte NetOwl schon mehr als drei Jahre Arbeit investiert worden (vgl. die Einleitung von 5.1.5), wohingegen beim BSEE-System für Entwicklung und Testen der Ressourcen lediglich ein Personenmonat eingesetzt worden war (vgl. die Angaben in [Bla98] unter *Analysis and Conclusions*).

[18] Der Begriff *Indikatoren* wird hier und im Folgenden im gleichen Sinne verwendet, wie er bereits in Unterkapitel 6.1 verwendet wurde.

[19] Dass die Leistung regelbasierter Systeme stark von Zeit und Geld abhängig ist und die Systeme schlecht portierbar sind, dieser Meinung ist auch Borthwick (vgl.[Bor99], Seite 8f).

6.2 Statistisch basierte Systeme im Vergleich

Anders als bei regelbasierten Systemen können statistisch basierte schlecht in einzelne Module (die in vorliegender Arbeit als *Methoden* bezeichnet werden) aufgeteilt werden. Vielmehr ist das ganze System eine einzige Methode, manchmal zusätzlich kombiniert mit weiteren Systemen. Auch ist es schwierig, Gemeinsamkeiten bei verschiedenen statistisch basierten Systemen zu finden. Das wenige, was an Gemeinsamkeiten angeführt werden kann, hat mehr mit dem allgemeinen Charakter von Statistik zu tun als spezifisch mit statistischen Systemen zur Eigennamen-Erkennung:

- Immer braucht es ein Trainingskorpus, das manuell oder halbautomatisch getaggt werden muss.[20]
- Zudem gilt für alle statistisch basierten Systeme das Problem, dass immer (zu) wenig Trainingsdaten vorhanden sind (*Sparse-Data-Problem*). Das heisst, dass im zu taggenden Korpus Fälle auftreten, die im Trainingskorpus nicht vorkommen.[21] Somit fehlen an einigen Stellen Parameter, die zur Berechnung von NE-Klassen nötig sind.

Da also unter statistisch basierten Eigennamen-Erkennungs-Systemen keine *spezifischen* Gemeinsamkeiten bestehen, soll an dieser Stelle das Augenmerk auf die Vor- und Nachteile der einzelnen statistisch basierten Systeme gerichtet werden.

6.2.1 PIE-System mit Erweiterung durch Kollokations-Statistik

Ein Sonderfall unter den statistisch basierten Systemen ist das in Kapitel 5.2.1 vorgestellte *PIE-System mit Erweiterung der Eigennamen-Erkennung durch Kollokations-Statistik*: Lediglich die Erweiterung der Eigennamen-Erkennung beruht auf statistischen Techniken, nämlich auf Klassifikation mittels *Naivem Bayes-Klassifikator* und *automatischem Extrahieren weiterer Regeln*. Im Folgenden soll beurteilt werden, wie lohnend diese beiden Techniken sind.

- Beim Betrachten der Leistungssteigerung, die durch den Einsatz des Naiven Bayes-Klassifikators erreicht wird, kommt man zum Schluss, dass sich diese Erweiterung des regelbasierten PIE-Systems lohnt: Ohne Naiven Bayes-Klassifikator erreicht PIE einen F-Wert von 83.70, mit Naivem Bayes-Klassifikator liegt der F-Wert bei 86.37.[22]

[20] Es gibt manchmal Möglichkeiten, auch ohne getaggtes Trainingskorpus auszukommen. Ein HMM beispielsweise kann auch mit einem Wörterbuch und dem *Forward-Backward-Algorithmus* statt mit einem Trainingskorpus erstellt werden (vgl. Seite 66).

[21] Gut vorstellbar ist beispielsweise der Fall bei einem HMM: Ein Wort, das im Trainingskorpus nicht vorkommt, jedoch im zu taggenden Text, hat keine Ausgabe-Wahrscheinlichkeit. Diese ist aber nötig zur Berechnung einer Tagfolge und somit der einzelnen Tags. Meist gibt es aber Lösungen oder zumindest Versuche, dem Sparse-Data-Problem zu begegnen. Beim HMM lauten die Lösungsansätze *Back-off-Modelle* und *Smoothing* (vgl. Seite 72).

[22] Am MUC-7-Wettbewerb nahm man mit beiden Versionen teil und erreichte so einmal den 9. und einmal den 5. Rang. Vgl. *Rangliste NE-Task Englisch* im *Anhang B*.

- Als Vorstufe eines Naiven Bayes-Klassifikators kann das automatische Extrahieren weiterer Regeln[23] gesehen werden. Klassifikation durch einen Naiven Bayes-Klassifikator wird erst dann angewandt, wenn ein Eigenname, der in einer bestimmten Relation zu einem bestimmten Hauptwort steht, im Trainingstext nicht weitgehend einheitlich klassifiziert ist. Solange ein (willkürlich festgelegter) Schwellwert nicht überschritten wird, reicht eine Klassifikation mittels automatisch extrahierter Regeln, was einen geringeren Rechenaufwand bedeutet. In Bezug auf die Qualität dieser Technik fällt auf, dass so einige Regeln gefunden werden, die bereits vorher von Hand ermittelt wurden. Diese Redundanz gibt einen Hinweis darauf, dass zumindest ein Teil der so ermittelten Regeln korrekt ist. Da die Regeln sehr vorsichtig erstellt wurden - bei Unsicherheiten wurde auf die Technik des Naiven Bayes-Klassifikators ausgewichen - kann davon ausgegangen werden, dass die Qualität der so ermittelten Regeln gut ist. Die Quantität ist jedoch nicht sehr hoch: Für 22 Millionen Wörter wurden mittels dieser Technik insgesamt - also inklusive der bereits von Hand erstellten - 3623 Regeln gefunden. Inwieweit mittels dem automatischen Regeln-Extrahieren die Leistung des Systems beeinflusst wurde, kann wegen fehlender Angaben hier nicht beurteilt werden. Dennoch lassen sich Überlegungen anstellen zur Frage, wie lohnend diese Methode ist. Bevor Regeln extrahiert werden können, muss eine Kollokations-Datenbasis erstellt werden. Da diese sowieso für den Naiven Bayes-Klassifikator angelegt werden muss, bedeutet dies kein Mehraufwand. Dagegen ist der Rechenaufwand geringer als beim Naiven Bayes-Klassifikator. Insofern scheint diese Methode sinnvoll. Einzig der willkürlich festgelegte Schwellwert scheint problematisch. Zwar erhält man beim Studieren von [Lin98] den Eindruck, dass diese Technik wirklich nur dann angewandt wird, wenn die so ermittelten Regeln als zuverlässig betrachtet werden können. Trotzdem bleibt der Schönheitsfehler der Willkür bestehen und die berechtigte Frage stellt sich, ob nicht gleich in *allen* Fällen der uneinheitlichen Klassifizierung im Trainingskorpus die Technik des Naiven Bayes-Klassifikators angewandt werden sollte. Der dadurch entstehende grössere Rechenaufwand ist dabei irrelevant, zumal es ja immer um offline-Berechnungen geht.

6.2.2 IdentiFinder

Das erste in dieser Arbeit beschriebene System, das vollständig statistisch arbeitet, ist *IdentiFinder*. Der Verzicht auf Regeln bewirkt, dass das System weitgehend[24] sprachunabhängig ist, das heisst leicht portierbar auf andere Sprachen, sofern ein genügend grosses, getaggtes Trainingskorpus vorliegt. Nebst dieser Sprachenunabhängigkeit überzeugt bei

[23] In [Lin98] werden die automatisch extrahierten Regeln *Erkennungsregeln* (engl. *recognition rules*) genannt. Eine exaktere Bezeichnung wäre *Klassifikations-Regeln*, weil es nur noch um das Klassifizieren von Sequenzen grossgeschriebener Wörter geht, von denen bereits angenommen wird, dass sie Eigennamen sind.

[24] Das System ist vor allem deshalb nicht vollständig sprachunabhängig, weil die Merkmale f (Features) sprachspezifisch sind. Wie die Autoren von [Bik99] auf S. 7 aber schreiben, macht die Berechnung der Merkmale einen sehr kleinen Teil der Implementierung aus (ca. 20 Zeilen Programm-Code).

IdentiFinder natürlich der F-Wert von 90.44, mit dem am MUC-7-Wettbewerb der 3. Rang erreicht wurde. Trotz dieser guten Leistung gibt es in IdentiFinder auch ein paar kritische Punkte.

Die Architektur der ineinander verschachtelten HMMs[25] ermöglicht, dass Tokens zu Gruppen zusammengefasst werden können, was dann nötig ist, wenn Eigennamen aus mehreren Tokens bestehen. Die dazu benötigten Formeln zur Berechnung einer Namensklasse (Formel (5.10) und (5.12)) respektive eines Tokens innerhalb einer Namensklasse (Formel (5.15)) berücksichtigen Faktoren, die nicht immer unproblematisch sind.

- In allen drei Formeln geht es um bedingte Wahrscheinlichkeiten mit je *zwei* Bedingungen. Zwei Bedingungen bedeuten eine grosse Einschränkung der Wahrscheinlichkeit. In den Formeln (5.12) und (5.15) spezifizieren zusätzliche Merkmale die Wahrscheinlichkeiten. Das bereits erwähnte Sparse-Data-Problem, unter dem statistische Techniken immer leiden, wird durch derart spezifische Wahrscheinlichkeiten noch verschärft, was sich in den bereits erwähnten Problemen *Unbekannte Wörter und Bigramme*[26] niederschlägt. Die zur Lösung angewandte Back-off-Methode kritisiert Borthwick als zu häufig eingesetzt und gefährlich.[27]

- Während die Wahrscheinlichkeits-Formeln auf der einen Seite einen zu spezifischen Kontext berücksichtigen, kann es auf der anderen Seite aber auch vorkommen, dass der berücksichtigte Kontext zu klein ist. Wenn zum Beispiel ein Eigenname, der unbekannte Tokens enthält, links und rechts von Kommas - die auch als Tokens gelten - abgeschlossen ist, reichen die Bigramme, die in den Wahrscheinlichkeits-Formeln betrachtet werden, nicht mehr aus, um zuverlässig zu klassifizieren. Ein Komma als Teil eines Bigramms verfügt über so wenig Information, dass das Bigramm fast wertlos wird. Die generelle Ausweitung des Kontextes auf Trigramme könnte hier Abhilfe schaffen, verschärft aber zugleich wieder das Sparse-Data-Problem. In [Bik99] wird ein Lösungsansatz beschrieben, der den Kontext immer nur dann auf Trigramme ausweitet, wenn sich der Bigramm-Kontext als problematisch erweist, wie zum Beispiel im geschilderten Fall mit den Kommas. Ob sich bei diesem Lösungsansatz der gewünschte Erfolg einstellt, ist in [Bik99] nicht explizit ausgedrückt.

[25] In Kapitel 5.2.2 ist einerseits von einer *Makrostruktur* die Rede, dem übergeordneten Modell, das die nächste Namensklasse berechnet. In der *Mikrostruktur* andererseits existiert pro Namensklasse ein Modell, das innerhalb einer Namensklasse das jeweilige nächste Token berechnet.

[26] Vgl. Seite 72.

[27] Vgl. [Bor99] Seite 15. Borthwick argumentiert, die Back-off-Methode fordere ihren Preis: Zum einen werde das System dadurch komplexer, zum anderen habe die Back-off-Methode mehr Einfluss auf das Resultat als IdentiFinder selbst. Allerdings formuliert Borthwick seine Warnung vorsichtig: "All of this backing off exacts a certain price. Firstly, there is a question of the complexity of the model as more layers of backing off are introduced. Secondly, the issue of how the different layers of backoff are weighted against each other becomes crucial. While BBN's is a reasonable approach, the choice of method will have a major impact on the outcome." Schwerwiegender beurteilt Borthwick, dass ein HMM nur mit einer kleinen Anzahl von Merkmalen (Features) arbeiten könne, weil sonst die Back-off-Methode zu häufig ausgelöst würde. Und da zu einem bestimmten Zeitpunkt immer nur ein Merkmal aktiv sein kann, eigne sich ein HMM nicht, mehrere und unterschiedliche Merkmale auszunutzen - wohingegen diese Fähigkeit die Stärke eines ME-Systems ausmache.

Systeme	**F-Werte Probelauf**	**F-Werte formeller Lauf**
MENE	92.29	84.22
Proteus (Pr)	94.24	86.21
Manitoba (Ma)	93.32	86.37
IsoQuest (IQ)	96.27	91.60
MENE + Proteus	95.61	88.80
MENE + Manitoba	95.49	88.91
MENE + IsoQuest	96.55	91.53
MENE + Pr + Ma	96.48	90.34
MENE + Pr + IQ	96.78	92.05
MENE + Ma + IQ	96.81	91.84
MENE + Pr + Ma + IQ	97.12	92.00

Tabelle 6.1: F-Werte für System-Kombinationen von MENE mit drei anderen Systemen, sowohl für den Probelauf als auch für den formellen Lauf, wie er für die MUC-7 durchgeführt wurde.[30]

Der zu spezifische respektive zu kleine Kontext kann gesamthaft betrachtet aber nicht so kritisch sein, als dass er zu sehr ins Gewicht fallen würde - ansonsten wäre nicht ein so hoher F-Wert erreicht worden. IdentiFinder zeigt, dass ein HMM zur Eigennamen-Erkennung grundsätzlich gut geeignet ist, auch wenn noch Verbesserungen möglich sind.

6.2.3 MENE

Auch das System *MENE*, das mit maximaler Entropie (ME) arbeitet, ist ein rein statistisches System, bietet aber im Gegensatz zum HMM IdentiFinder die Möglichkeit, mit anderen - externen - Systemen kombiniert zu werden. Ob die externen Systeme regel- oder statistisch basiert arbeiten, spielt dabei keine Rolle, da MENE lediglich den Output dieser Systeme benutzt, um daraus *Features für externe Systeme*[28] zu kreieren.

Damit sind die beiden hervorstechenden Eigenschaften von MENE bereits angesprochen: Zum einen ermöglicht das Einsetzen gewichteter *Features*, verschiedenste kontextuelle Evidenz auszunutzen, zum anderen kann MENE mit externen Systemen kombiniert werden, was zur Leistungssteigerung beitragen kann.

Auf den ersten Blick erscheinen die genannten beiden Eigenschaften von MENE als grosse Vorteile. Im Folgenden soll erklärt werden, wie sich jedoch bei genauerem Hinschauen die Leistungsangaben relativieren.

- Vor allem wenn man die Erhöhung der F-Werte betrachtet, die sich dank des Ein-

[28] Vgl. Unterkapitel 5.2.3, Abschnitt *4. Features für externe Systeme*, (Seite 80.)
[30] Die Tabelle ist [Bor99] entnommen (vgl. dort Seite 56, Tabelle 7.2).

satzes von Features für externe Systeme ergibt, möchte man meinen, diese Features eigneten sich ausserordentlich zur Leistungssteigerung von MENE. In bedingten Masse stimmt dies, man betrachte die Zahlen in Tabelle 6.1. Doch bevor man sich von Werten wie 97.12 zum Urteil hinreissen lässt, dank dieser Features würden Resultate erreicht, die denen von menschlichen Annotatoren gleichgestellt werden könnten, gilt es Folgendes zu bedenken: Der äusserst hohe F-Wert von 97.12 wurde im *Probelauf* erreicht, das heisst in Texten, die aus der gleichen Domäne wie die Trainingstexte stammten. Der entsprechende Wert im *formellen Durchlauf* liegt um einiges tiefer bei 92.00.[31] Doch auch dieser Wert ist immer noch beachtlich hoch, an der MUC-7 wäre damit der 2. Rang erreicht worden. Das Problematische an diesem Wert ist jedoch der *enorme Aufwand*, der betrieben werden muss, um einen solchen zu erreichen. Die Situation, in der sich die Entwickler von MENE befanden, als sie die Features für externe Systeme einführten, entspricht nicht der Realität: Für diejenigen Texte, anhand derer die Systeme getestet wurden, stand der Output dreier anderer Systeme zur Verfügung - eine künstliche Situation, die nur dank der MUC herrschte. Nur schon den Output eines einzigen Systems zu erhalten, bedeutet, dass ein solches gebaut werden muss - bei dreien verdreifacht sich der zusätzliche Aufwand.

Aus all diesem kann nun gefolgert werden, dass MENE zwar ermöglicht, äusserst gute F-Werte zu erreichen, dass der Aufwand dafür jedoch sehr hoch ist und eine Leistung, die mit der menschlichen vergleichbar ist, nur dann erreicht werden kann, wenn die Domäne der Testtexte dieselbe ist, wie diejenige der Trainingstexte. Da diese beiden Bedingungen in der Realität schwer zu erfüllen sind, muss der Nutzen des Features für externe Systeme deshalb in Frage gestellt werden.

