Sprachmodelladaption mit Hilfe des World Wide Web

Studienarbeit
von
Sven Haidan

Interactive Systems Laboratories (ISL)
Carnegie Mellon University, Pittsburgh, PA, USA
Universität Fridericiana zu Karlsruhe (TH), Karlsruhe, Deutschland

Betreuer:
Prof. Dr. rer.nat. Alexander Waibel
Dipl.-Inform. Matthias Eck
Dr. rer. nat. Stephan Vogel

15. August 2007 - 15. November 2007

Zusammenfassung

In dieser Studienarbeit wird untersucht, wie das Sprachmodell eines statistischen Übersetzungssystems mit Hilfe von Information-Retrieval-Methoden und des World-Wide-Webs angepasst werden kann. Dabei werden für jeden Satz eines Dokumentes, das übersetzt werden soll, ähnliche Sätze ermittelt, aus denen dann ein Adaptionssprachmodell erstellt wird. Diese Arbeit lehnt sich an die Herangehensweise von [14] an, unterscheidet sich aber darin, dass die Textbasis zur Konstruktion des Sprachmodells nicht lokal vorliegt, sondern erst über Suchmaschinen aus dem World-Wide-Web erzeugt werden muss. Wir werden in unseren Experimenten verschiedene Methoden zur Anfragegenerierung für Suchmaschinen vorstellen. Ebenso werden verschiedene Methoden zur späteren Bewertung und Auswahl von Daten aus den zurückgelieferten Dokumenten untersucht. Wir vergleichen diese Methoden und stellen die Ergebnisse in Bezug auf die Übersetzungsleistung vor.

Danksagungen

Zuallererst möchte ich mich bei meinem Betreuer Matthias Eck für seine großartige Unterstützung bedanken. Ich bin ihm sowohl dankbar für seine fachlichen Ratschläge und Anregungen bezüglich dieser Studienarbeit, als auch für seine Unterstützung vor und während meines Aufenthaltes an der CMU in Pittsburgh.
Weiterhin möchte ich Herrn Professor Dr. Alex Waibel danken, dass er mir die Möglichkeit gegeben hat, die Forschungsarbeiten für diese Studienarbeit an der Carnegie-Mellon-University in Pittsburgh, durchzuführen. Ebenfalls bedanke ich mich für die Unterstützung und die Ratschläge der SMT-Research-Group am ISL. Stephan Vogel, Joy Ying und Ashish Venugopal konnten wir wertvolle Ratschläge geben und waren mir eine Hilfe bei der Benutzung des CMU-Statistical-Machine-Translation-Systems. Besonderer Dank geht an Bing Zhao für seine Unterstützung mit den Sprachmodell-komponenten und seine Ratschläge und Geduld in den bereichernden Diskussionen. Auch danke ich Thomas Schaaf und Silja Hildebrand für ihre Unterstützung. Zum Schluss geht mein Dank noch an Michael Bett für die technische Unterstützung an der CMU.

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Die Übersetzungsqualität eines statistischen Übersetzungssystems hängt sowohl von den verwendeten Modellen, als auch von den Korpora ab, mit deren Hilfe die Parameter der verwendeten Modelle geschätzt werden. Dabei hat sich herausgestellt, dass die Übersetzungsqualität umso besser ist, je stärker die Trainingsdaten mit den Testdaten übereinstimmen.
So übersetzt beispielsweise ein Übersetzungssystem, das auf Liebesbriefen trainiert wurde, Liebesbriefe besser als wissenschaftliche Artikel. Die Liebesbriefe stimmen hier stylistisch mit den Trainingsdaten überein. Ein anderer Aspekt wäre das Thema eines Textes. Ein System, das wissenschaftliche Artikel übersetzen soll und auf wissenschaftlichen Artikeln trainiert wurde (gleicher Stil), kann verbessert werden, wenn z.B. bekannt ist, dass es sich bei den wissenschaftlichen Artikeln um medizinische Artikel handelt. Das System kann nun auf medizinischen Artikeln trainiert werden. Anstatt immer ein neues System zu trainieren, kann auch ein Basissystem verwendet und dynamisch angepasst werden. Dies kann geschehen, indem das Sprachmodell mit einem weiteren Sprachmodell, dass aus einem spezifischerem, meist kleinerem Korpus aufgebaut wurde, kombiniert wird.

1.2 Ziel der Arbeit

Diese Arbeit hat das Ziel, das Sprachmodell eines statistischen Übersetzungssystem zu verbessern, indem dieses an jeden zu übersetzenden Satz eines Textes angepasst wird. Dadurch soll die Übersetzungsleistung des Gesamtsystems verbessert werden. Die Anpassung erfolgt, indem basierend auf einer ersten Übersetzung passende Texte im World-Wide-Web gesucht und dann zur Konstruktion eines adaptierten Sprachmodells herangezogen werden. Um geeignete Sätze zu finden, werden mit Hilfe einer Suchmaschine geeignete Webseiten gesucht, aus denen dann in einem zweiten Schritt ein geeigneter Korpus generiert wird. Das Web als Datenquelle bietet sich insofern an, als dort im Allgemeinen wesentlich mehr Daten vorliegen als in einem lokalen Index. Zudem wächst das WWW kontinuierlich und nimmt auch schneller neue Wörter auf als dies bei einem lokalen Index möglich ist.

2 Automatische Sprachübersetzung

2.1 Einleitung

Sprache kann auf verschiedene Weise verarbeitet werden. Dazu gehört z.B. der Ansatz, der Sprachen durch Grammatiken beschreibt. Weitaus erfolgreicher ist allerdings der Ansatz, der statistische Modelle und Methoden verwendet. Über einen Korpus - eine große Sammlung von Texten - werden Statistiken erstellt, welche dann die Basis bilden um Sprache weiter zu verarbeiten. Um dies tun zu können, muss der Rechner lernen. Dies geschieht, indem basierend auf Lernmaterial, z.B. Texten in einer oder mehreren Sprachen (bilingual, multi-lingual), Parameter für Modelle geschätzt werden. Die Qualität einer maschinellen Übersetzung ist sowohl von den gewählten Modellen des statistischen Übersetzungssystems und von den Methoden, mit denen die Parameter der Modelle geschätzt werden können, als auch von den vorliegenden Lernmaterialien abhängig.
Sprache ist aus mehreren Gründen schwierig zu verarbeiten. Sprache ist nicht statisch - sie verändert sich über die Zeit. Wörter ändern ihre Bedeutung, werden in anderen Zusammenhängen benutzt oder verschwinden gar ganz aus dem Wortschatz. Auch verwenden verschiedene Personengruppen verschiedene Wörter und Ausdrücke, obwohl man trotzdem der Aussage zustimmen würde, dass sie alle dieselbe Sprache, z.B. Deutsch, sprechen. Auch kann die Frage, ob ein konkret vorliegender Satz korrektes Deutsch darstellt, von verschiedenen Sprechern durchaus anders gesehen und beantwortet werden. Ein großes Problem der Sprachverarbeitung ist, die Mehrdeutigkeit von Sprache aufzulösen. Mehrdeutigkeit kann auf verschiedenen Ebenen auftreten: Mehrdeutigkeit bzgl. der Bedeutung eines Wortes: das spanische Wort „sol" kann sowohl für die Sonne stehen, als auch für die Note „g". Die Wortart kann nicht eindeutig sein: „play" im Englischen kann zugleich das Nomen „Schauspiel", als auch das Verb „spielen" sein. Ebenfalls können Mehrdeutigkeiten bzgl. syntaktischer Struktur und semantischem Gültigkeitsbereich auftreten.