- Die Hauptattraktivität eines ME-Systems liegt in der Möglichkeit, unterschiedlichste Evidenz ausnutzen zu können. MENE tut dies, indem mit acht verschiedenen Feature-Typen gearbeitet wird.[32] In [Bor99] wird untersucht, wieviel die einzelnen Features zum Resultat beitragen.[33] Aus den verschiedenen F-Werten ist herauszulesen, dass in erster Linie die lexikalischen Features produktiv[34] sind, da ohne sie der F-Wert enorm sinken würde (von 84.22 mit lexikalischen Features auf 58.35 ohne sie).[35] Schon merklich weniger, aber immer noch beachtlich produktiv sind die

[31] Der *formelle Durchlauf* wird hier deshalb als der entscheidende angesehen, weil die Testtexte aus einer anderen Domäne als die Trainingstexte stammen, was realitätsnaher ist, als wenn sie wie beim *Probelauf* aus derselben Domäne stammen. Für eine genaue Beschreibung des Unterschieds zwischen Probelauf (engl. *dry run*) und formellen Durchlauf (engl. *formal run*) vgl. [Bor99], Seite 5.

[32] Vgl. die Angaben in *e) Features: Typen und Gewinnung* auf Seite 79ff.

[33] Vgl. [Bor99], Seite 59, Tabelle 7.5.

[34] Mit dem Ausdruck *produktiv* ist der Grad gemeint, wie stark zur NER beigetragen wird.

[35] Als Bezugsgrösse wurden hier die F-Werte des formellen Durchlaufs gewählt, zum einen weil dieser wie oben schon angesprochen durch die Verschiebung in eine andere Domäne eher der realen Anforderungen an ein Eigennamen-Erkennungs-System entspricht, zum anderen weil es reicht, einen Durchlauf zu betrachten, um die intendierten Phänomene aufzuzeigen. Der F-Wert von 84.22 entspricht dem Wert, der mit dem System erreicht wurde, das zum Zeitpunkt der MUC-7 existierte, ohne externe Systeme hinzuzuziehen (*Basic MUC-7 MENE-only System*). Darum ist der F-Wert auch um 4.58 tiefer, als derjenige,

binären Features: ohne sie würde ein F-Wert von nur 79.61 statt 84.22 erreicht werden. Überhaupt nicht respektive fast nicht produktiv sind das Sektions-Feature[36] und das Wörterbuch-Feature: Wird nur das Sektions-Feature nicht angewandt, bleibt der F-Wert unverändert 84.22, und nur ohne das Wörterbuch-Feature sinkt er sehr wenig auf 83.38. Das Referenz-Auflösungs-Feature trägt mittelmässig zum Resultat bei: Sein Mitwirken erhöht den F-Wert um etwas mehr als 2.[37]

Die genannten Zahlen führen zum Schluss, dass wirklich nutzbringend nur die lexikalischen und die binären Features sind, allenfalls noch das Referenz-Auflösungs-Feature. Betrachtet man diese drei Feature-Typen aufmerksam, so wird man feststellen, dass sie dieselbe Evidenz ausnutzen, die auch Regeln in regelbasierten Systemen benutzen. Beispielsweise entspricht der Regel (2) aus dem BSEE-System von FACILE / CONCERTO[38] eine Kombination aus dem lexikalischen Feature g_8 und einem binären Feature in der Form von ${g_1}$[39], ausser dass in g_1 f nicht als `location_start`, sondern beispielsweise als `person_start` klassifiziert würde.

Nebem dem Unterschied, dass in regelbasierten Systemen *mehr als eine* Evidenz in *eine* Regel verpackt wird, während in statistisch basierten *jeder Evidenz ein Feature* entspricht, liegt der Hauptunterschied darin, dass *regelbasierte Systeme meist ohne Gewichtung* arbeiten, weil Regeln sich dazu schlecht eignen.[40] Der grosse Nutzen eines ME-Systems, unterschiedlichste Evidenz ausnutzen zu können, wirkt sich also vor allem dahin gehend aus, dass jede Evidenz *gemäss ihrer Wichtigkeit* zur Geltung kommt, indem das ME-System ihr Gewicht errechnet. Dies eröffnet die Möglichkeit, bei der Klassifizierung auch schwache Evidenz zu berücksichtigen, die in einem regelbasierten System vernachlässigt werden müsste, weil sie zu häufig auch für Muster zuträfe, die zu falscher Klassifikation führen würde. Wie gesehen kommen aber von

---

der im MUC-7-Wettbewerb erreicht wurde (im MUC-7-Wettbewerb inkludierte MENE die Features für externe Systeme, die den Output des Systems *Proteus* miteinbezogen, womit der F-Wert von 88.80 erreicht wurde). Auch arbeitete das *Basic MUC-7 MENE-only System* noch mit weniger Features, erst mit fünf: binären, lexikalischen, Sektions-, Wörterbuch-Features und Features für externe Systeme. In [Bor99] sind noch drei weitere Features erwähnt (vgl. [Bor99], Seite 43ff:): Features für externe Systeme, Konsistenz-Features (engl. *Consistency*) und Kompositions-Features (engl. *Compound*). In Tabelle 7.5, wo Borthwick die Beiträge der einzelnen Features untersucht, fehlen die beiden letzteren, somit können für sie keine genauen Schlüsse gezogen werden. Es darf aber angenommen werden, dass die beiden nicht sonderlich viel zur Eigennamen-Erkennungs-Leistung beitragen, sonst wären sie in der Tabelle 7.5 sicherlich aufgeführt. Der Leistungsbeitrag der Features für externe Systeme wurde bereits im vorigen Punkt ausführlich erläutert.

[36] Um nicht zu sehr ins Detail zu gehen, wurde das Sektions-Feature in dieser Arbeit nicht erläutert. Für nähere Erklärungen dazu vgl. [Bor99], Seite 38f.

[37] Da das Referenz-Auflösungs-Feature erst nach der MUC-7-Tagung eingefügt wurde, wurde hier nicht wie vorhin getestet, um wieviel sich die Leistung vom *Basic MUC-7 MENE-only System* (vgl. obige Fussnote) verringert, wenn das Feature weggelassen wird, sondern um wieviel das *Basic MUC-7 MENE-only System* durch das Feature verbessert wird. Dass hier nur ein ungefährer Wert angegeben wird, liegt daran, dass beim Testen des Referenz-Auflösungs-Features mit zwei verschiedenen Trainingsdaten gearbeitet wurde und so zwei verschiedene Werte entstanden: einmal 86.26 und einmal 86.56.

[38] Vgl. Unterkapitel 5.1.2, Seite 31.

[39] Vgl. Unterkapitel 5.2.3, Formeln 5.20 und 5.31, Seiten 75 und 80.

[40] Weshalb Regel-Gewichtung schwierig ist, wird in 6.1.1 ergründet.

MENEs acht Feature-Typen neben den beiden sehr wirksamen Typen lexikalischer und binärer Features nur noch das Referenz-Auflösungs-Feature als ein taugliches schwaches Feature in Frage. Die anderen sind so schwach, dass sie keinen Einfluss mehr nehmen und deshalb als untauglich gelten dürfen.

So kommt man zum Schluss, dass MENE das Potenzial eines ME-Systems, verschiedenste Features ausnutzen zu können, nur vordergründig ausschöpft, sich in Wirklichkeit aber auf wenige Features beschränkt. Unter anderem könnte dies an der Schwierigkeit liegen, wirklich gute Features zu finden. Das Ermitteln der Features bleibt trotz aller Automatisierung Handarbeit und hängt von der Intuition der Linguistin ab. Allein die Tatsache, dass eine Konferenz wie die MUC abgehalten wurde, beweist, dass die genialen Ideen nicht so einfach zu haben sind.

6.2.4 Entscheidungs-Bäume

Das vorliegende Kapitel beinhaltet vorwiegend Gedanken zu dem in 5.2.4 beschriebenen Entscheidungs-Baum-System zur NER von Sekine. Allgemeine Probleme von Entscheidungs-Bäumen werden nur bedingt angesprochen, da diese in [Mit97] bereits ausführlich erläutert sind.[41]

Die Leistung eines NER-Systems, das mit Entscheidungs-Bäumen arbeitet, zu beurteilen, fällt hier schwerer als bei allen anderen, weil am MUC-7-NE-Task kein System mit Entscheidungs-Bäumen arbeitete. Das in Kapitel 5.2.4 beschriebene System wurde im Rahmen des *Multilingual Entity Task (MET-2)* fürs Japanische entwickelt und wurde mit anderen, insbesondere weniger Texten als die MUC-7-NE-Task-Systeme getestet. Für die MET-2-Systeme wurden eigene "Ranglisten"[42] erstellt, eine fürs Japanische und eine fürs Chinesische. Am japanischen NE-Task nahmen neben Sekine nur noch zwei andere Forschergruppen teil.[43] Sekines System erreichte mit 79.51 den tiefsten F-Wert von allen dreien. Im *Anhang A.2* können die Auswertungen der drei Systeme eingesehen werden. Auffällig ist der niedrige Personennamen-Recall-Wert bei allen drei Systemen. Sekine erklärt dies mit einem markanten Unterschied zwischen Trainingstexten und Testtexten: Weil die Textkorpora aus verschiedenen Domänen stammten, kamen in ersteren kaum ausländi-

[41] Vgl. [Mit97], Kapitel 3.7, Seite 66ff: Als allgemeine Probleme bei Entscheidungs-Bäumen nennt Mitchell Überanpassung der Daten (*3.7.1 Avoiding Overfitting the Data*), nicht-diskrete Werte (*3.7.2 Incorporating Continuous-Valued Attributes*), Favorisierungsproblem bei Attributen (*3.7.3 Alternative Measures for Selecting Attributes*), fehlende Attribut-Werte (*3.7.4 Handling Training Examples with Missing Attribute Values*) und Attribute mit unterschiedlichen Kosten (*3.7.5 Handling Attributes with Differing Costs*). Ein weiteres Problem ist auf Seite 62 angesprochen: Weil kein Backtracking gemacht wird, besteht die Gefahr, dass nur lokal optimale Lösungen gefunden werden und dass dadurch global optimale Lösungen übersehen werden könnten.

[42] Eigentlich handelt es sich dabei um eine detaillierte Auflistung der System-Auswertungen (vgl. *Anhang A.2: Punkteverteilung NE-Task - Japanisch*). Weil dabei immer auch der F-Wert angegeben ist, und man diese einander gegenüberstellen kann, ist hier von "Rangliste" die Rede. Eine solche wurde für diese Arbeit auch erstellt und ist in *Anhang B* als *Rangliste NE-Task Japanisch (MET-2)* einzusehen.

[43] Es sind dies eine Forschergruppe der Firma *Nippon Telegraph and Telephone Corporation (NTT)* und wie auch schon beim MUC-7-NE-Task eine Forschergruppe der Firma *Oki Electric Industry Co.*

sche Personennamen vor, in letzteren aber fast nur.[44] Japanische Personennamen werden in anderen Schriftzeichen-Typen geschrieben als nicht-japanische. Weil der Schriftzeichen-Typ ein wichtiges Merkmal im japanischen Schriftsystem darstellt, entstanden durch diese Inadäquatheit von Trainings- und Testtexten solch tiefe Personennamen-Recall-Werte. Diese Erklärung ist plausibel. Warum jedoch der Präzisions-Wert für Personennamen bei Sekines System bei lediglich 74 liegt, ist damit nicht zu erklären, insbesondere da die Werte bei den beiden anderen Systemen einiges höher sind (92 bei NNT und sogar 99 bei Oki).

In [Sek98b] wird über ein weiteres Experiment mit Domänenänderung berichtet: Dort wird dargelegt, dass dabei mit geringem Aufwand auch eine neue Klasse (engl. *position class*)[45] eingeführt werden kann und das Ergebnis zufriedenstellend sei. Dann muss aber eingeräumt werden, dass beispielsweise Organisationsnamen schlechter erkannt würden als in der vorigen Domäne. Dies liege unter anderem an einem Suffix, das bei chinesischen Firmennamen verwendet würde und das im Trainingskorpus (das aus der vorigen Domäne stammt) nicht vorgekommen sei. Deshalb sei das Suffix nicht im Organisations-Suffix-Wörterbuch. Obwohl der Grund für diesen Leistungsrückgang wieder in der Inadäquatheit von Trainings- und Testtexten liegt, ist hier doch auch ein Umstand angesprochen, der hier als Schwäche von Sekines NE-System angesehen wird: Die Entscheidungs-Bäume hängen viel zu sehr von den Wörterbüchern ab. Betrachtet man in [Sek98b] die Zahlen, wieviele Einträge die Wörterbücher enthalten, fällt auf, dass es sehr viele sind: 7'018 Organisationsnamen, 17'851 Personennamen, 14'863 Ortsnamen und zusätzlich Namenspräfixe und -suffixe.[46] Dies führt zur Feststellung, dass vor allem auf Grund der Wörterbücher-Attribute entschieden wird, ob ein Eigenname vorliegt oder nicht. Es ist deshalb zu vermuten, dass in dem Falle, wo weder das zu klassifizierende Token noch sein Präfix respektive Suffix in einem Wörterbuch vorkommt, das Token fälschlicherweise als `other` klassifiziert wird.

Ohne näher darauf einzugehen, sei zum Schluss noch Folgendes angemerkt:

- In Sekines Entscheidungs-Baum-System werden neben dem aktuellen Token das vorangehende und das folgende als Attribute miteinbezogen. Dieses Dreier-Fenster hätte beispielsweise zum Fünfer-Fenster vergrössert und dadurch die Präzision gesteigert werden können.

- Ein Nachteil von Sekines Entscheidungs-Baum-System ist auch, dass dieses zwar mit Informationen über Wortarten, Schriftzeichen-Typ und Wörterbücher arbeitet, die Information aber, um welches Wort es sich handelt, nicht miteinbezogen wird. Lexikalische Informationen dieser Art wären nützlich, jedoch ist es keine leichte Aufgabe, diese in Entscheidungs-Baum-Systeme zu integrieren, und bedarf elaborierter Ansätze.[47]

[44] In den Trainingstexten handelte es sich um Zeitungsberichte über Unfälle in Japan, worin selten ausländische Personen verwickelt sind. Die Testtexte hingegen handelten von Satelliten-Starts, in welche vor allem Personen nicht japanischer Herkunft involviert sind.

[45] Mit der *position class* sollten Ausdrücke wie *President*, *Professor* als NE erkannt werden.

[46] Vgl. *Table 2* in [Sek98b].

[47] Dieser Ansicht ist auch Borthwick; vgl. [Bor99], Seite 13.

- Zur Wahl der Attribute wurde bereits angemerkt, dass die Wörterbuch-Attribute zu einflussreich seien. Welche Attribute sinnvoll sind und deshalb miteinbezogen werden sollen respektive welche nicht, bleibt auch hier der guten Intuition der EntwicklerInnen überlassen.

6.2.5 Fazit für statistisch basierte Systeme

Bei statistisch basierten Systemen scheinen auf den ersten Blick keine vergleichbaren Nachteile wie die in 6.1.3 geschilderten Nachteile regelbasierter Systeme zu existieren. Die Leistung statistisch basierter Systeme korreliert nicht mit der Grösse von Ressourcen an guten Indikatoren und Regeln. Während bei regelbasierten Systemen die aufwändige Erstellung *vieler* guter Indikatoren und Regeln auch *viel* Zeit und Geld kostet, kann bei statistisch basierten Systemen oft auf Standard-Module zurückgegriffen werden, mit Hilfe derer schon beachtliche Leistungen erbracht werden können. Entscheidungs-Baum-Systeme basieren beispielsweise auf einem Standard-Algorithmus von Quinlan, und auch MENE setzt ein "Ab-der-Stange-Werkzeug"[48] ein. Bei näherer Betrachtung fällt jedoch auf, dass auch in statistisch basierten Systemen noch viel Handarbeit steckt, die sehr zeitaufwändig sein kann: Angefangen beim Taggen der Trainingskorpora, über das Finden guter Features (MENE) respektive guter Attribute (Entscheidungs-Bäume) bis hin zu Verfeinerungen zur Leistungssteigerung[49] der Systeme. Um also ein *ausgefeiltes* statistisches System zu bauen, werden wahrscheinlich ebenso Ressourcen an Zeit und Geld benötigt wie beim Bau eines leistungsfähigen regelbasierten Systems. Diese vorsichtig formulierte Folgerung kann gestützt werden, wenn man sich die Ranglisten des MUC-7-NE-Tasks[50] ansieht: Es ist nicht festzustellen, dass statistisch basierte Systeme eine bessere Leistung erbrächten als regelbasierte oder umgekehrt. Wird jedoch nur *grobe* Eigennamen-Erkennung benötigt und steht

[48] Vgl. die Textstelle aus [Bor99], die auf Seite 78 zitiert wird.