2.2 Grundlagen

Die Aufgabe der automatischen Sprachübersetzung (Statistical-Machine-Translation; SMT) ist es, einen Text in einer Quellsprache Q in eine Zielsprache Z zu überführen. Dabei müssen die in der Einleitung erwähnten Probleme beachtet werden. Sprachübersetzung ist eine schwierige Aufgabe, da hierfür ein tiefes Verständnis des Textes und der Situation nötig ist. Man gibt sich je nach Bedürfnislage mit Abstufungen bzgl. der Übersetzungsqualität zufrieden. Wenn nur ein ungefähres Verständis eines Textes nötig ist, z.B. wenn man wissen möchte, worum es auf einer Webseite geht, dann ist eine ungefähre Übersetzung ausreichend. Interessante Seiten können dann von einem menschlichen Übersetzer oder einem höherwertigen Übersetzungssystem oder mit mehr Ressourcen (Speicher, Zeit, Prozessorleistung) übersetzt werden. Zum anderen ist Übersetzung von Texten, die nur ein eingeschränktes Vokabular und eine ähnliche Struktur in den Formulierungen haben (z.B. Wettervorhersage) leichter zu bewerkstelligen. Auch können Texte extra so formuliert werden, dass sie leicht in andere Sprachen übersetzt werden können. Dieses Prinzip wurde z.B. bei einigen Gebrauchsanleitungen angewendet.
Moderne Übersetzungssysteme basieren zum größten Teil auf statistischen Modellen. Die mathematischen Modelle, auf denen SMT fußt, basieren auf dem Noisy-Channel-Ansatz (siehe Abbildung 1), wie er auch in der Informationstheorie verwendet wird.

Da diese Arbeit auf Sprachmodelladaption abzielt, stellen wir die Übersetzungsmodelle und die Dekoder nur kurz vor.
Der Ansatz der statistischen Übersetzungssysteme basiert auf der Formel von Bayes :

$$\widehat{t} = \arg\max_{t} P(t|s) \tag{1}$$

Abstrakt gesprochen ist der Prozess der Übersetzung die Transformation einer Sequenz von Wörtern $s = s^n = s_1^n = s_1 \dots s_n$ der Länge n in der Quellsprache in eine Sequenz von Wörtern $t = t^m = t_1^n = t_1 \dots t_m$ der Länge m in der Zielsprache. Hierbei wird der Vorgang der Übersetzung als die Wiederherstellung eines verrauschten Signals gesehen. Der Sender hat die Sequenz in der Zielsprache in den Kanal geschickt und sie ist verrauscht als Sequenz in der Quellsprache angekommen. Die Aufgabe des Übersetzungssystems ist es, das Ausgangssignal (die Sequenz in der Zielsprache) wieder herzustellen. Anhand der Formel von Bayes 1 können wir alle Elemente eines Übersetzungssystems identifizieren. Dazu formulieren wir die Formel (1) mit Hilfe von Bayes' Theorem um und erhalten

$$\widehat{t} = \arg\max_{t} P(t|s) = \arg\max_{t} \frac{P(s|t) * P(t)}{P(s)} = \arg\max_{t} P(s|t) * P(t) \tag{2}$$

Weil s bekannt und somit fest ist, kann $P(s)$ weggelassen werden, da wir über t maximieren. Ziel ist es, diejenige Phrase zu finden, die mit höchster Wahrscheinlichkeit durch den Kanal übertragen wurde. Anhand der Gleichung 2 lassen sich folgende Kom-

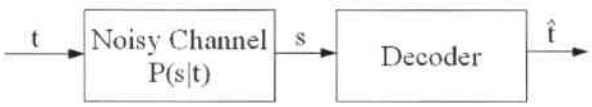

Abbildung 1: Das Noisy-Channel-Modell in der Linguistik

ponenten eines statistischen Übersetzungssystems identifizieren:

1. Das **Übersetzungsmodell** (Translation-Model,TM), das die Wahrscheinlichkeiten $P(s|t)$ schätzt, also die Wahrscheinlichkeit, dass s die Übersetzung von t ist.

2. Das **Sprachmodell** (Language-Model,LM), das die Wahrscheinlichkeit $P(t)$ in der Zielsprache schätzt, also die Wahrscheinlichkeit, dass t eine gültige Phrase in der Zielsprache ist.

3. Der **Dekodierer** (Decoder), der die Aufgabe hat, die beste Hypothese $\widehat{t}$ unter allen möglichen Übersetzungen t zu finden. Dafür verwendet er die Kombination der Schätzungen aus dem Sprach- und dem Übersetzungsmodell.

Dies ist das Basismodell. Es können zusätzlich noch weitere Modelle hinzugefügt werden. Als Beispiel wäre ein Satzlängen-Modell zu nennen. In den folgenden Abschnitten werden wir das Übersetzungsmodell und das Sprachmodell genauer erklären, wobei wir den Schwerpunkt auf die Beschreibung des Sprachmodells und der Sprachmodelladaption legen.

2.3 Übersetzungsmodell

Die Aufgabe des Übersetzungsmodells ist es anzugeben, welche Wörter oder Phrasen der Quellsprache mit welchen Wörtern der Zielsprache korrespondieren. Zur Schätzung der Übersetzungsmodelle werden bilinguale Korpora verwendet.
Der erste Schritt bei der Schätzung eines Modells ist eine Zuordnung, die angibt, welche Sätze der Quellsprache mit welchen der Zielsprache korrespondieren (sentence alignment). Danach wird auf Satzebene weitergearbeitet. Auf Satzebene stellt sich die Frage, wie ein Wort korrekt übersetzt werden kann, als auch die Frage nach der Wahl der korrekten Position innerhalb der Zielphrase.
Dabei unterscheiden sich die verschiedenen Übersetzungsmodelle vor allem in der Frage, welche Einschränkungen und Annahmen an diese Beziehungen gestellt werden. Dies kann z.B. eine paarweise (Wort in der Quellsprache und Wort in der Zielsprache) Beziehung sein, aber es könnte auch denkbar sein, dass eine Gruppe von Wörtern in der Quellsprache einer Gruppe von Wörtern in der Zielsprache zugeordnet wird. Es kann auch sein, dass ein Wort in einer Sprache keinem Wort in der anderen Sprache entspricht. Eine einschränkende Bedingung wäre die, dass jedem Wort in der Quellsprache genau ein Wort in der Zielsprache zugeordnet wird. Formal werden also Beziehungen zwischen dem Paar (s, t) gesucht, wobei $s = s^n = s_1 \dots s_n$ und $t = t^m = t_1 \dots t_m$ Man beachte, dass die Wortsequenzen unterschiedliche Länge haben können.
Übersetzungsmodelle sind üblicherweise entweder wort- oder phrasenbasiert. Das einfachste Übersetzungsmodell wäre eine wortgetreue Übersetzung (Unigramm-Modell). Dabei ergäbe sich: $p(s|t) = \prod_{i=1}^{n} p(s_i|t_i)$. Die Sätze wären also genau gleich lang und es gäbe keine Umpositionierung von Wörtern. Diese Annahme ist nicht sehr realistisch. Bessere und realistischere Modelle weisen den Wörtern eine Fertilität zu: Die Anzahl von Wörtern, in die sie übersetzt werden. Das Wort „Operationstisch" hätte z.B. bzgl. dem Englischen die Fertilität 2, da es zu „operating table" übersetzt wird. Für „operating" und „table" ist es schon schwieriger zu sagen, welches Wort die Fertilität 1 bzw. 0 hat. Die richtige Reihenfolge der Wörter muss dann durch das Sprachmodell erkannt werden, welches wir im nächsten Abschnitt vorstellen werden.
Andere Modelle beziehen Parameter wie die Position des Wortes im Satz, sowie die Satzlänge in die Betrachtung mit ein (Offset-Modell). Für Details bezüglich des Übersetzungsmodells verweisen wir auf [7], wo die IBM-Modelle 1-5 genauer beschrieben werden. In [36] wird ein Alignment mit Hilfe von Hidden-Markov-Modellen (HMMs) beschrieben.
Neben den Übersetzungsmodellen, die Wörter in der Quell- und Zielsprache einander zuordnen, gibt es eine weitere Klasse von Modellen, bei denen ganze Phrasen in der Quell- und Zielsprache einander zugeordnet werden. Hier sind das ISA-Modell [40] und das PESA-Modell [35] zu nennen. Das ISA-Modell führt die Segmentierung von Sätzen sowie die gegenseitige Zuordnung von Phrasen innerhalb eines Schrittes durch ohne eine vorher trainierte Wort-zu-Wort-Zuordnung zu benötigen.