[49] Um bei Entscheidungs-Bäumen auch die Information einzubeziehen, um welches Token es sich beim zu klassifizierenden handelt, reicht das Standard-Modul nicht aus; eine Integration der gewünschten Information wäre sehr aufwändig (vgl. dazu den zweitletzten Punkt in Unterkapitel 6.2.4). Bei MENE erfordert das Einbeziehen von Features für externe Systeme, dank denen der F-Wert auf eine Höhe gesteigert werden kann, die den F-Werten menschlicher Annotatoren gleichkommt, ebenfalls einen sehr grossen Zeitaufwand (vgl. Anmerkungen Seite 102). Auch bei IdentiFinder wird angemerkt, dass durchaus Leistungsverbesserungen möglich sind und angestrebt werden. In [Bik99], Seite 17, steht dazu Folgendes:

> *Further Work*
> While our initial results have been quite favorable, there is still much that can be done potentially to improve performance and completely close the gap between learned and rulebased name-finding systems. We would like to incorporate the following into the current model:
>
> - a hierarchical model to capture nested names, e.g., Bank of Boston
> - longer-distance information, to find names not captured by our bigram model
> - training heuristics to supplement annotated data with large volumes of unmarked language.

[50] Vgl. *Anhang B.*

wenig Zeit und Geld zur Verfügung, so ist mit einem statistisch basierten System vermutlich eine bessere Leistung zu erreichen als mit einem regelbasierten.

Will man ein Eigennamen-Erkennungs-System bauen, das einfach in andere Domänen und Sprachen portiert werden kann, so wählt man mit Vorteil ein statistisch basiertes. Der Mehraufwand beim Portieren besteht hauptsächlich darin, dass Trainingstexte aus der neuen Domäne respektive Sprache getaggt werden müssen. Im Vergleich zum Schreiben neuer Regeln, wie es bei regelbasierten Systemen nötig wäre, ist dies ein geringer Aufwand, wenn auch ein nicht zu vernachlässigender. Im Folgenden soll ergründet werden, welcher zusätzliche Aufwand - neben dem bereits erwähnten zusätzlichen Taggen von Trainingskorpora - bei den einzelnen statistischen Systemen nötig ist.

Bei MENE können die nutzbringenden[51] Feature-Typen weitgehend unverändert (oder mit wenig Aufwand angepasst) übernommen werden - was jedoch nicht ausschliesst, dass neue hinzugefügt werden könnten. Ein gewisser Leistungsabfall bei der Portierung kann erwartet werden, dieser dürfte aber gering sein. Auch bei BNN IdentiFinder ist bei der Protierung in eine andere Sprache nur geringe Modifikation nötig.[52] Die Portierung in eine andere Domäne sollte keinen besonderen Mehraufwand nötig machen. Einzig bei Entscheidungs-Bäumen ist zusätzlicher, nicht unerheblicher Aufwand nötig. Dies vor allem weil die Attribut-Art *spezielle Wörterbücher* mit Listen von Eigennamen arbeitet, welche bei der Portierung neu erstellt werden müssten. Zudem ist fraglich, ob die Attribut-Art *Schriftzeichen-Typen* noch von Nutzen sein wird, wenn ein Entscheidungs-Baum-System beispielsweise in eine Sprache portiert wird, die das lateinische Alphabet verwendet. Doch abgesehen von all diesem soeben erwähnten Mehraufwand kann grundsätzlich festgestellt werden, dass statistisch basierte Systeme leichter in andere Domänen und Sprachen zu portieren sind als regelbasierte. Regelbasierte Systeme sind nur mit grossem Aufwand portierbar, da die verwendeten Regeln häufig starr auf einen kleinen, sprach- und domänenabhängigen Kontext ausgerichtet sind.

6.3 Anwendung auf Produktenamen

6.3.1 Regelbasiertes Erkennen von Produktenamen

Dass es enorm zeitaufwändig ist, gute Regeln zu finden, wurde auch in einer Arbeit über automatisches Erkennen von Produktenamen festgestellt: In [Rot01] werden rund 90 potenzielle Regeln oder auch nur Indikatoren aufgeführt, die vielleicht zur Erkennung von Produktenamen verwendet werden könnten, die jedoch noch modifiziert und dann getestet werden müssen, ob damit wirklich brauchbare Resultate erzielt werden können. Da modifizieren und testen so zeitaufwändig ist, dass es den Rahmen einer Seminararbeit überstiegen hätte, beschränkte man sich in [Rot01] auf das Erstellen von nur 11 Regeln, mit denen

[51] Mit "nutzbringenden Feature-Typen" sind diejenigen gemeint, die in 6.2.3 als solche identifiziert worden sind (vgl. Seite 103).

[52] Vgl. Unterkapitel 6.2.2, dort besonders die Fussnote darüber, warum IdentiFinder nicht vollständig sprachunabhängig ist, wie wenig jedoch der sprachabhängige Teil ausmacht.

keine hohe Ausbeute, aber dafür eine hohe Präzision erreicht wurde.[53]

Neben dem grossen Zeitaufwand kommt bei Produktenamen erschwerend hinzu, dass sich die Faktoren, die regelbasierte Systeme ausnutzen, oft weniger gut eignen als bei Personen-, Orts- und Organisationsnamen. Im Folgenden werden die wichtigsten Methoden von regelbasierten Systemen dahin gehend beleuchtet, inwieweit sie zur Produktenamen-Erkennung nützlich sein könnten.

- Auf Grund der Kurzlebigkeit[54] von Produktenamen kann nicht mit vorgängig erstellten *Produktenamenlisten* gearbeitet werden, wie dies bei den meisten regelbasierten Systemen mit entsprechenden Listen getan wird.

- Auch *Listen* in der Art von Vornamenslisten zur Erkennung von Nachnamen (also für Personennamen) gibt es für die Produktenamen-Erkennung nicht.

- Interne und externe Evidenz, die von Regeln ausgenutzt wird, ist häufig nicht genügend ausgeprägt, dass man ausschliesslich mit Regeln arbeiten könnte.

  Externe Evidenz, die sich im Kontext eines Produktenamens findet, ist äusserst domänenabhängig: Geht es um die Domäne *Computer*, werden andere Indikatoren gebraucht als beispielsweise in der Domäne *Autos*. Dieses Problem kann auch mit domänenspezifischen Indikatoren-Listen nicht so einfach gelöst werden: Indikatoren alleine reichen nicht aus, um Eigennamen - in diesem Fall Produktenamen - zu finden. Erst wenn diese Eingang in konkrete Regeln finden, können sie angewandt werden. Doch gerade darin, solche Regeln zu finden, liegt die Schwierigkeit: In [Rot01] wird gezeigt, dass viele potenzielle Regeln wieder verworfen werden mussten, weil sie zu ungenau waren (zu kleine Präzision).[55] Dies rührt daher, dass bei regelbasierten Systemen nicht auf Grund mehrerer einzelner Indizien berechnet wird, wie hoch die Wahrscheinlichkeit ist, dass es sich um einen Eigennamen handelt, sondern dass immer auf Grund eines komplexen, starren Kontextes sogleich entschieden wird, ob das fragliche Token ein Eigenname ist oder nicht.[56] Regeln sind also sehr unflexibel, was bei der Erkennung von Produktenamen deshalb ein Handicap ist, weil es wie dargelegt kaum konstante Indikatoren gibt, auf die sich Regeln abstützen könnten.

  Interne Evidenz ist bei Produktenamen äusserst gering. Indikatoren wie zum Beispiel *Inc.*, *Corp.*, *GmbH* usw. bei Organisationsnamen gibt es bei Produktenamen nicht. Manchmal enthalten Produktenamen eine oder mehrere Ziffern, manchmal auch Sonderzeichen wie Punkt, Komma, Bindestrich etc. (beispielsweise im Produktenamen des Computer-Monitors *BENQ Office FP751 Sound 17TFT* oder des Autos *BMW Z3*

[53] Vgl. [Rot01], Seite 20: der Ausbeute-Wert liegt bei 23.5% , der Präzisionswert bei 100%.

[54] Vgl. zur Kurzlebigkeit Unterkapitel 2.4.2.

[55] Vgl. [Rot01], Seite 10: "Zahlreiche solcher Muster habe ich im ersten Schritt gesammelt und im zweiten auf deren Güte getestet. Viele mussten wieder verworfen werden, weil sie zuviele Ausdrücke lieferten, bei denen es sich nicht um die gewünschten Produktenamen handelte."

[56] Der Versuch, mittels Gewichtung der Regeln diesen ihre Starrheit zu nehmen, wurde zwar unternommen (vgl. BSEE von FACILE / CONCERTO und NetOwl), scheint aber, wie in 6.1.2 dargelegt, wenig erfolgreich zu sein.

2.8i Roadster). Auch kommt es vor, dass bei Produktenamen die Gross- und Kleinschreibung in unüblicher Weise angewandt wird: Zum Beispiel kann inmitten des Namens ein Grossbuchstabe stehen (beispielsweise im Namen des Textmining-Systems *dtSearch* oder des Druckergerätes *HP DeskJet*), oder der Produktename kann ausschliesslich aus Grossbuchstaben - gelegentlich noch in Kombination mit Ziffern - bestehen (zum Beispiel *FULL FILL* - der Produktename einer Früchteschale - oder *MD 441 U* - der Name eines Mikrofons der Firma Sennheiser). Doch solche Fälle sind zu wenig häufig, dass sie die Ausbeute merklich verbessern würden. Zudem beschränken sich solche Schreibweisen nicht auf Produktenamen. Auch Abkürzungen[57] bestehen häufig nur aus Grossbuchstaben (zum Beispiel *HTML*, *FSME*[58], *WWF* oder *IBM*), und das Vorhandensein eines Grossbuchstabens inmitten des Wortes ist wiederum auch bei Organisationsnamen zu beobachten (zum Beispiel *UniSpital Zürich* oder *SchauSpielHausZürich* und auch die gleichnamige Herstellerfirma des oben erwähnten Textmining-Systems *dtSearch*). Einzig eine Ziffer als Bestandteil eines Namens ist eine äusserst zuverlässige interne Evidenz für Produktenamen.[59] Auch Grossschreibung am Wortbeginn - die wichtigste interne Evidenz, die anzeigt, dass es sich um einen Eigennamen handeln kann, und die dementsprechend auch von allen regelbasierten Systemen ausgenutzt wird - darf auch bei Produktenamen als ein ziemlich zuverlässiger erster Hinweis darauf angesehen werden, dass es sich um einen solchen handeln könnte.

- *Koreferenz-Regeln* wären sicherlich auch bei der Produktenamen-Erkennung einsetzbar. Allerdings sind Koreferenz-Regeln nur dann effektiv, wenn zuvor schon viele Produktenamen erkannt worden sind. Wenn der Ausbeute-Wert vor dem Einsatz von Koreferenz-Regeln klein ist, wird er mit deren Einsatz nur wenig steigen.

6.3.2 Statistisch basiertes Erkennen von Produktenamen

Rein regelbasierte Systeme sind zur Produktenamen-Erkennung also eher ungeeignet, weil zur Erhöhung der Ausbeute mehr Regeln nötig sind. Diese verursachen aber, weil sie zu wenig flexibel[60] sind, zugleich eine höhere Fehlerrate, das heisst ein Sinken der Präzision. Eine andere Möglichkeit, das Problem der Starrheit zu umgehen, ist, die Regeln zu modularisieren, wie dies in MENE geschieht: Die Indizien, die auf einen Eigennamen hinweisen, werden in Form von Features dargestellt, welche gewichtet und mit den Gewichten anderer Features verrechnet werden. Als Resultat ergibt sich der Wert der Wahrscheinlichkeit, mit welcher das Token, dessen Features verrechnet werden, ein bestimmter Eigenname ist. Nun wurde aber in 6.2.3 festgestellt, dass die in MENE verwendeten Features, die auch

[57] Alle Arten von Abkürzungen, insbesondere aber auch solche von Organisationsnamen.

[58] *FSME* bedeutet Früh-Sommer-Meningo-Encephalitis und ist die übliche Abkürzung für eine Krankheit, die durch Zecken übertragen wird.

[59] Hiervon können Namen von Anlässen, Tagungen usw., die eine Jahreszahl enthalten, eine Ausnahme bilden; bspw. *Expo.02*. Allerdings kann man bei grosszügiger Sichtweise diese auch als Produktenamen ansehen.

[60] Zur Inflexibilität von Regeln vgl. Bemerkungen über externe Evidenz in 6.3.1.

wirklich nutzbringend sind, dieselbe Evidenz ausnutzen, die auch Regeln in regelbasierten Systemen benutzen - allerdings mit dem entscheidenden Unterschied, dass in MENE die Evidenz gewichtet werden kann. Letztlich musste aber doch festgestellt werden, dass MENE das Potenzial eines ME-Systems zu wenig ausschöpft, was vor allem an der Schwierigkeit liegt, gute Features zu finden. Diese sind jedoch Voraussetzung für erfolgreiche Eigennamen-Erkennung mit einem ME-System und somit kann hier Folgendes geschlossen werden: Produktenamen-Erkennung mittels eines ME-Systems ist eine prüfenswerte Variante. Wichtigster Faktor für den Erfolg sind leistungsstarke Features. Falls diese nicht gefunden werden können, muss das Verwenden eines ME-Systems zur Produktenamen-Erkennung in Frage gestellt werden.

In Unterkapitel 6.2.4 wurde erwähnt, dass bei Entscheidungs-Bäumen experimentiert wurde, eine neue Klasse (*position class*) einzuführen, wobei das Resultat als zufriedenstellend bewertet wurde. Dieser Umstand spricht *für* eine Eignung von Entscheidungs-Bäumen zur Produktenamen-Erkennung. *Dagegen* spricht jedoch die grosse Abhängigkeit von Wörterbüchern. Wie bereits in 2.4.2 dargelegt, sind Produktenamen äusserst kurzlebig - und Produktenamen-Listen daher kein probates Mittel, um sie zu erkennen. Da sowohl das in 5.2.4 ausführlich vorgestellte System von Sekine als auch *RoboTag* von Bennett et al.[61] Listen mit Eigennamen verwenden, müssen sie als ungeeignet zur Produktenamen-Erkennung eingestuft werden.

Bleibt die Frage, ob sich ein HMM zur Erkennung von Produktenamen eignet. Für eine Eignung sprechen vor allem die Faktoren, dass das HMM, so wie es bei IdentiFinder umgesetzt wurde, erstens keine Eigennamen-Listen verwendet und zweitens einen überzeugenden F-Wert vorweist. Scheinbar dagegen spricht, dass ein HMM nicht - so wie oben zur Umgehung der Starrheit von Regeln vorgeschlagen - mehrere und unterschiedliche Merkmale ausnutzen kann. Doch braucht es dies auch nicht zwingend zu tun, da dieses Unterteilen in einzelne Features nur darum getan wird, um die Starrheit der Regeln aufzubrechen. Ein HMM kommt jedoch ohne Regeln aus, das Problem stellt sich also erst gar nicht. Eher könnten sich die Probleme als kritisch erweisen, die bereits in 6.2.2 besprochen wurden: Der zu spezifische respektive zu kleine Kontext, der das Sparse-Data-Problem verschärft. Doch sind Lösungsansätze vorhanden, wobei hier nicht beurteilt werden kann, wie geeignet diese sind.[62] Wie beim ME-System kann aber gesagt werden, dass Produktenamen-Erkennung mittels eines HMM-Systems eine prüfenswerte Variante ist. Wichtigster Faktor für den Erfolg sind diesmal ein grosses Trainingkorpus, damit das Sparse-Data-Problem entschärft werden kann. Dass sich die Leistung von IdentiFinder durchaus noch steigern lässt und angestrebt wird, wurde bereits in 6.2.2 und in 6.2.5[63] angemerkt. Solch eine Leistungssteigerung würde sich dann auch auf den Bereich der Produktenamen-Erkennung auswirken, was natürlich begrüssenswert wäre.

[61] Vgl. Seite 90.
[62] Vgl. dazu wieder 6.2.2.
[63] Vgl. in 6.2.5 vor allem die lange Fussnote mit dem Zitat aus [Bik99].

6.4 Schluss

In vorliegender Arbeit wurde nur zwischen regelbasierten und statistischen Systemen unterschieden. Eine dritte Gruppe wäre denkbar gewesen: hybride Systeme, die die beiden Ansätze kombinieren. Als Vertreter von hybriden Systemen hätten PIE und LT TTT stehen können: PIE war anfänglich ein regelbasiertes System, bei dem durch Erweiterung um Kollokations-Statistik eine höhere Leistungsfähigkeit erreicht wurde. LT TTT wurde hier unter den regelbasierten vorgestellt, obwohl auch viele statistische Techniken angewandt werden.[64]

Mit hybriden Systemen wird versucht, von den jeweiligen Vorteilen eines Ansatzes zu profitieren, um gleichzeitig die jeweiligen Nachteile zu überbrücken. Dass dieses Ansinnen mit gewissem Erfolg umgesetzt werden kann, zeigt zum einen die Leistungssteigerung bei PIE durch Erweiterung mittels Kollokations-Statistik[65], zum anderen die Tatsache, dass LT TTT beim MUC-7-Wettbewerb am besten abgeschnitten hat.