2.4 Sprachmodell

Sprachmodelle werden in vielen Gebieten verwendet, z.B. in der automatischen Spracherkennung (Automatic-Speech-Recognition, ASR), Part-Of-Speech-Tagging, Parsing, Information-Retrieval und beim maschinellen Übersetzen (Machine-Translation, MT).
Die Aufgabe des Sprachmodells im Rahmen des maschinellen Übersetzen ist es, den Decoder mit Informationen über die Zielsprache zu versorgen. Mit Hilfe dieser Information kann der Decoder bei seiner Suche nach einer geeigneten Übersetzung ent-

scheiden, ob die aktuelle Hypothese eine gute Phrase in der Zielsprache ist. Betrachten wir die Phrase $t = t^m = t_1 \dots t_m$ so können wir die Wahrscheinlichkeit dieser Phrase auch mit Hilfe von bedingten Wahrscheinlichkeiten ausdrücken.

$$\begin{aligned} P(t) &= P(t^m) = P(t_1, t_2, \dots, t_m) \\ &= P(t_1) * P(t_2|t_1) * P(t_3|t_1, t_2) * \dots * P(t_m|t_1, t_2, \dots, t_{m-2}, t_{m-1}) \\ &= \prod_{i=1}^{m} P(t_i|t_1, \dots, t_{i-1}) \end{aligned} \tag{3}$$

Wir verwenden auch die Schreibweise h_i um die Historie von Wort t_i anzugeben. Wir erhalten also $h_i = h(t_i) = t_1, \dots, t_{i-1}$. Das Lernen eines Sprachmodells besteht nun darin, alle Wahrscheinlichkeiten für das Auftreten von Wortfolgen zu schätzen, die die Parameter des Modells darstellen. Dies ist allerdings problematisch, da sehr viele Parameter geschätzt werden müssten. Diese benötigen viele Trainingsdaten, um die Parameter korrekt und zuverlässig schätzen zu können. Bei der Berechnung von N-gram-Modellen liegen meist zuwenig Daten vor, um alle Parameter robust und zuverlässig zu schätzen (Data Sparsity). Desweiteren hätte man keinen Wahrscheinlichkeitsschätzwert für eine Phrase, die in einer Hypothese, aber nicht im Trainingset auftaucht. Hinzu kommt, dass der Bereich, den der Dekoder absuchen müsste, viel zu groß wäre um dies in akzeptabler Zeit zu erledigen.

2.4.1 N-gram-Sprachmodell

Eine Lösung dieses Problems besteht darin, die Historien auf eine feste Anzahl an Wörtern festzulegen. Dies entspricht einer Klassenbildung. Einer Klasse gehören all die Historien eines Wortes T_i an, die die letzten $n-1$ Wörter gemeinsam haben. Diese Art von Sprachmodell nennt man N-gram-LM.
Aufgrund der Anzahl zu schätzender Parameter nimmt n normalerweise Werte von 2-4 an und hängt von der Menge an vorhandenem Trainingstext ab. Kleinere n ermöglichen eine zuverlässigere Schätzung der Wahrscheinlichkeiten, während größere n ein präziseres Modell ergeben. Die Verwendung eines N-gram-Modells ist eine Möglichkeit, kurzfristige Kontextinformation zu erhalten. Langfristige Kontextinformation kann aber aufgrund der begrenzten Historie nicht beachtet werden. Um auch langfristige Kontextinformation zu verwenden, kann das Sprachmodell von Zeit zu Zeit angepasst werden, um z.B. einem Themawechsel oder einem Stilwechsel gerecht zu werden (vgl Abschnitt 4.1). Das N-gram-Modell ist ein sehr einfaches Modell, was den Vorteil hat, dass es leicht zu trainieren und einfach in einen Decoder zu integrieren ist. Die Einfachheit ist aber auch ein Nachteil, da Abhängigkeiten über längere Distanzen nicht beachtet werden (Themenänderung, Stiländerung), aber auch Phrasen nicht in die gleich Klasse eingeordnet werden, obwohl man dies erwarten würde. z.B. würde man die Phrasen *Peter liest Zeitung* und *Hans liest Zeitung* in die gleiche Klasse einordnen. Ein einfaches n-gam-Modell kann dies hingegen nicht. Das N-gram-Modell wird benutzt, da es am besten funktioniert. Das SuffixArray-LM [39] erlaubt sogar beliebig lange Histories.

Smoothing Zur Schätzung der Wahrscheinlichkeiten des Auftretens eines Wortes oder einer Phrase verwendet man den Maximum-Likelihood-Schätzer. Sei $n(t)$ die Anzahl der Vorkommnisse von Phrase t und N die Anzahl der betrachteten Einheiten (z.B. Unigrams oder Bigrams). Der Schätzwert der Wahrscheinlichkeit des Auftretens einer Phrase t berechnet sich dann nach $P(t) = \frac{n(t)}{N}$. Hierbei treten jedoch

Probleme auf. Zum einen werden bei kleineren Korpora diejenigen Phrasen, die im Trainigset auftreten überbewertet. Diejenigen, die nur in geringer Anzahl auftreten, werden dann überbewertet, falls sie im Trainingskorpus auftreten und unterbewertet, falls dies nicht der Fall ist (Bias). Problematisch ist, dass für im Trainingskorpus nicht aufgetretene Phrasen keine Wahrscheinlichkeitsmasse bereitgestellt wird. Dies kann aufgrund der kombinatorischen Explosion sehr schnell der Fall sein. Jeder Phrase, die eine Subphrase enthält, die nicht im Trainingskorpus vorhanden ist, wird fälschlicherweise eine Wahrscheinlichkeit von 0 zugeordnet. Techniken zur Abschwächung des Bias und zur Bereitstellung von Wahrscheinlichkeitsmasse für vorher nicht gesehene Phrasen werden unter dem Begriff **Smoothing** zusammengefasst. Wir stellen hier die drei geläufigen Techniken lineare Interpolation, Discounting und Backing-off vor.

Lineare Interpolation Die einfachste Art und Weise, um Wahrscheinlichkeiten $P(t_i|h_i) = 0$ zu verhindern, ist eine lineare Kombination der kleineren N-gramme. Für den Fall von Trigrammen ergibt sich folgende Formel:

$$P(t_i|t_{i-2}, t_{i-1}) = \lambda_1 \frac{n(t_i)}{N} + \lambda_2 \frac{n(t_{i-1}), t_i}{n(t_{i-1})} + \lambda_3 \frac{n(t_{i-2}), t_{i-1}, t_i}{n(t_{i-2}, t_{i-1})} \tag{4}$$

mit $\sum_{i=1}^{3} \lambda_i = 1$. Die Anzahlen $n(t)$ werden auf einem Teil des Trainingstextes ermittelt. Die λ_i werden trainiert, indem die Wahrscheinlichkeit des Rests des Trainingskorpus maximiert wird.

Discounting Bei der Discounting-Methode wird explizit Wahrscheinlichkeitsmasse für zuvor nicht gesehene Phrasen bereitgestellt. Dazu werden die Anzahlen(*Counts*) $n(t)$ durch Multiplikation mit einem Discount-Koeffizienten $d_{n(t)}, 0 \leq d_{n(t)} \leq 1$ $\forall n(t) \geq 1$ verringert: $n^*(t) = d_{n(t)} * n(t)$. Damit ergibt sich die Wahrscheinlichkeit $P(t) = \frac{d_{n(t)} * n(t)}{N}$.

Backing-off Anstatt wie bei der linearen Interpolation Werte zu kombinieren, greift man beim Backing-off auf weniger spezifische Werte zurück. Wir stellen dies am Beispiel eines Trigram-Modells vor. Um zu garantieren, dass die Summe der bedingten Wahrscheinlichkeiten $\sum v \in V P(v|t_{i-2}, t_{i-1})$ (V ist das Vokabular) dennoch 1 ergibt, werden sogenannten Back-off-Gewichte $\alpha(t_{i-2}, t_{i-1})$ eingeführt. Damit ergibt sich die Wahrscheinlichkeit zu

$$P(t_i|t_{i-2}, t_{i-1}) = \begin{cases} \frac{n^*(t_{i-2}^i)}{n(t_{i-2}^{i-1})} & \text{falls } n(t_{i-2}^i) \leq 1 \\ \alpha(t_{i-2}^{i-1}) P(t_i|t_{i-1}) & \text{sonst} \end{cases} \tag{5}$$