Angespornt durch diese Erfolge möchte man nun gerne behaupten, der hybride Ansatz sei die optimale Lösung. Doch dieser Schluss ist ein verfrühter. Zweifellos ist die Kombination von regel- und statistisch basierten Systemen ein vielversprechender Ansatz, doch dass es der beste wäre, kann auf Grund der erwähnten Erfolge von PIE und LT TTT nicht geschlossen werden. Denn wäre es wirklich der beste, so hätte PIE beim MUC-7-Wettbewerb gleich hinter oder vor LT TTT platziert sein müssen, also mindestens den 2. Rang erreichen müssen. Tatsächlich erreichte das hybride System PIE jedoch "nur" den 5. Rang. Es kommt hinzu, dass hybride Systeme komplex sind. Während beispielsweise MENE ein System ist, das ausschliesslich auf ME beruht, so werden in LT TTT mehrere ME-Modelle als jeweils *eine* Methode unter vielen eingesetzt.[66] Auch ein HMM verwendet LT TTT, und Eigennamen-Erkennung geschieht in fünf Schritten, wobei wiederum statistische Methoden eingesetzt werden. LT TTT ist also ein äusserst komplexes und somit aufwändiges System. Dasselbe lässt sich ohne weitere Erläuterung von PIE sagen, da es augenfällig ist, dass es mit der Erweiterung um Kollokations-Statistik aufwändiger ist als ohne. Stehen nicht grosse Ressourcen an Zeit und Geld zur Verfügung, ist der hybride Ansatz eher ungeeignet.

Man kommt also zu dem schon eingangs dieses Kapitels erwähnten Schluss, dass ein endgültiges Urteil, welcher Ansatz der beste ist, nicht gefällt werden kann. Dieser Schluss, der bereits in 6.2.5 ausgeweitet wurde, kann nun weiter vervollständigt werden:

Es ist nicht festzustellen, dass einer der drei Ansätze - regelbasiert, statistisch oder

[64] Weitere Systeme, die beide Techniken kombinieren, wurden bereits in 6.1.2 erwähnt. Dort ist auch nachzulesen, warum es sich nicht lohnt, deren hybriden Charakter weiter zu verfolgen. Auch MENE wird in [Bor99] als System beschrieben, das beide Ansätze kombiniert, weil in das statistische ME-System der Output des regelbasierten Systems Proteus einbezogen wird. Dennoch wird MENE in vorliegender Arbeit nicht als hybrides System gewertet, da nicht die *Methoden*, sondern lediglich der *Output* von Proteus für MENE verwendet wurden. Auf welche Art der Output gewonnen wurde, ist irrelevant. Dies ist auch daran zu sehen, dass in der Weiterentwicklung von MENE zusätzlich der Output von PIE integriert wurde (PIE wird in vorliegender Arbeit bei den statistischen Systemen behandelt).

[65] Zur Leistungssteigerung von PIE vgl. den ersten Punkt in 6.2.1.

[66] Beispielsweise wird ein ME-Modell eingesetzt, um Punkt-Satzzeichen zu disambiguieren, oder eines, um Wortarten zu taggen.

hybrid - generell eine bessere Leistung erbrächte als die anderen. Wird nur eine grobe Eigennamen-Erkennung benötigt und steht wenig Zeit und Geld zur Verfügung, so ist mit einem statistisch basierten System vermutlich die beste Leistung zu erreichen. Eine Erweiterung eines regelbasierten System um statistische Methoden kann die Leistung des Systems verbessern, macht es aber zugleich komplexer und somit zeit- und kostenintensiver. Einzig bezüglich Portierbarkeit in andere Domänen und Sprachen ist festzustellen, dass sich dafür statistische Systeme am besten eignen. Zur Erkennung von Produktenamen sind statistische und hybride Systeme dann geeignet, wenn sie nicht mit Listen von Eigennamen arbeiten. Rein regelbasierte Systeme eignen sich schlecht zur Produktenamen-Erkennung.

Zusammenfassung

Maschinelle Sprachverarbeitung findet heute viele - unterschiedlich komplexe - Anwendungen: Suchmaschinen, maschinelle Übersetzung, Textmining, Informations-Extraktion usw. Ein wichtiger Teilschritt auf dem Wege zur maschinellen Sprachverarbeitung ist die Eigennamen-Erkennung, weil sie zum einen die syntaktische und semantische Analyse unterstützt und zum anderen als Modul in einer komplexeren Anwendung, wie zum Beispiel Informations-Extraktion, dient.

Ziel dieser Arbeit ist, den Stand der Kunst in der Eigennamen-Erkennung zu ermitteln. Dazu werden die folgenden Fragen gestellt: Welches sind in der Computerlinguistik die heute üblichen Ansätze oder Methoden, um in Texten Eigennamen automatisch zu erkennen? Welche erbringen die beste Leistung? Die meisten Systeme, die Eigennamen erkennen können, beschränken sich auf Personen-, Orts- und Organisationsnamen. Hier interessiert zudem die Frage, wie sich die derzeitigen Ansätze eignen, die ebenfalls wichtige Klasse von Produktenamen zu erkennen.

Zur Beantwortung dieser Fragen hat sich die Message Understanding Conference (MUC-7) als reichhaltiger Fundus herausgestellt. Von den an der MUC-7 teilnehmenden Systemen, die Eigennamen erkennen können, werden für diese Arbeit die meisten der englischen Systeme und ein japanisches berücksichtigt. Es sind dies:

- *BSEE von FACILE/CONCERTO* (UMIST, University of Manchester, UK)
- *LaSIE* (NLP group, University of Sheffield, UK)
- *LT TTT* (Language Technology Group, University of Edinburgh, UK)
- *NetOwl* (IsoQuest Inc., Fairfax, VA/USA)
- *Oki Informations-Extraktions-System* (Oki Electric Industry Co., Ltd., Osaka, Japan)
- *LOLITA* (Laboratory for Natural Language Engineering, Department of Computer Science, University of Durham, UK)
- *PIE-System mit Erweiterung der Eigennamen-Erkennung durch Kollokations-Statistik* (Department of Computer Science, University of Manitoba, Winnipeg, Canada)
- *IdentiFinder* (BBN Technologies, Cambridge, MA/USA)
- *MENE* (Computer Science Department, New York University, New York, NY/USA)

- *Japanese NE-System used for MET-2* (Computer Science Department, New York University, New York, NY/USA)

Zusätzlich wird ein System vorgestellt, dessen Beschreibung als Klassiker in der Eigennamen-Erkennungs-Literatur gelten darf: *Proper Name Facility* (*PNF*) des Systems *Sparser*. Entwickler: David D. McDonald, damals (1993) als ausserordentliches Fakultätsmitglied an der Brandeis University, Waltham, MA/USA und Berater (*Consultant*) finanziert von der ARPA und kommerziellen Auftraggebern.

Im Zusammenhang mit dem Begriff *Eigenname* taucht ein weiterer Begriff auf: *Named Entity (NE)*. Bei der Einordnung der beiden Begriffe können zwei wichtige Punkte festgestellt werden: Zum einen, dass häufig das sprachliche Zeichen (der Eigenname) und das Bezeichnete (die NE) miteinander vermengt werden. Mehr noch: Der Begriff *Named Entity* wird meist dann verwendet, wenn eigentlich das sprachliche Zeichen (und nicht, wie erwartet, die Entität) gemeint ist. Der ebenfalls häufig verwendete Begriff *Named Entity Recognition (NER)* meint also Erkennung von sprachlichen Zeichen für Named Entities (NEs). Zum anderen stellt man fest, dass sehr unterschiedliche Auffassungen darüber herrschen, was unter Eigennamen oder eben NEs gezählt werden soll. Dies hängt wohl auch und gerade mit der Begriffsvermengung zusammen. An der MUC-7 werden neben den erwähnten drei Eigennamenarten auch Angaben von Datum, Zeit, Prozenten und Geldbeträgen als NEs definiert. Andere betrachten ausschliesslich Bezeichner für einzelne Lebewesen und Dinge (wie zum Beispiel Länder, Städte, Strassen, Schiffe, Gewässer, Sterne usw.) als Eigennamen respektive NEs, wobei auch hier die Grenzen wieder verschieden gezogen werden. Für diese Arbeit werden nur Bezeichner für Personen, Orte, Organisationen und Produkte als Eigennamen definiert (vier Kategorien).

Die Probleme, die sich bei der Eigennamen-Erkennung stellen, sind sehr ähnlich wie bei der Bestimmung gewöhnlicher Nomen und Nominalphrasen. Dies betrifft vor allem Abgrenzungsprobleme hinsichtlich Struktur und Semantik.

Bei den Lösungsansätzen zur Eigennamen-Erkennung können grundsätzlich zwei Arten unterschieden werden: regelbasierte und statistisch basierte. Regelbasierte Systeme wenden Regeln unter Verwendung von regulären Ausdrücken an. Ausgenutzt wird dabei vor allem Grossschreibung (Eigennamen sind zumeist grossgeschrieben) und Koreferenz. Wichtigste Hilfsmittel sind Listen von Indikatoren (wie zum Beispiel *GmbH* bei Firmen) und vorgefertigte Listen von Eigennamen. Vereinzelt werden Filter, Syntax-Analyse und Gewichtung der Regeln eingesetzt. Gewichtung wird deshalb angewandt, um die Starrheit von Regeln zu umgehen, das heisst, um Regeln flexibler zu machen. Allerdings stellt sich heraus, dass diese Idee sehr schwierig umzusetzen ist und die beiden Systeme, die Gewichtung einsetzen, mit ihrer Umsetzung nicht überzeugen können. Ein weiteres Problem beim regelbasierten Ansatz ist, dass die Leistung eines Systems stark von der Anzahl und der Qualität der Regeln abhängig ist. Dies bedeutet, dass eine unterschiedlich gute Leistung von regelbasierten Systemen ihre Wurzeln weniger in unterschiedlichen Methoden hat als in unterschiedlich grossen Mengen an Zeit und Geld, die bei der Entwicklung zur Verfügung stehen. Die erwähnte Starrheit von Regeln macht sich auch darin bemerkbar, dass regelbasierte Systeme stark sprach- und domänenabhängig sind. Sollen sie in eine andere Sprache

oder Domäne portiert werden, ist erneut kostenintensive und zeitaufwändige Arbeit nötig, wobei bereits verrichtete Arbeit grösstenteils nutzlos wird.

Die geschilderten Nachteile von regelbasierten Systemen sind eine Motivation, dass andere Systeme mit einem statistisch basierten Ansatz arbeiten. Die in dieser Arbeit vorgestellten statistischen Systeme arbeiten sehr verschieden, weshalb sich auch in der Methodik wenig Gemeinsamkeiten festmachen lassen. Einzige methodische Gemeinsamkeit ist, dass immer ein getaggtes Trainingskorpus benötigt wird, auf Grund dessen ein Sprachmodell erstellt wird, das sodann auf ungetaggte Texte angewandt werden kann. Als Folge dieser Methodik ergibt sich das allen statistischen Systemen gemeinsame Sparse-Data-Problem. Oft können für statistische Systeme Standard-Module eingesetzt werden, was den Vorteil hat, dass ohne grossen Aufwand bereits eine gewisse Leistung erbracht werden kann. Um jedoch ein statistisches System zu bauen, das wettbewerbsfähige Leistung erbringt, reicht das blosse Verwenden eines Standard-Moduls nicht aus - teilweise aufwändige Handarbeit wie beispielsweise geschicktes Ermitteln guter Merkmale ist nötig.

Zuletzt kann man auch Mischformen der beiden Ansätze feststellen. Mit hybriden Systemen wird versucht, von den Vorteilen beider Ansätze zu profitieren, um die jeweiligen Nachteile zu kompensieren. Es lässt sich feststellen, dass eine Erweiterung eines regelbasierten Systems um statistische Methoden die Leistung des Systems verbessern kann und der hybride Ansatz es deshalb wert ist, im Auge behalten zu werden. Trotz dieser Erfolge kann nicht grundsätzlich behauptet werden, dass der hybride Ansatz der beste wäre. Zugleich ist er der komplexeste und somit zeit- und kostenintensiv in der Entwicklung.

Man kommt also zum Schluss, dass sich nicht grundsätzlich sagen lässt, einer der drei Ansätze eigne sich am besten zur Entwicklung eines ausgefeilten Eigennamen-Erkennungs-Systems. Allenfalls lässt sich festhalten, dass bei geringen finanziellen und zeitlichen Ressourcen der statistische Ansatz die beste Leistung verspricht. Auch bezüglich Portierbarkeit in eine andere Sprache oder Domäne ist ein statistisches System das geeignetste. Regelbasierte Systeme sind nur mit grossem Aufwand portierbar, da die verwendeten Regeln häufig starr auf einen kleinen, sprach- und domänenabhängigen Kontext ausgerichtet sind.

Bezüglich der Eignung der zwei respektive drei Ansätze zur Produktenamen-Erkennung kann festgehalten werden, dass sich rein regelbasierte Ansätze schlecht dafür eignen. Bei Produktenamen fehlen Indikatoren, wie sie in all den beschriebenen regelbasierten Systemen in der Hauptsache angewandt werden. Können sich die inflexiblen Regeln nicht auf klare Indikatoren abstützen, bewirkt dies eine hohe Fehlerrate. Daher eignen sich zur Produktenamen-Erkennung besser statistische und hybride Systeme, aber nur dann, wenn sie nicht mit Listen von Eigennamen - in diesem Falle Produktenamen - arbeiten. Grund für diese Einschränkung ist die Kurzlebigkeit von Produktenamen: Produktenamen von heute können morgen schon veraltet sein, dafür sind morgen bereits neue Produktenamen erfunden.

Anhang A

Punkteverteilungen

In diesem Unteranhang finden sich die Punkteverteilungen (*scores*), die an der MUC-7 vorgenommen wurden. Die Tabellen sind der MUC-7-Homepage[1] entnommen. Obwohl für diese Arbeit nur Teilabschnitte der Tabellen von Belang sind, wurden sie der Vollständigkeit halber (fast)[2] ungekürzt übernommen. Im Folgenden werden Hinweise gegeben, um die wichtigsten Teile zu verstehen. Wer Näheres über die Lesart der Tabellen erfahren möchte, sei in erster Linie auf [Chi] verwiesen.[3]

Durch waagrechte gestrichelte Linien ist eine Tabelle in verschiedene Gebiete unterteilt: `SUBTASK SCORES`, `SECT SCORES`, `OBJ SCORES`, `SLOT SCORES` und `ALL SLOTS`. Hier interessiert ersteres. Die Subtasks beziehen sich auf das Erkennen derjenigen Ausdrücke, die an der MUC als NEs angesehen werden, und werden dementsprechend mit `organizatio[n]`, `person`, `location`, `date`, `time`, `money` und `percent` bezeichnet. Diese sieben Subtasks sind in drei Gruppen unterteilt: `enamex`, `timex` und `numex`. Die in dieser Arbeit als Eigennamen bezeichneten Ausdrücke finden sich in der Gruppe `enamex`: Organisationsnamen (`organizatio[n]`), Personennamen (`person`) und Ortsnamen (`location`).

In der Spalte `POS` ist abzulesen, wieviele NEs im jeweiligen Subtask zu finden sind. `ACT` gibt die Anzahl an, wieviele vom evaluierten System getaggt wurden - was noch keine Aussage darüber ist, ob richtig oder falsch getaggt wurde. Erst die weiteren Spalten geben Auskunft über die Korrektheit des Taggens: `COR` (*correct*) gibt die Anzahl der korrekt getaggten und `INC` (*incorrect*) die Anzahl der anders als im Lösungsschlüssel getaggten NEs. Im Gegensatz dazu gibt `SPU` (*spurious*) die Anzahl derjenigen Ausdrücke an, die keine NEs sind, vom evaluierten System aber fälschlicherweise als solche getaggt wurden. Die mit `MIS` (*missing*) überschriebene Spalte sagt aus, wieviele NEs vom System nicht gefunden wurden. Und `REC` (*Recall*) und `PRE` (*precision*) geben Ausbeute und Präzision pro Subtask an.

Interessant ist auch das Gebiet `OBJ SCORES`. Hier ist jeweils die Gesamtzahl aller NEs einer Gruppe (`enamex`, `timex`, `numex`) angegeben.

In der untersten Zeile sind drei F-Werte (`F-MEASURES`) aufgeführt. Der Unterschied

[1] Vgl. [Chi01a] und [Chi01b].

[2] Lediglich die Zeilen, die im Original überall die Werte 0 haben, wurden weggelassen.