2.4.2 Verallgemeinerung der N-gram-Sprachmodells

Anstatt wie beim N-gram-Modell auf einzelnen Wörtern zu arbeiten, kann man die Wörter auch Klassen zuordnen. Diese Klassen können z.B. Wortarten sein oder z.B. die Klasse der Wochentage oder irgendeine automatisch gefundene Klassifizierung. Dabei kann ein Wort natürlich auch mehreren Klassen zugeordnet sein. So kann das englische Wort *matter* sowohl Substantiv (*Angelegenheit*) als auch Verb sein (*von Bedeutung sein*). Sei $g(t_i)$ die Klasse des Wortes t_i. Es gibt mehrere Möglichkeiten die Wahrscheinlichkeit $P(t_i|h_i)$ zu schätzen. Eine häufig benutzte Möglichkeit ist

$$P(t_i|hi) = P(t_i|g(t_i)) * P(g(t_i)|g(t_{i-2}), g(t_{i-1})) \tag{6}$$

Ein Vorteil von klassenbasierten Sprachmodellen ist, dass sie durch die Reduktion auf die Klassen wesentlich kompakter sind als herkömmliche N-gram-Modelle, da es wesentlich weniger Klassen als Wörter gibt. Dies ermöglicht zum einen eine Einsparung an Speicherplatz und Rechenzeit, die entweder für eine Erhöhung des Parameters n oder für die Integration weiterer Modelle genutzt werden kann. Zum anderen sind die Schätzwerte dadurch wesentlich zuverlässiger, da mehr Daten für die Schätzung vorliegen und es können auch Vorhersagen getroffen werden für Phrasen, die vorher noch nicht gesehen wurden. Trat die Phrase $t_{i-2}t_{i-1}t_i$ z.B. nicht oder nur unzureichend oft für eine zuverlässige Schätzung der N-gram-Parameter im Trainingskorpus auf, so ist es doch wesentlich wahrscheinlicher, dass die Klassenfolge der Phrase $g(t_{i-2})g(t_{i-1})g(t_i)$ im Trainingskorpus vorliegt. Ein Nachteil der Klassen-basierten Sprachmodells ist, dass spezifische wortbezogene Information verloren gehen kann. Dies kann durch die Kombination eines Klassen-basierten und eines herkömmlichen N-gram-Modells verhindert werden. Die Einschränkungen des N-gram-Modells durch die Nichtbeachtung von langfristigen Kontextinformationen kann ein klassenbasiertes Sprachmodell jedoch auch nicht aufheben.

2.4.3 Weitere Ansätze zur Sprachmodellbildung

Es gibt verschiedene Ansätze, die komplett anders sind als der N-gram-Ansatz. Wir stellen diese Modelle hier nur kurz vor und verweisen auf die Literatur, da in der heutigen Praxis in erster Linie das N-gram-Modell und seine Varianten verwendet werden.

Baumbasierte Sprachmodelle [2] verwenden Entscheidungsbäume, an dessen Knoten Fragen an die Worthistorie gestellt werden. Aufgrund der großen Menge an Fragen ist die Suche nach einem guten Entscheidungsbaum zu aufwendig.

Probabilistisch-kontextfreie Grammatiken sind ein weiterer Ansatz ein Sprachmodell zu definieren[28] [5] [29]. Eine Grammatik besteht aus Terminalsymbolen (den Wörtern), Nichtterminalsymbolen und Regeln, wie die Nichtterminalsymbole zueinander in Beziehung gesetzt werden können. Mit Hilfe einer Grammatik kann ein Satz syntaktisch geparst werden. Jede Regel hat eine durch Training ermittelte Wahrscheinlichkeit, mit dessen Hilfe über alle Parsemöglichkeiten eines Satzes seine Wahrscheinlichkeit ermittelt werden kann. Der größte Nachteil dieses Ansatzes ist der hohe Rechenaufwand. Ein weiterer ist der Verlust von semantischer Information, da es sich um einen syntaktischen Ansatz handelt. Im Vergleich zu den N-gram-Modellen kann dieser Ansatz jedoch Abhängigkeiten über längere Entfernungen erkennen.