[3] Ausführlichere Informationen finden sich in [Chi97b], [Chi92] und [Vil95].

zwischen den dreien ist die Gewichtung von Ausbeute (`R` für *Recall*) und Präzision (`P` für *Precision*). Beim ersten von links werden beide gleich gewichtet, beim zweiten wird der Präzisions-Wert doppelt gezählt, beim dritten zählt der Ausbeute-Wert doppelt. In dieser Arbeit wird immer nur von einem F-Wert gesprochen: Damit ist derjenige gemeint, der an der MUC-7 benutzt wird, um die Leistung der Systeme mit anderen zu vergleichen - derjenige, bei dem Ausbeute und Präzision gleich gewichtet werden (`P&R`).

A.1 Punkteverteilung NE-Task - Englisch

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	1880	1831	1828	0	1	51	2	5	97	100	3	0	0	3
person	887	883	882	0	1	4	0	0	99	100	0	0	0	1
location	1324	1311	1307	0	4	13	0	5	99	100	1	0	0	1
timex														
date	1261	1323	1257	0	1	3	65	1	100	95	0	5	0	5
time	220	231	219	0	1	0	11	0	100	95	0	5	0	5
numex														
money	227	229	227	0	0	0	2	0	100	99	0	1	0	1
percent	100	103	100	0	0	0	3	0	100	97	0	3	0	3
SECT SCORES														
slug	158	154	154	0	0	4	0	3	97	100	3	0	0	3
date	110	110	110	0	0	0	0	0	100	100	0	0	0	0
nwords	90	90	90	0	0	0	0	0	100	100	0	0	0	0
preamble	1242	1236	1225	0	3	14	8	7	99	99	1	1	0	2
text	9779	9810	9527	0	128	124	155	378	97	97	1	2	1	4
trailer	408	408	408	0	0	0	0	0	100	100	0	0	0	0
OBJ SCORES														
enamex	4081	4015	4013	0	0	68	2	10	98	100	2	0	0	2
numex	327	332	327	0	0	0	5	0	100	98	0	2	0	2
timex	1480	1550	1477	0	0	3	73	1	100	95	0	5	0	5
SLOT SCORES														
enamex														
text	4081	4015	4010	0	3	68	2	10	98	100	2	0	0	2
type	4091	4025	4017	0	6	68	2	10	98	100	2	0	0	2
status	0	0	0	0	0	0	0	87	0	0	0	0	0	0
min	0	0	0	0	0	0	0	9	0	0	0	0	0	0
numex														
text	327	332	279	0	48	0	5	0	85	84	0	2	15	16
type	327	332	327	0	0	0	5	0	100	98	0	2	0	2
status	0	0	0	0	0	0	0	12	0	0	0	0	0	0
min	0	0	0	0	0	0	0	33	0	0	0	0	0	0
timex														
text	1480	1550	1405	0	72	3	73	1	95	91	0	5	5	10
type	1481	1554	1476	0	2	3	76	1	100	95	0	5	0	5
status	0	0	0	0	0	0	0	140	0	0	0	0	0	0
min	0	0	0	0	0	0	0	85	0	0	0	0	0	0
ALL SLOTS	11787	11808	11514	0	131	142	163	388	98	98	1	1	1	4
								P&R		2P&R		P&2R		
F-MEASURES								97.60		97.54		97.65		

Tabelle A.1: Punkteverteilung Annotator 1 (Mensch)

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	1850	1868	1807	0	17	26	44	35	98	97	1	2	1	5
person	882	879	877	0	2	3	0	5	99	100	0	0	0	1
location	1326	1304	1296	0	4	26	4	3	98	99	2	0	0	3
timex														
date	1205	1128	1115	0	3	87	10	57	93	99	7	1	0	8
time	192	159	151	0	8	33	0	28	79	95	17	0	5	21
numex														
money	217	200	200	0	0	17	0	10	92	100	8	0	0	8
percent	100	101	100	0	0	0	1	0	100	99	0	1	0	1
SECT SCORES														
slug	160	159	158	0	0	2	1	2	99	99	1	1	0	2
date	110	100	100	0	0	10	0	0	91	100	9	0	0	9
nwords	90	84	84	0	0	6	0	0	93	100	7	0	0	7
preamble	1236	1220	1214	0	2	20	4	22	98	100	2	0	0	2
text	9540	9308	9108	0	90	342	110	705	95	98	4	1	1	6
trailer	408	404	397	0	7	4	0	0	97	98	1	0	2	3
OBJ SCORES														
enamex	4058	4048	4003	0	0	55	45	33	99	99	1	1	0	2
numex	317	301	300	0	0	17	1	10	95	100	5	0	0	6
timex	1397	1287	1277	0	0	120	10	84	91	99	9	1	0	9
SLOT SCORES														
enamex														
text	4058	4048	3993	0	10	55	45	33	98	99	1	1	0	3
type	4058	4051	3980	0	23	55	48	43	98	98	1	1	1	3
status	0	0	0	0	0	0	0	187	0	0	0	0	0	0
min	0	0	0	0	0	0	0	14	0	0	0	0	0	0
numex														
text	317	301	297	0	3	17	1	10	94	99	5	0	1	7
type	317	301	300	0	0	17	1	10	95	100	5	0	0	6
status	0	0	0	0	0	0	0	18	0	0	0	0	0	0
min	0	0	0	0	0	0	0	2	0	0	0	0	0	0
timex														
text	1397	1287	1225	0	52	120	10	84	88	95	9	1	4	13
type	1397	1287	1266	0	11	120	10	85	91	98	9	1	1	10
status	0	0	0	0	0	0	0	186	0	0	0	0	0	0
min	0	0	0	0	0	0	0	57	0	0	0	0	0	0
ALL SLOTS	11544	11275	11061	0	99	384	115	729	96	98	3	1	1	5
								P&R		2P&R			P&2R	
F-MEASURES								96.95		97.64			96.26	

Tabelle A.2: Punkteverteilung Annotator 2 (Mensch)

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	1862	1830	1567	0	111	184	152	23	84	86	10	8	7	22
person	885	920	856	0	19	10	45	2	97	93	1	5	2	8
location	1310	1306	1212	0	38	60	56	19	93	93	5	4	3	11
timex														
date	1199	1072	1051	0	3	145	18	63	88	98	12	2	0	14
time	190	157	151	0	4	35	2	30	79	96	18	1	3	21
numex														
money	215	195	184	0	0	31	11	12	86	94	14	6	0	19
percent	100	98	97	0	0	3	1	0	97	99	3	1	0	4
SECT SCORES														
slug	158	140	128	0	2	28	10	3	81	91	18	7	2	24
date	110	110	110	0	0	0	0	0	100	100	0	0	0	0
nwords	90	90	90	0	0	0	0	0	100	100	0	0	0	0
preamble	1240	1198	1141	0	15	84	42	9	92	95	7	4	1	11
text	9516	9208	8379	0	313	824	516	536	88	91	9	6	4	16
trailer	408	410	407	0	1	0	2	0	100	99	0	0	0	1
OBJ SCORES														
enamex	4057	4056	3803	0	0	254	253	34	94	94	6	6	0	12
numex	315	293	281	0	0	34	12	12	89	96	11	4	0	14
timex	1389	1229	1209	0	0	180	20	92	87	98	13	2	0	14
SLOT SCORES														
enamex														
text	4057	4056	3716	0	87	254	253	34	92	92	6	6	2	14
type	4057	4056	3635	0	168	254	253	44	90	90	6	6	4	16
status	0	0	0	0	0	0	0	87	0	0	0	0	0	0
min	0	0	0	0	0	0	0	9	0	0	0	0	0	0
numex														
text	315	293	266	0	15	34	12	12	84	91	11	4	5	19
type	315	293	281	0	0	34	12	12	89	96	11	4	0	14
status	0	0	0	0	0	0	0	12	0	0	0	0	0	0
min	0	0	0	0	0	0	0	2	0	0	0	0	0	0
timex														
text	1389	1229	1155	0	54	180	20	92	83	94	13	2	4	18
type	1389	1229	1202	0	7	180	20	93	87	98	13	2	1	15
status	0	0	0	0	0	0	0	103	0	0	0	0	0	0
min	0	0	0	0	0	0	0	48	0	0	0	0	0	0
ALL SLOTS	11522	11156	10255	0	331	936	570	548	89	92	8	5	3	15
								P&R		2P&R			P&2R	
F-MEASURES								90.44		91.32			89.57	

Tabelle A.3: Punkteverteilung IdentiFinder

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	1841	1411	1208	0	83	550	120	44	66	86	30	9	6	38
person	884	867	777	0	18	89	72	3	88	90	10	8	2	19
location	1309	1293	1041	0	42	226	210	20	80	81	17	16	4	31
timex														
date	1201	1112	1032	0	8	161	72	61	86	93	13	6	1	19
time	193	167	161	0	4	28	2	27	83	96	15	1	2	17
numex														
money	218	184	177	0	0	41	7	9	81	96	19	4	0	21
percent	100	102	99	0	0	1	3	0	99	97	1	3	0	4
SECT SCORES														
slug	158	122	94	0	2	62	26	3	59	77	39	21	2	49
date	110	110	110	0	0	0	0	0	100	100	0	0	0	0
nwords	90	90	90	0	0	0	0	0	100	100	0	0	0	0
preamble	1240	1200	1034	0	42	164	124	9	83	86	13	10	4	24
text	9486	8342	7178	0	342	1966	822	566	76	86	21	10	5	30
trailer	408	408	407	0	1	0	0	0	100	100	0	0	0	0
OBJ SCORES														
enamex	4034	3571	3169	0	0	865	402	57	79	89	21	11	0	29
numex	318	286	276	0	0	42	10	9	87	97	13	3	0	16
timex	1394	1279	1205	0	0	189	74	87	86	94	14	6	0	18
SLOT SCORES														
enamex														
text	4034	3571	3020	0	149	865	402	57	75	85	21	11	5	32
type	4034	3571	3026	0	143	865	402	67	75	85	21	11	5	32
status	0	0	0	0	0	0	0	87	0	0	0	0	0	0
min	0	0	0	0	0	0	0	9	0	0	0	0	0	0
numex														
text	318	286	257	0	19	42	10	9	81	90	13	3	7	22
type	318	286	276	0	0	42	10	9	87	97	13	3	0	16
status	0	0	0	0	0	0	0	12	0	0	0	0	0	0
min	0	0	0	0	0	0	0	2	0	0	0	0	0	0
timex														
text	1394	1279	1141	0	64	189	74	87	82	89	14	6	5	22
type	1394	1279	1193	0	12	189	74	88	86	93	14	6	1	19
status	0	0	0	0	0	0	0	103	0	0	0	0	0	0
min	0	0	0	0	0	0	0	48	0	0	0	0	0	0
ALL SLOTS	11492	10272	8913	0	387	2192	972	578	78	87	19	9	4	28
								P&R		2P&R		P&2R		
F-MEASURES								81.91		84.76		79.24		

Tabelle A.4: Punkteverteilung BSEE von FACILE / CONCERTO

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	1852	1829	1620	0	22	210	187	33	87	89	11	10	1	21
person	885	858	835	0	10	40	13	2	94	97	5	2	1	7
location	1308	1291	1222	0	19	67	50	21	93	95	5	4	2	10
timex														
date	1198	1080	1067	0	4	127	9	64	89	99	11	1	0	12
time	191	159	155	0	3	33	1	29	81	97	17	1	2	19
numex														
money	215	213	201	0	0	14	12	12	93	94	7	6	0	11
percent	100	101	100	0	0	0	1	0	100	99	0	1	0	1
SECT SCORES														
slug	160	174	139	0	1	20	34	1	87	80	13	20	1	28
date	110	110	110	0	0	0	0	0	100	100	0	0	0	0
nwords	90	90	90	0	0	0	0	0	100	100	0	0	0	0
preamble	1242	1252	1183	0	9	50	60	7	95	94	4	5	1	9
text	9488	9026	8404	0	172	912	450	564	89	93	10	5	2	15
trailer	408	410	407	0	1	0	2	0	100	99	0	0	0	1
OBJ SCORES														
enamex	4045	3978	3728	0	0	317	250	46	92	94	8	6	0	13
numex	315	314	301	0	0	14	13	12	96	96	4	4	0	8
timex	1389	1239	1229	0	0	160	10	92	88	99	12	1	0	12
SLOT SCORES														
enamex														
text	4045	3978	3650	0	78	317	250	46	90	92	8	6	2	15
type	4045	3978	3677	0	51	317	250	56	91	92	8	6	1	14
status	0	0	0	0	0	0	0	87	0	0	0	0	0	0
min	0	0	0	0	0	0	0	9	0	0	0	0	0	0
numex														
text	315	314	291	0	10	14	13	12	92	93	4	4	3	11
type	315	314	301	0	0	14	13	12	96	96	4	4	0	8
status	0	0	0	0	0	0	0	12	0	0	0	0	0	0
min	0	0	0	0	0	0	0	2	0	0	0	0	0	0
timex														
text	1389	1239	1192	0	37	160	10	92	86	96	12	1	3	15
type	1389	1239	1222	0	7	160	10	93	88	99	12	1	1	13
status	0	0	0	0	0	0	0	103	0	0	0	0	0	0
min	0	0	0	0	0	0	0	48	0	0	0	0	0	0
ALL SLOTS	11498	11062	10333	0	183	982	546	572	90	93	9	5	2	14
								P&R		2P&R		P&2R		
F-MEASURES								91.60		92.68		90.55		

Tabelle A.5: Punkteverteilung NetOwl (System 1)

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	1852	1685	1503	0	23	326	159	33	81	89	18	9	2	25
person	885	817	806	0	4	75	7	2	91	99	8	1	0	10
location	1308	1252	1190	0	16	102	46	21	91	95	8	4	1	12
timex														
date	1196	507	502	0	0	694	5	66	42	99	58	1	0	58
time	187	0	0	0	0	187	0	33	0	0	100	0	0	100
numex														
money	215	200	188	0	0	27	12	12	87	94	13	6	0	17
percent	100	101	100	0	0	0	1	0	100	99	0	1	0	1
SECT SCORES														
slug	160	162	130	0	2	28	30	1	81	80	18	19	2	32
date	110	110	110	0	0	0	0	0	100	100	0	0	0	0
nwords	90	90	90	0	0	0	0	0	100	100	0	0	0	0
preamble	1242	1010	945	0	9	288	56	7	76	94	23	6	1	27
text	9476	7740	7231	0	139	2106	370	576	76	93	22	5	2	27
trailer	408	12	7	0	1	400	4	0	2	58	98	33	13	98
OBJ SCORES														
enamex	4045	3754	3542	0	0	503	212	46	88	94	12	6	0	17
numex	315	301	288	0	0	27	13	12	91	96	9	4	0	12
timex	1383	507	502	0	0	881	5	98	36	99	64	1	0	64
SLOT SCORES														
enamex														
text	4045	3754	3461	0	81	503	212	46	86	92	12	6	2	19
type	4045	3754	3499	0	43	503	212	56	87	93	12	6	1	18
status	0	0	0	0	0	0	0	87	0	0	0	0	0	0
min	0	0	0	0	0	0	0	9	0	0	0	0	0	0
numex														
text	315	301	278	0	10	27	13	12	88	92	9	4	3	15
type	315	301	288	0	0	27	13	12	91	96	9	4	0	12
status	0	0	0	0	0	0	0	12	0	0	0	0	0	0
min	0	0	0	0	0	0	0	2	0	0	0	0	0	0
timex														
text	1383	507	485	0	17	881	5	98	35	96	64	1	3	65
type	1383	507	502	0	0	881	5	99	36	99	64	1	0	64
status	0	0	0	0	0	0	0	103	0	0	0	0	0	0
min	0	0	0	0	0	0	0	48	0	0	0	0	0	0
ALL SLOTS	11486	9124	8513	0	151	2822	460	584	74	93	25	5	2	29
								P&R		2P&R			P&2R	
F-MEASURES								82.61		88.71			77.30	

Tabelle A.6: Punkteverteilung NetOwl (System 2)

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	1863	1802	1330	0	121	412	351	22	71	74	22	19	8	40
person	885	887	737	0	71	77	79	2	83	83	9	9	9	24
location	1312	1229	1005	0	56	251	168	17	77	82	19	14	5	32
timex														
date	1199	1007	928	0	3	268	76	63	77	92	22	8	0	27
time	191	155	144	0	4	43	7	29	75	93	23	5	3	27
numex														
money	216	262	204	0	0	12	58	11	94	78	6	22	0	26
percent	100	112	97	0	0	3	15	0	97	87	3	13	0	16
SECT SCORES														
slug	158	0	0	0	0	158	0	3	0	0	100	0	0	100
date	110	110	110	0	0	0	0	0	100	100	0	0	0	0
nwords	90	90	90	0	0	0	0	0	100	100	0	0	0	0
preamble	1240	1026	785	0	73	382	168	9	63	77	31	16	9	44
text	9526	9280	7341	0	605	1580	1334	526	77	79	17	14	8	32
trailer	408	402	396	0	0	12	6	0	97	99	3	1	0	4
OBJ SCORES														
enamex	4060	3918	3320	0	0	740	598	31	82	85	18	15	0	29
numex	316	374	301	0	0	15	73	11	95	80	5	20	0	23
timex	1390	1162	1079	0	0	311	83	91	78	93	22	7	0	27
SLOT SCORES														
enamex														
text	4060	3918	3043	0	277	740	598	31	75	78	18	15	8	35
type	4060	3918	3072	0	248	740	598	41	76	78	18	15	7	34
status	0	0	0	0	0	0	0	87	0	0	0	0	0	0
min	0	0	0	0	0	0	0	9	0	0	0	0	0	0
numex														
text	316	374	275	0	26	15	73	11	87	74	5	20	9	29
type	316	374	301	0	0	15	73	11	95	80	5	20	0	23
status	0	0	0	0	0	0	0	12	0	0	0	0	0	0
min	0	0	0	0	0	0	0	2	0	0	0	0	0	0
timex														
text	1390	1162	959	0	120	311	83	91	69	83	22	7	11	35
type	1390	1162	1072	0	7	311	83	92	77	92	22	7	1	27
status	0	0	0	0	0	0	0	103	0	0	0	0	0	0
min	0	0	0	0	0	0	0	48	0	0	0	0	0	0
ALL SLOTS	11532	10908	8722	0	678	2132	1508	538	76	80	18	14	7	33
								P&R		2P&R		P&2R		
F-MEASURES								77.74		79.06		76.46		

Tabelle A.7: Punkteverteilung Kent Ridge Digital Labs System

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	1854	1784	1692	0	10	152	82	31	91	95	8	5	1	13
person	883	872	842	0	24	17	6	4	95	97	2	1	3	5
location	1308	1326	1239	0	14	55	73	21	95	93	4	6	1	10
timex														
date	1201	1079	1063	0	3	135	13	61	89	99	11	1	0	12
time	191	156	151	0	4	36	1	29	79	97	19	1	3	21
numex														
money	216	210	204	0	0	12	6	11	94	97	6	3	0	8
percent	100	103	100	0	0	0	3	0	100	97	0	3	0	3
SECT SCORES														
slug	158	148	141	0	1	16	6	3	89	95	10	4	1	14
date	110	110	110	0	0	0	0	0	100	100	0	0	0	0
nwords	90	90	90	0	0	0	0	0	100	100	0	0	0	0
preamble	1236	1204	1176	0	6	54	22	13	95	98	4	2	1	7
text	9504	9100	8613	0	147	744	340	548	91	95	8	4	2	13
trailer	408	408	407	0	1	0	0	0	100	100	0	0	0	0
OBJ SCORES														
enamex	4045	3982	3821	0	0	224	161	46	94	96	6	4	0	9
numex	316	313	304	0	0	12	9	11	96	97	4	3	0	6
timex	1392	1235	1221	0	0	171	14	89	88	99	12	1	0	13
SLOT SCORES														
enamex														
text	4045	3982	3776	0	45	224	161	46	93	95	6	4	1	10
type	4045	3982	3773	0	48	224	161	56	93	95	6	4	1	10
status	0	0	0	0	0	0	0	87	0	0	0	0	0	0
min	0	0	0	0	0	0	0	9	0	0	0	0	0	0
numex														
text	316	313	289	0	15	12	9	11	91	92	4	3	5	11
type	316	313	304	0	0	12	9	11	96	97	4	3	0	6
status	0	0	0	0	0	0	0	12	0	0	0	0	0	0
min	0	0	0	0	0	0	0	2	0	0	0	0	0	0
timex														
text	1392	1235	1181	0	40	171	14	89	85	96	12	1	3	16
type	1392	1235	1214	0	7	171	14	90	87	98	12	1	1	14
status	0	0	0	0	0	0	0	103	0	0	0	0	0	0
min	0	0	0	0	0	0	0	48	0	0	0	0	0	0
ALL SLOTS	11506	11060	10537	0	155	814	368	564	92	95	7	3	1	11
								P&R		2P&R		P&2R		
F-MEASURES								93.39		94.51		92.29		

Tabelle A.8: Punkteverteilung LT TTT

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	1847	1946	1556	0	46	245	344	38	84	80	13	18	3	29
person	886	886	789	0	39	58	58	1	89	89	7	7	5	16
location	1307	1301	1061	0	104	142	136	22	81	82	11	10	9	26
timex														
date	1196	1056	1039	0	2	155	15	66	87	98	13	1	0	14
time	188	145	144	0	1	43	0	32	77	99	23	0	1	23
numex														
money	215	213	189	0	0	26	24	12	88	89	12	11	0	21
percent	100	106	100	0	0	0	6	0	100	94	0	6	0	6
SECT SCORES														
slug	160	204	121	0	21	18	62	1	76	59	11	30	15	45
date	110	110	110	0	0	0	0	0	100	100	0	0	0	0
nwords	90	90	90	0	0	0	0	0	100	100	0	0	0	0
preamble	1242	1340	1082	0	66	94	192	7	87	81	8	14	6	25
text	9468	9150	7909	0	333	1226	908	584	84	86	13	10	4	24
trailer	408	412	407	0	1	0	4	0	100	99	0	1	0	1
OBJ SCORES														
enamex	4040	4133	3595	0	0	445	538	51	89	87	11	13	0	21
numex	315	319	289	0	0	26	30	12	92	91	8	9	0	16
timex	1384	1201	1186	0	0	198	15	97	86	99	14	1	0	15
SLOT SCORES														
enamex														
text	4040	4133	3445	0	150	445	538	51	85	83	11	13	4	25
type	4040	4133	3406	0	189	445	538	61	84	82	11	13	5	26
status	0	0	0	0	0	0	0	87	0	0	0	0	0	0
min	0	0	0	0	0	0	0	9	0	0	0	0	0	0
numex														
text	315	319	267	0	22	26	30	12	85	84	8	9	8	23
type	315	319	289	0	0	26	30	12	92	91	8	9	0	16
status	0	0	0	0	0	0	0	12	0	0	0	0	0	0
min	0	0	0	0	0	0	0	2	0	0	0	0	0	0
timex														
text	1384	1201	1129	0	57	198	15	97	82	94	14	1	5	19
type	1384	1201	1183	0	3	198	15	98	85	99	14	1	0	15
status	0	0	0	0	0	0	0	103	0	0	0	0	0	0
min	0	0	0	0	0	0	0	48	0	0	0	0	0	0
ALL SLOTS	11478	11306	9719	0	421	1338	1166	592	85	86	12	10	4	23
								P&R		2P&R			P&2R	
F-MEASURES								85.31		85.70			84.93	

Tabelle A.9: Punkteverteilung für das System von The MITRE Corporation

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	1830	1118	766	0	249	815	103	55	42	69	45	9	25	60
person	883	1100	681	0	16	186	403	4	77	62	21	37	2	47
location	1311	1513	974	0	124	213	415	18	74	64	16	27	11	44
timex														
date	1198	1038	947	0	2	249	89	64	79	91	21	9	0	26
time	192	160	153	0	2	37	5	28	80	96	19	3	1	22
numex														
money	216	175	168	0	0	48	7	11	78	96	22	4	0	25
percent	100	102	97	0	0	3	5	0	97	95	3	5	0	8
SECT SCORES														
slug	158	26	12	0	0	146	14	3	8	46	92	54	0	93
date	110	110	110	0	0	0	0	0	100	100	0	0	0	0
nwords	90	90	90	0	0	0	0	0	100	100	0	0	0	0
preamble	1236	1120	763	0	75	398	282	13	62	68	32	25	9	50
text	9458	8656	6238	0	664	2556	1754	594	66	72	27	20	10	44
trailer	408	410	406	0	0	2	4	0	100	99	0	1	0	1
OBJ SCORES														
enamex	4024	3731	2810	0	0	1214	921	67	70	75	30	25	0	43
numex	316	277	265	0	0	51	12	11	84	96	16	4	0	19
timex	1390	1198	1104	0	0	286	94	91	79	92	21	8	0	26
SLOT SCORES														
enamex														
text	4024	3731	2564	0	246	1214	921	67	64	69	30	25	9	48
type	4024	3731	2421	0	389	1214	921	77	60	65	30	25	14	51
status	0	0	0	0	0	0	0	87	0	0	0	0	0	0
min	0	0	0	0	0	0	0	9	0	0	0	0	0	0
numex														
text	316	277	244	0	21	51	12	11	77	88	16	4	8	26
type	316	277	265	0	0	51	12	11	84	96	16	4	0	19
status	0	0	0	0	0	0	0	12	0	0	0	0	0	0
min	0	0	0	0	0	0	0	2	0	0	0	0	0	0
timex														
text	1390	1198	1025	0	79	286	94	91	74	86	21	8	7	31
type	1390	1198	1100	0	4	286	94	92	79	92	21	8	0	26
status	0	0	0	0	0	0	0	103	0	0	0	0	0	0
min	0	0	0	0	0	0	0	48	0	0	0	0	0	0
ALL SLOTS	11460	10412	7619	0	739	3102	2054	610	66	73	27	20	9	44
								P&R		2P&R		P&2R		
F-MEASURES								69.67		71.73		67.72		

Tabelle A.10: Punkteverteilung für das System von National Taiwan University

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	1853	1680	1489	0	67	297	124	32	80	89	16	7	4	25
person	885	875	815	0	18	52	42	2	92	93	6	5	2	12
location	1308	1223	1136	0	36	136	51	21	87	93	10	4	3	16
timex														
date	1199	1075	1046	0	3	150	26	63	87	97	13	2	0	15
time	191	153	149	0	4	38	0	29	78	97	20	0	3	22
numex														
money	215	192	177	0	0	38	15	12	82	92	18	8	0	23
percent	100	101	96	0	0	4	5	0	96	95	4	5	0	9
SECT SCORES														
slug	158	84	76	0	0	82	8	3	48	90	52	10	0	54
date	110	110	110	0	0	0	0	0	100	100	0	0	0	0
nwords	90	90	90	0	0	0	0	0	100	100	0	0	0	0
preamble	1240	1084	1056	0	10	174	18	9	85	97	14	2	1	16
text	9496	8822	8073	0	249	1174	500	556	85	92	12	6	3	19
trailer	408	408	407	0	1	0	0	0	100	100	0	0	0	0
OBJ SCORES														
enamex	4046	3778	3561	0	0	485	217	45	88	94	12	6	0	16
numex	315	293	273	0	0	42	20	12	87	93	13	7	0	19
timex	1390	1228	1202	0	0	188	26	91	86	98	14	2	0	15
SLOT SCORES														
enamex														
text	4046	3778	3498	0	63	485	217	45	86	93	12	6	2	18
type	4046	3778	3440	0	121	485	217	55	85	91	12	6	3	19
status	0	0	0	0	0	0	0	87	0	0	0	0	0	0
min	0	0	0	0	0	0	0	9	0	0	0	0	0	0
numex														
text	315	293	250	0	23	42	20	12	79	85	13	7	8	25
type	315	293	273	0	0	42	20	12	87	93	13	7	0	19
status	0	0	0	0	0	0	0	12	0	0	0	0	0	0
min	0	0	0	0	0	0	0	2	0	0	0	0	0	0
timex														
text	1390	1228	1156	0	46	188	26	91	83	94	14	2	4	18
type	1390	1228	1195	0	7	188	26	92	86	97	14	2	1	16
status	0	0	0	0	0	0	0	103	0	0	0	0	0	0
min	0	0	0	0	0	0	0	48	0	0	0	0	0	0
ALL SLOTS	11502	10598	9812	0	260	1430	526	568	85	93	12	5	3	18
								P&R		2P&R		P&2R		
F-MEASURES								88.80		91.03		86.67		

Tabelle A.11: Punkteverteilung MENE

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	1839	1147	1061	0	46	732	40	46	58	93	40	3	4	44
person	884	775	725	0	3	156	47	3	82	94	18	6	0	22
location	1310	1311	1183	0	20	107	108	19	90	90	8	8	2	17
timex														
date	1210	1124	1075	0	3	132	46	52	89	96	11	4	0	14
time	190	158	154	0	3	33	1	30	81	97	17	1	2	19
numex														
money	215	206	200	0	0	15	6	12	93	97	7	3	0	10
percent	100	105	100	0	0	0	5	0	100	95	0	5	0	5
SECT SCORES														
slug	158	120	112	0	0	46	8	3	71	93	29	7	0	33
date	110	110	110	0	0	0	0	0	100	100	0	0	0	0
nwords	90	90	90	0	0	0	0	0	100	100	0	0	0	0
preamble	1240	1044	932	0	10	298	102	9	75	89	24	10	1	31
text	9490	7882	7239	0	249	2002	394	562	76	92	21	5	3	27
trailer	408	406	404	0	0	4	2	0	99	100	1	0	0	1
OBJ SCORES														
enamex	4033	3233	3038	0	0	995	195	58	75	94	25	6	0	28
numex	315	311	300	0	0	15	11	12	95	96	5	4	0	8
timex	1400	1282	1235	0	0	165	47	81	88	96	12	4	0	15
SLOT SCORES														
enamex														
text	4033	3233	2926	0	112	995	195	58	73	91	25	6	4	31
type	4033	3233	2969	0	69	995	195	68	74	92	25	6	2	30
status	0	0	0	0	0	0	0	87	0	0	0	0	0	0
min	0	0	0	0	0	0	0	9	0	0	0	0	0	0
numex														
text	315	311	280	0	20	15	11	12	89	90	5	4	7	14
type	315	311	300	0	0	15	11	12	95	96	5	4	0	8
status	0	0	0	0	0	0	0	12	0	0	0	0	0	0
min	0	0	0	0	0	0	0	2	0	0	0	0	0	0
timex														
text	1400	1282	1183	0	52	165	47	81	85	92	12	4	4	18
type	1400	1282	1229	0	6	165	47	82	88	96	12	4	0	15
status	0	0	0	0	0	0	0	103	0	0	0	0	0	0
min	0	0	0	0	0	0	0	48	0	0	0	0	0	0
ALL SLOTS	11496	9652	8887	0	259	2350	506	574	77	92	20	5	3	26
								P&R		2P&R			P&2R	
F-MEASURES								84.05		88.69			79.87	

Tabelle A.12: Punkteverteilung Oki System (Englisch)

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	1846	1751	1165	0	220	461	366	39	63	67	25	21	16	47
person	883	955	705	0	91	87	159	4	80	74	10	17	11	32
location	1318	1334	976	0	181	161	177	11	74	73	12	13	16	35
timex														
date	1200	1098	1010	0	2	188	86	62	84	92	16	8	0	21
time	192	178	160	0	4	28	14	28	83	90	15	8	2	22
numex														
money	215	221	198	0	0	17	23	12	92	90	8	10	0	17
percent	100	3	3	0	0	97	0	0	3	100	97	0	0	97
SECT SCORES														
slug	158	120	88	0	10	60	22	3	56	73	38	18	10	51
date	110	110	110	0	0	0	0	0	100	100	0	0	0	0
nwords	90	90	90	0	0	0	0	0	100	100	0	0	0	0
preamble	1234	1046	891	0	37	306	118	15	72	85	25	11	4	34
text	9508	9314	7054	0	750	1704	1510	544	74	76	18	16	10	36
trailer	408	400	399	0	1	8	0	0	98	100	2	0	0	2
OBJ SCORES														
enamex	4047	4040	3338	0	0	709	702	44	82	83	18	17	0	30
numex	315	224	201	0	0	114	23	12	64	90	36	10	0	41
timex	1392	1276	1176	0	0	216	100	89	84	92	16	8	0	21
SLOT SCORES														
enamex														
text	4047	4040	3181	0	157	709	702	44	79	79	18	17	5	33
type	4047	4040	2846	0	492	709	702	54	70	70	18	17	15	40
status	0	0	0	0	0	0	0	87	0	0	0	0	0	0
min	0	0	0	0	0	0	0	9	0	0	0	0	0	0
numex														
text	315	224	172	0	29	114	23	12	55	77	36	10	14	49
type	315	224	201	0	0	114	23	12	64	90	36	10	0	41
status	0	0	0	0	0	0	0	12	0	0	0	0	0	0
min	0	0	0	0	0	0	0	2	0	0	0	0	0	0
timex														
text	1392	1276	1062	0	114	216	100	89	76	83	16	8	10	29
type	1392	1276	1170	0	6	216	100	90	84	92	16	8	1	22
status	0	0	0	0	0	0	0	103	0	0	0	0	0	0
min	0	0	0	0	0	0	0	48	0	0	0	0	0	0
ALL SLOTS	11508	11080	8632	0	798	2078	1650	562	75	78	18	15	8	34
								P&R		2P&R			P&2R	
F-MEASURES								76.43		77.31			75.57	

Tabelle A.13: Punkteverteilung LOLITA

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	1860	1822	1590	0	72	198	160	25	85	87	11	9	4	21
person	883	969	803	0	21	59	145	4	91	83	7	15	3	22
location	1307	1245	1054	0	71	182	120	22	81	85	14	10	6	26
timex														
date	1198	1106	1037	0	4	157	65	64	87	94	13	6	0	18
time	193	167	159	0	6	28	2	27	82	95	15	1	4	18
numex														
money	215	215	198	0	0	17	17	12	92	92	8	8	0	15
percent	100	105	98	0	0	2	7	0	98	93	2	7	0	8
SECT SCORES														
slug	158	136	115	0	5	38	16	3	73	85	24	12	4	34
date	110	110	110	0	0	0	0	0	100	100	0	0	0	0
nwords	90	92	90	0	0	0	2	0	100	98	0	2	0	2
preamble	1238	1176	1064	0	30	144	82	11	86	90	12	7	3	19
text	9508	9334	8048	0	356	1104	930	544	85	86	12	10	4	23
trailer	408	410	406	0	2	0	2	0	100	99	0	0	0	1
OBJ SCORES														
enamex	4050	4036	3611	0	0	439	425	41	89	89	11	11	0	19
numex	315	320	296	0	0	19	24	12	94	93	6	8	0	13
timex	1391	1273	1206	0	0	185	67	90	87	95	13	5	0	17
SLOT SCORES														
enamex														
text	4050	4036	3513	0	98	439	425	41	87	87	11	11	3	21
type	4050	4036	3447	0	164	439	425	51	85	85	11	11	5	23
status	0	0	0	0	0	0	0	87	0	0	0	0	0	0
min	0	0	0	0	0	0	0	9	0	0	0	0	0	0
numex														
text	315	320	261	0	35	19	24	12	83	82	6	8	12	23
type	315	320	296	0	0	19	24	12	94	93	6	8	0	13
status	0	0	0	0	0	0	0	12	0	0	0	0	0	0
min	0	0	0	0	0	0	0	2	0	0	0	0	0	0
timex														
text	1391	1273	1120	0	86	185	67	90	81	88	13	5	7	23
type	1391	1273	1196	0	10	185	67	91	86	94	13	5	1	18
status	0	0	0	0	0	0	0	103	0	0	0	0	0	0
min	0	0	0	0	0	0	0	48	0	0	0	0	0	0
ALL SLOTS	11512	11258	9833	0	393	1286	1032	558	85	87	11	9	4	22
								P&R		2P&R		P&2R		
F-MEASURES								86.37		86.95		85.79		

Tabelle A.14: Punkteverteilung PIE mit Kollokations-Statistik-Erweiterung (System 1)

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	1842	1460	1305	0	53	484	102	43	71	89	26	7	4	33
person	883	881	765	0	14	104	102	4	87	87	12	12	2	22
location	1307	1185	1044	0	25	238	116	22	80	88	18	10	2	27
timex														
date	1198	1117	1038	0	4	156	75	64	87	93	13	7	0	18
time	193	166	158	0	6	29	2	27	82	95	15	1	4	19
numex														
money	215	215	198	0	0	17	17	12	92	92	8	8	0	15
percent	100	105	98	0	0	2	7	0	98	93	2	7	0	8
SECT SCORES														
slug	158	120	105	0	1	52	14	3	66	88	33	12	1	39
date	110	110	110	0	0	0	0	0	100	100	0	0	0	0
nwords	90	92	90	0	0	0	2	0	100	98	0	2	0	2
preamble	1238	1078	1001	0	17	220	60	11	81	93	18	6	2	23
text	9472	8448	7384	0	300	1788	764	580	78	87	19	9	4	28
trailer	408	410	406	0	2	0	2	0	100	99	0	0	0	1
OBJ SCORES														
enamex	4032	3526	3206	0	0	826	320	59	80	91	20	9	0	26
numex	315	320	296	0	0	19	24	12	94	93	6	8	0	13
timex	1391	1283	1206	0	0	185	77	90	87	94	13	6	0	18
SLOT SCORES														
enamex														
text	4032	3526	3110	0	96	826	320	59	77	88	20	9	3	29
type	4032	3526	3114	0	92	826	320	69	77	88	20	9	3	28
status	0	0	0	0	0	0	0	87	0	0	0	0	0	0
min	0	0	0	0	0	0	0	9	0	0	0	0	0	0
numex														
text	315	320	261	0	35	19	24	12	83	82	6	8	12	23
type	315	320	296	0	0	19	24	12	94	93	6	8	0	13
status	0	0	0	0	0	0	0	12	0	0	0	0	0	0
min	0	0	0	0	0	0	0	2	0	0	0	0	0	0
timex														
text	1391	1283	1119	0	87	185	77	90	80	87	13	6	7	24
type	1391	1283	1196	0	10	185	77	91	86	93	13	6	1	19
status	0	0	0	0	0	0	0	103	0	0	0	0	0	0
min	0	0	0	0	0	0	0	48	0	0	0	0	0	0
ALL SLOTS	11476	10258	9096	0	320	2060	842	594	79	89	18	8	3	26
								P&R		2P&R			P&2R	
F-MEASURES								83.70		86.62			80.98	

Tabelle A.15: Punkteverteilung PIE mit Kollokations-Statistik-Erweiterung (System 2)

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	1843	1682	1453	0	93	297	136	42	79	86	16	8	6	27
person	885	843	759	0	31	95	53	2	86	90	11	6	4	19
location	1309	1206	1075	0	24	210	107	20	82	89	16	9	2	24
timex														
date	1201	1120	1057	0	3	141	60	61	88	94	12	5	0	16
time	190	162	154	0	6	30	2	30	81	95	16	1	4	20
numex														
money	215	198	183	0	0	32	15	12	85	92	15	8	0	20
percent	100	105	100	0	0	0	5	0	100	95	0	5	0	5
SECT SCORES														
slug	160	146	127	0	3	30	16	1	79	87	19	11	2	28
date	110	110	110	0	0	0	0	0	100	100	0	0	0	0
nwords	90	90	90	0	0	0	0	0	100	100	0	0	0	0
preamble	1238	1196	1086	0	26	126	84	11	88	91	10	7	2	18
text	9480	8682	7672	0	354	1454	656	572	81	88	15	8	4	24
trailer	408	408	407	0	1	0	0	0	100	100	0	0	0	0
OBJ SCORES														
enamex	4037	3731	3435	0	0	602	296	54	85	92	15	8	0	21
numex	315	303	283	0	0	32	20	12	90	93	10	7	0	16
timex	1391	1282	1220	0	0	171	62	90	88	95	12	5	0	16
SLOT SCORES														
enamex														
text	4037	3731	3335	0	100	602	296	54	83	89	15	8	3	23
type	4037	3731	3287	0	148	602	296	64	81	88	15	8	4	24
status	0	0	0	0	0	0	0	87	0	0	0	0	0	0
min	0	0	0	0	0	0	0	9	0	0	0	0	0	0
numex														
text	315	303	244	0	39	32	20	12	77	81	10	7	14	27
type	315	303	283	0	0	32	20	12	90	93	10	7	0	16
status	0	0	0	0	0	0	0	12	0	0	0	0	0	0
min	0	0	0	0	0	0	0	2	0	0	0	0	0	0
timex														
text	1391	1282	1132	0	88	171	62	90	81	88	12	5	7	22
type	1391	1282	1211	0	9	171	62	91	87	94	12	5	1	17
status	0	0	0	0	0	0	0	103	0	0	0	0	0	0
min	0	0	0	0	0	0	0	48	0	0	0	0	0	0
ALL SLOTS	11486	10632	9492	0	384	1610	756	584	83	89	14	7	4	22
								P&R		2P&R		P&2R		
F-MEASURES								85.83		87.87		83.89		

Tabelle A.16: Punkteverteilung LaSIE

A.2 Punkteverteilung NE-Task - Japanisch

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	637	579	480	0	3	154	96	0	75	83	24	17	1	35
person	139	91	67	0	1	71	23	0	48	74	51	25	1	59
location	910	737	639	0	4	267	94	0	70	87	29	13	1	36
timex														
date	558	562	534	0	0	24	28	0	96	95	4	5	0	9
time	120	119	114	0	0	6	5	0	95	96	5	4	0	9
numex														
money	71	66	64	0	0	7	2	0	90	97	10	3	0	12
percent	42	40	38	0	0	4	2	0	90	95	10	5	0	14
SECT SCORES														
DOC	56	44	38	0	2	16	4	0	68	86	29	9	5	37
DATE	304	304	304	0	0	0	0	0	100	100	0	0	0	0
DOCNO	0	0	0	0	0	0	0	0	0	0	0	0	0	0
HL	258	210	188	0	12	58	10	0	73	90	22	5	6	30
TEXT	4336	3830	3184	0	160	992	486	0	73	83	23	13	5	34
OBJ SCORES														
enamex	1686	1407	1194	0	0	492	213	0	71	85	29	15	0	37
numex	113	106	102	0	0	11	4	0	90	96	10	4	0	13
timex	678	681	648	0	0	30	33	0	96	95	4	5	0	9
SLOT SCORES														
enamex														
text	1686	1407	1073	0	121	492	213	0	64	76	29	15	10	43
type	1686	1407	1186	0	8	492	213	0	70	84	29	15	1	38
numex														
text	113	106	97	0	5	11	4	0	86	92	10	4	5	17
type	113	106	102	0	0	11	4	0	90	96	10	4	0	13
timex														
text	678	681	608	0	40	30	33	0	90	89	4	5	6	14
type	678	681	648	0	0	30	33	0	96	95	4	5	0	9
ALL SLOTS	4954	4388	3714	0	174	1066	500	0	75	85	22	11	4	32
									P&R		2P&R		P&2R	
F-MEASURES									79.51		82.51		76.72	

Tabelle A.17: Punkteverteilung Entscheidungs-Baum-System von Sekine (NYU)

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	637	591	478	0	7	152	106	0	75	81	24	18	1	36
person	139	97	89	0	1	49	7	0	64	92	35	7	1	39
location	910	747	679	0	1	230	67	0	75	91	25	9	0	31
timex														
date	558	536	524	0	0	34	12	0	94	98	6	2	0	8
time	120	116	115	0	0	5	1	0	96	99	4	1	0	5
numex														
money	71	70	69	0	0	2	1	0	97	99	3	1	0	4
percent	42	41	39	0	0	3	2	0	93	95	7	5	0	11
SECT SCORES														
DOC	56	44	40	0	0	16	4	0	71	91	29	9	0	33
DATE	304	304	304	0	0	0	0	0	100	100	0	0	0	0
DOCNO	0	0	0	0	0	0	0	0	0	0	0	0	0	0
HL	258	192	166	0	8	84	18	0	64	86	33	9	5	40
TEXT	4336	3856	3404	0	82	850	370	0	79	88	20	10	2	28
OBJ SCORES														
enamex	1686	1435	1255	0	0	431	180	0	74	87	26	13	0	33
numex	113	111	108	0	0	5	3	0	96	97	4	3	0	7
timex	678	652	639	0	0	39	13	0	94	98	6	2	0	8
SLOT SCORES														
enamex														
text	1686	1435	1196	0	59	431	180	0	71	83	26	13	5	36
type	1686	1435	1246	0	9	431	180	0	74	87	26	13	1	33
numex														
text	113	111	106	0	2	5	3	0	94	95	4	3	2	9
type	113	111	108	0	0	5	3	0	96	97	4	3	0	7
timex														
text	678	652	619	0	20	39	13	0	91	95	6	2	3	10
type	678	652	639	0	0	39	13	0	94	98	6	2	0	8
ALL SLOTS	4954	4396	3914	0	90	950	392	0	79	89	19	9	2	27
								P&R		2P&R			P&2R	
F-MEASURES								83.72		86.83			80.83	

Tabelle A.18: Punkteverteilung NTT System

	POS	ACT	COR	PAR	INC	MIS	SPU	NON	REC	PRE	UND	OVG	SUB	ERR
SUBTASK SCORES														
enamex														
organizatio	637	456	444	0	3	190	9	0	70	97	30	2	1	31
person	139	71	70	0	1	68	0	0	50	99	49	0	1	50
location	910	863	847	0	4	59	12	0	93	98	6	1	0	8
timex														
date	558	556	540	0	0	18	16	0	97	97	3	3	0	6
time	120	110	110	0	0	10	0	0	92	100	8	0	0	8
numex														
money	71	73	71	0	0	0	2	0	100	97	0	3	0	3
percent	42	44	42	0	0	0	2	0	100	95	0	5	0	5
SECT SCORES														
DOC	56	54	52	0	0	4	2	0	93	96	7	4	0	10
DATE	304	304	304	0	0	0	0	0	100	100	0	0	0	0
DOCNO	0	0	0	0	0	0	0	0	0	0	0	0	0	0
HL	258	210	207	0	3	48	0	0	80	99	19	0	1	20
TEXT	4336	3778	3647	0	51	638	80	0	84	97	15	2	1	17
OBJ SCORES														
enamex	1686	1390	1369	0	0	317	21	0	81	98	19	2	0	20
numex	113	117	113	0	0	0	4	0	100	97	0	3	0	3
timex	678	666	650	0	0	28	16	0	96	98	4	2	0	6
SLOT SCORES														
enamex														
text	1686	1390	1335	0	34	317	21	0	79	96	19	2	2	22
type	1686	1390	1361	0	8	317	21	0	81	98	19	2	1	20
numex														
text	113	117	111	0	2	0	4	0	98	95	0	3	2	5
type	113	117	113	0	0	0	4	0	100	97	0	3	0	3
timex														
text	678	666	640	0	10	28	16	0	94	96	4	2	2	8
type	678	666	650	0	0	28	16	0	96	98	4	2	0	6
ALL SLOTS	4954	4346	4210	0	54	690	82	0	85	97	14	2	1	16
								P&R		2P&R			P&2R	
F-MEASURES								90.54		94.23			87.12	

Tabelle A.19: Punkteverteilung Oki System (Japanisch)

Anhang B

Ranglisten

Rang	**Systemname**	**Entwickler**	**F-Wert (P&R)**
1	*Oki System as Used for MET-2*	*Oki Electric Industry Co., Ltd., Osaka, Japan*	*90.54*
2	*?*	*Nippon Telegraph and Telephone Corporation (NTT)*	*83.72*
3	*Japanese NE System Used for MET-2*	Computer Science Department, New York University, New York, NY/USA	79.51

Tabelle B.1: **Rangliste NE-Task Japanisch (MET-2)**. *Kursiv* ausgezeichnet sind diejenigen Systeme, die in vorliegender Arbeit nicht beschrieben werden. Wenn beim Systemname ein Fragezeichen steht, konnte kein Beschrieb über das System gefunden werden.

Rang	**Systemname**	**Entwickler**	**F-Wert (P&R)**
	Annotator 1 (Mensch)		97.60
	Annotator 2 (Mensch)		96.95
1	LT TTT	Language Technology Group, University of Edinburgh, UK	93.39
2	NetOwl (System 1)	IsoQuest Inc., Fairfax, VA/USA	91.60
3	IdentiFinder	BBN Technologies, Cambridge, MA/USA	90.44
4	MENE	Computer Science Department, New York University, New York, NY/USA	88.80
5	PIE, Erweiterung durch Kollokations-Statistik (System 1)	Department of Computer Science, University of Manitoba, Winnipeg, Manitoba/Canada	86.37
6	LaSIE	NLP group, University of Sheffield, UK	85.83
7	*?*	*The MITRE Corporation, Bedford, MA/USA und McLean, VA/USA*	*85.31*
8	Oki System as Used for MUC-7	Oki Electric Industry Co., Ltd., Osaka, Japan	84.05
9	PIE, Erweiterung durch Kollokations-Statistik (System 2)	Department of Computer Science, University of Manitoba, Winnipeg, Manitoba/Canada	83.70
10	NetOwl (System 2)	IsoQuest Inc., Fairfax, VA/USA	82.61
11	BSEE von FACILE/CONCERTO	UMIST, University of Manchester, UK	81.91
12	*The Kent Ridge Digital Labs System used for MUC-7*	*Kent Ridge Digital Labs (NUS), Singapore*	*77.74*
13	LOLITA	Laboratory for Natural Language Engineering, Department of Computer Science, University of Durham, UK	76.43
14	*?*	*National Taiwan University, Taipei, Taiwan*	*69.67*

Tabelle B.2: **Rangliste NE-Task Englisch (MUC-7)**. *Kursiv* ausgezeichnet sind diejenigen Systeme, die in vorliegender Arbeit nicht beschrieben werden. Wenn beim Systemname ein Fragezeichen steht, konnte kein Beschrieb über das System gefunden werden.

Anhang C

Abkürzungsliste

C.1 Verwendete Abkürzungen

Im Folgenden sind die wichtigsten der verwendeten Abkürzungen aufgeführt. In Klammer steht das Unterkapitel, in dem erklärt wird, was in dieser Arbeit unter dem abgekürzten Begriff verstanden wird. Ein kleines (s) hinter einer Abkürzung bedeutet, dass sie auch im Genitiv oder im Plural verwendet werden kann.

PER	Personennamen
ORG	Organisationsnamen
LOC	Ortsnamen (location)
NE (s)	Named Entity (2.4.1)
NER	Named Entity Recognition (2.4.1)
NE-Task	Named Entity Recognition Task (2.4.1)
MUC	Message Understanding Conference (2.4.3)
IE	Information Extraction (3.3)
TE	Template Element (Task) (3.3)
TR	Template Relation (Task) (3.3)
ST	Scenario Template (Task) (3.3)
CO	Coreference (Task) (3.3)
NP (s)	Nominalphrase (4.1.1)
PP (s)	Präpositionalphrase (4.1.1)
HMM (s)	Hidden Markov Modell (5.2.2)
ME	Maximale Entropie (5.2.3)
MET	Multilingual Entity (Task) (5.2.4)
NLP	Natural Language Processing

Literaturverzeichnis

[Ben97] Bennett, S. W.; Aone, C.; Lovell, C.: *Learning to Tag Multilingual Texts Through Observation.* In: Proceedings of the Second Conference on Empirical Methods in NLP. 1997.

[Bik98] Bikel, D.; Schwartz, R.; Weischedel, R.; Miller, S.: *Nymble: a High-Performance Learning Name-finder.* BBN Corporation, Cambridge (Massachusetts), 1998.
http://acl.ldc.upenn.edu/A/A97/A97-1029.pdf

[Bik99] Bikel, D.; Schwartz, R.; Weischedel, R.: *An Algorithm that Learns Whats in a Name.* BBN Systems & Technologies, Cambridge (Massachusetts), 1999.
http://www.cis.upenn.edu/~dbikel/algthatlearns.doc.pdf

[Bla98] Black, W. J.; Rinaldi, F.; Mowatt, D.: *FACILE: Description of the NE System Used for MUC-7.* UMIST Manchester (UK), 1998.
http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_proceedings/facile_muc7.pdf

[Bor98] Borthwick, A.; Sterling, J.; Agichtein, E.; Grishman R.: *NYU: Description of the MENE Named Entity System as Used in MUC-7.* New York University, 1998.
http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_proceedings/nyu_english_named_entity.pdf

[Bor99] Borthwick, A.: *A Maximum Entropy Approach to Named Entity Recognition.* New York University, 1999. (Diss.)
http://cs.nyu.edu/cs/projects/proteus/publication/papers/borthwick_thesis.ps

[Car01] Carstensen, K.-U.; Ebert, Ch.; Endriss, C.; Jekat, S.; Klabunde, R.; Langer, H. (Hgg.): *Computerlinguistik und Sprachtechnologie. Eine Einführung.* Spektrum, Akademischer Verlag, Heidelberg, Berlin, 2001.

[Chi92] Chinchor, N.: *The Statistical Significance of the MUC-4 Results.* In: Proceedings, Fourth Message Understanding Conference (MUC-4). Morgan Kaufmann, San Mateo (CA), 1992.

[CHI97A] Chinchor, N.: *MUC-7 Named Entity Task Definition. Version 3.5.* 17. September 1997.
http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/ne_task.html

[CHI97B] Chinchor, N.; Hirschman, L.; Lewis, D. D.: *Evaluating Message Understanding Systems: An Analysis of the Third Message Understanding Conference (MUC-3).* In: Computational Linguistics 19 (3), S. 409-449. 1993.

[CHI01A] Chinchor, N.; Harman, D.: *Named Entity Scores - English.* 15. März 2001.
http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/ne_english_score_report.html

[CHI01B] Chinchor, N.; Harman, D.: *Named Entity Scores - Japanese.* 15. März 2001.
http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/ne_japanese_score_report.html

[CHI] Chinchor, N.: *MUC-7 Test Scores Introduction (Science Applications International Corporation).* O.J.
http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_proceedings/muc7_score_intro.pdf

[CUC99] Cucchiarelli, A.; Luzi, D.; Velardi, P.: *Semantic Tagging of Unknown Proper Nouns.* Universities of Ancona and Rome (Italy). In: Int. Journal of Natural Language Engineering, Special Issue on 'Semantic Tagging'. 1999.

[CUN99] Cunningham, H.: *Information Extraction - a User Guide (Second Edition).* University of Sheffield (UK), 1999.
ftp://ftp.dcs.shef.ac.uk/home/hamish/auto_papers/Cun99c.ps.gz

[DUD98] Drosdowski, G. (Hg.): *Duden Grammatik der deutschen Gegenwartssprache.* 6. Auflage. Bd. 4 von: Der Duden in 10 Bänden: Das Standardwerk zur deutschen Sprache. Bibliographisches Institut Mannheim, Wien, Zürich, 1998.

[FUK98] Fukumoto, J.; Masui, F.; Shimohata, M.; Sasaki, M.: *Description of the Oki System as Used for MUC-7.* Oki Electric Industry Co., Ltd., Osaka (Japan), 1998.
http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_proceedings/oki_muc7.pdf

[GAI95] Gaizauskas, R.; Wakao, T.; Humphreys, K.; Cunningham, H.; Wilks, Y.: *University of Sheffield: Description of the LaSIE System as used for MUC-6.* University of Sheffield (UK), 1995.
ftp://ftp.dcs.shef.ac.uk/home/robertg/muc6.ps

[Gal92] Gale, W.; Church K.W.; Yarowsky, D.: *One sense per discourse.* In: Proceedings of the DARPA speech and Natural Language workshop. Harriman, NY, February 1992.

[Gal96] Gallippi, A. F.: *Learning to Recognize Names Across Languages.* In: Proceedings of the 16th International Conference on Computational Linguistics (COLING-96). 1996.

[Gar98] Garigliano, R.; Urbanowicz, A.; Nettleton, D. J.: *University of Durham: Description of the LOLITA system as used in MUC-7.* University of Durham (UK), 1998.
`http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_proceedings/lindek_uman_muc7.pdf`

[Hum98] Humphreys, K.; Gaizauskas, R.; Azzam, S.; Huyck, C.; Mitchell, B.; Cunningham, H.; Wilks, Y.: *University of Sheffield: Description of the LaSIE-II System as Used for MUC-7.* University of Sheffield (UK), 1998.
`http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_proceedings/sheffield_muc7.pdf`

[Hum00] Humphreys, K.; Gaizauskas, R.; Cunningham, H.: *LaSIE Technical Specifications.* University of Sheffield (UK), 2000.
`ftp://ftp.dcs.shef.ac.uk/home/robertg/lasie_specs_tr.pdf`

[Jay91] Jaynes, E.T.: *Notes on Present Status and Future Prospects.* Washington University, St. Louis, 1991.
`http://bayes.wustl.edu/etj/articles/present.status.pdf`

[Kob99] Kober, K.; Krumeich, A.; von der Landwehr, K.; Langer, H.; Rehm, G.: *Projektbericht pronto: Probleme der Eigennamenerkennung.* Institut für Semantische Informationsverarbeitung, Universität Osnabrück. Unveröffentlichtes Manuskript.

[Kru98] Krupka, G. R.; Hausman, K.: *IsoQuest Inc.: Description of the NetOwlTM Extractor System as Used for MUC-7.* IsoQuest Inc., Fairfax (VA), 1998.
`http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_proceedings/isoquest.pdf`

[Lin95] Lin, D.: *University of Manitoba: Description of the PIE System Used for MUC-6.* University of Manitoba (Canada). In: Proceedings of the Message Understanding Conference (MUC-6). 1995. S. 113-126.

[Lin98] Lin, D.: *Using Collocation Statistics in Information Extraction.* University of Manitoba (Canada) 1998.
`http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_proceedings/lindek_uman_muc7.pdf`

[Man96] Mani, I.; MacMillan, R.: *Identifying Unknown Proper Names in Newswire Text.* In: Boguraev, B.; Pustejovsky, J. (Hgg.): Corpus Processing for Lexical Acquisition. Cambridge (Massachusetts). A Bradford Book, MIT, 1996. S. 41-59.

[Man99] Manning, C. D; Schütze, H.: *Foundations of Statistical Natural Language Processing.* MIT, Cambridge, (Massachusetts), London (England), 1999.

[McD93] McDonald, D. D.: *Internal and External Evidence in the Identification and Semantic Categorization of Proper Names.* In: Boguraev, B.; Pustejovsky, J. (Hgg.): Corpus Processing for Lexical Acquisition. Cambridge (Massachusetts). A Bradford Book, The MIT Press, 1996. S. 21-39.

[Mil98] Miller, S.; Crystal, M.; Fox, H.; Ramshaw, L.; Schwartz, R.; Stone, R.; Weischedel, R. and the Annotation Group (BBN Technologies) BBN: *Algorithms that learn to extract information - Description of the SIFT System as Used for MUC-7.* BBN Technologies, Cambridge (Massachusetts), 1998.
`http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_proceedings/bbn_muc7.pdf`

[Mik97] Mikheev, A.: *Automatic Rule Induction for Unknown Word Guessing.* In: Computational Linguistics 23 (3), S. 405-423. 1997.

[Mik98] Mikheev, A.; Grover, C.; Moens, M.: *Description of the LTG System Used for MUC-7.* University of Edinburgh (UK), 1998.
`http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_proceedings/ltg_muc7.pdf`

[Mik99] Mikheev, A.; Grover, C.; Moens, M.: *XML tools and architecture for Named Entity recognition.* University of Edinburgh (UK). In: Journal of Markup Languages: Theory and Practice. Vol 1, Issue 3. MIT, 1999. S. 89-113.
`http://www.ltg.ed.ac.uk/papers/99mikheev_markup.pdf`

[Mil90] Miller, G.: *Wordnet: An online lexical database.* In: International Journal of Lexicography, 3(4), 1990.

[Mit97] Mitchell, T. M.: *Machine Learning.* McGraw-Hill, 1997.

[Mor95] Morgan, R.; Garigliano, R.; Cllaghan, P.; Poria, S.; Smith, M.; Urbanowicz, A.; Collingham, R.; Costantino, M.; Cooper, C. *University of Durham: Description of the LOLITA system as used in MUC-6.* University of Durham (UK). In: Proceedings of the Message Understanding Conference (MUC-6). 1995. S. 71-85.

[Pai96] Paik, W.; Liddy, E.D.; Yu, E.; McKenna, M.: *Categorizing and Standardizing Proper Nouns for Efficient Information Retrival.* In: Boguraev, B.; Pustejovsky, J. (Hgg.): Corpus Processing for Lexical Acquisition. Cambridge (Massachusetts). A Bradford Book, The MIT Press, 1996. S. 61-73.

[Qui86] Quinlan, J. R.: *Induction of decision trees*. Machine Learning, Vol. 1, Nr. 1., 1986.

[Qui93] Quinlan, J. R.: *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo (CA), 1993.

[Qui85] Quirk, R.; Greenbaum, S.; Leech, G.; Svartvik, J.: A Comprehensive Grammar of the English Language. Longman Group Limited, New York, 1985.

[Rat97] Ratnaparkhi, A.: *A Simple Introduction to Maximum Entropy Models for Natural Language Processing*. University of Pennsylvania, 1997.
`ftp://ftp.cis.upenn.edu/pub/ircs/tr/97-08.ps.Z`

[Rat98] Ratnaparkhi, A.: *Maximum Entropy Models for Natural Language Ambiguity Resolution*. University of Pennsylvania, 1998. (Diss.)
`ftp://ftp.cis.upenn.edu/pub/ircs/tr/98-15/98-15.ps.gz`

[Rin99] Rinaldi, F.; Black, W.J.: *A Named Entity Extraction System and its Web Extensions*. UMIST Manchester (UK), 1999.
`http://www.ifi.unizh.ch/CL/rinaldi/PAPERS/vextal.pdf`

[Rot01] Roth, J.: *Automatische Erkennung von Produktenamen. Schlussbericht über das Perl-Programm ProdEx*. Universität Zürich, Philosophische Fakultät, 2001. (Seminararbeit)

[Sch00] Schneider, G.: *Token, Types, Häufigkeiten und automatische Wortartenerkennung (Statistik-basiertes Tagging). Morphologieanalyse und Lexikonaufbau (6. Vorlesung)*. Vorlesung Universität Zürich, Institut für Computerlinguistik, 2000.
`http://www.ifi.unizh.ch/CL/gschneid/LexMorphVorl/Lexikon06.Freq.html`

[Sek98a] Sekine, S.: *NYU: Description of the Japanese NE System used for MET-2*. New York University, 1998.
`http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_proceedings/nyu_muc7.pdf`

[Sek98b] Sekine, S.; Grishman, R.; Shinnou, H.: *A Decision Tree Method for Finding and Classifying Names in Japanese Texts*. New York University, 1998.
`http://www.cs.nyu.edu/~sekine/papers/wvlc98.ps`

[Ste01] Steiner, I.: *Warum "Named Entities" für die Chunk-Analyse wichtig sind*. In: Lobin, H. (Hg.): Proceedings der GLDV-Frühjahrstagung 2001. Universität Giessen (Deutschland), 2001. S. 245-252.

[Vil95] Vilain, M.; Burger, J.; Aberdeen, J.; Connolly, D.; Hirschman, L.: *A Model-Theoretic Coreference Scoring Scheme*. In: Proceedings, Fourth Message Understanding Conference (MUC-4). Morgan Kaufmann, San Mateo (CA), 1995.

[Vol01] Volk, M.: *The Automatic Resolution of Prepositional Phrase - Attachment Ambiguities in German*. University of Zurich, Faculty of Arts, 2001. (Habil.)

[Wac97] Wacholder, N.; Ravin, Y.; Choi, M.: *Disambiguation of Proper Names in Text.* Columbia University New York and TJ Watson Research Center IBM Yorktown Heights (NY), 1997.
`http://www.research.ibm.com/talent/documents/anlp97.pdf`

[Tak96] Wakao, T.; Gaizauskas, R.; Wilks, Y.: *Evaluation of an Algorithm for the Recognition and Classification of Proper Names.* University of Sheffield (UK). In: Proceedings of the 16th International Conference on Computational Linguistics (COLING96). 1996.
`ftp://ftp.dcs.shef.ac.uk/home/robertg/coling96a.ps`

Lebenslauf

Jeannette Roth - Herbstweg 14 - 8050 Zürich

"Da steh ich nun ich armer Tor! Und bin so klug als wie zuvor", möchte ich mit Goethes Faust manchmal sagen. Da hab ich nun ein reifes Alter erreicht (geboren bin ich am 28. November 1970), habe von 1977-1986 die obligatorische Schule und von 1986-1990 das Gymnasium, Typus C, im Wohnkanton Aargau absolviert, nach einem Zwischenjahr (Postbotin in Baden, Aide infirmière in Neuchâtel) in Zürich eine kaufmännische Lehre gemacht (1991-1993), um seither an der philosophischen Fakultät der Universität Zürich Deutsche Sprach- und Literaturwissenschaft, Computerlinguistik und Informatik zu studieren. Und nach all dieser Aus-Bildung weiss ich doch noch immer nicht, welche Berufung mir bestimmt ist...

"Dass ich erkenne, was die Welt im Innersten zusammenhält." - Im Gegensatz zu Faust glaube ich jedoch erkannt zu haben, was die Welt im Innersten zusammenhält: Es sind dies die Freundschaft, die Liebe und was mit diesen verbunden ist - die Hilfsbereitschaft. Und solche durfte ich beim Erstellen dieser Lizentiatsarbeit mannigfach erfahren. All den lieben Menschen, die dazu beigetragen haben, dass dieses Werk zu Stande kommen konnte, sei an dieser Stelle gedankt. Allen voran meinen Eltern für ihre grosse Geduld in allen Bereichen und meinem Freund Silvio Meier für technische, inhaltliche und moralische Unterstützung. Grossen Dank schulde ich auch Cerstin Mahlow fürs vielfache Durchlesen und die ebenso häufige technische Beratung, Fabio Rinaldi für gleiche Dienste und Esther Kaufmann für die letztmalige Gesamtlektüre. Auch weiteren Computerlinguistinnen und -linguisten am Institut für Computerlinguistik sei gedankt, sei es für inhaltliche, (sprach-)technische oder moralische Unterstützung: Martin Volk, Manfred Klenner, Simon Clematide, James Dowdall, Jennya Dobreva, Kai-Uwe Carstensen und meinem Betreuer Prof. Dr. Michael Hess. All den (Studien-)freundinnen und -freunden gebührt ebenso ein herzliches Dankeschön - gerade sie waren und sind der Kitt, der meine Welt im Innersten zusammenhält.

Zürich, 6. Dezember 2002
Jeannette Roth