2.4.4 Metriken zur Evaluierung von Sprachmodellen

Warum braucht man Metriken zur Evaluierung von Sprachmodellen? Bei der Entwicklung und Verbesserung von Sprachmodellen ist es wichtig, die vorgenommenen Änderungen schnell auf ihre Wirksamkeit zu überprüfen. Bewertungen durch Menschen können nur für die Übersetzungsleitung als Ganzes vorgenommen werden. Sie sind hier fehl am Platz, da sie zum einen zu teuer sind, zu viel Zeit in Anspruch nehmen und nicht wiederverwendbar sind. Zum anderen ist nicht immer klar, wie Verbesserungen in der Übersetzungsleitsung auf die einzelnen Komponenten des Systems zu verteilen sind, da Sprachmodell und Übersetzungsmodell sich gegenseitig unterstützen. Generell ist festzustellen, dass Metriken zur Evaluierung von Sprachmodellen blind sind für alles, was nichts mit dem Sprachmodell zu tun hat. Hierzu zählen die bereits genannte

Interaktion mit dem Übersetzungsmodell und die Suchalgorithmen (Decoder).
Das populärste Maß zur Bewertung eines Sprachmodells ist die Perplexität (PP). Perplexität ist ein informationstheoretisches Maß. Die Perplexität $PP(p_M)$ eines Sprachmodells p_M über dem Test-Set $T = \{w_1, \ldots, w_t\}$ (Test-Set-Perplexität) berechnet sich als:

$$PP_T(p_M) = \left(\prod_{i=1}^{t} p_M(w_i|w_1 \ldots w_{i-1}) \right)^{-\frac{1}{t}} = b^{-\sum_{i=1}^{t} \log_b(p(w_i|w_1 \ldots w_{i-1}))/t}. \quad (7)$$

Als Basis b wird gerne $b = 2$ oder $b = 10$ gewählt. Die Logarithmusformel der Perplexität wird zum einen benutzt, um die Werte addieren zu können. Dies ist effizienter anstatt sie wie nach Definition zu multiplizieren. Ein weiterer Grund ist, dass das Produkt aus vielen Wahrscheinlichkeiten immer kleiner wird und somit zu klein werden kann und auf dem Rechner einen Unterlauf erzeugt. Je geringer die Perplexität ist, desto besser ist das Modell, d.h. umso besser beschreibt es die Daten.
Eine Perplexität von k lässt sich interpretieren als ein Maß für die Überraschung bei der zufälligen Ziehung aus einer Menge von k gleichwahrscheinlichen Elementen. Sie kann also auch als der durchschnittliche Verzweigungsfaktor angesehen werden. $\log PP_T(p_M)$ ist eine obere Schranke für die Anzahl an Bits pro Wort, das mit Modell M kodiert wird.
Bei der Perplexität muss man unterscheiden zwischen Test-Set-Perplexität, welche angibt, wie gut ein Test-Set durch ein Sprachmodell vorausgesagt wird. Demgegenüber sagt die Training-Set-Perplexität aus, wie gut die Datenbasis, aus der das Sprachmodell erzeugt wurde, durch das Sprachmodell beschrieben wird.
Die Perplexität hat den Vorteil, dass sie effizient zu berechnen und unabhängig vom Übersetzungsmodell ist. So kann innerhalb kurzer Zeit eine Änderung am Sprachmodell bewertet werden. Dies ist wichtig, wenn verschiedene Änderungen auf ihre Tauglichkeit überprüft werden sollen. Der Nachteil ist, dass sie leider nicht gut mit der Word-Error-Rate (WER) [ASR] und der Qualität der Übersetzung korreliert. Perplexität korreliert aber gut mit WER bei N-gram-Sprachmodellen, welche über In-Domain-Data trainiert wurden [9].
Perplexität kann man nicht über Sprachmodelle mit unterschiedlichem Vokabular vergleichen und nur über normalisierte LMs. Da rechentechnisch leichter zu handhaben, wird statt der Perplexität häufig die log-Perplexität ($log - PP$ [bit]) verwendet. Ein weiterer Vorteil von Perplexity ist, dass dieses Maß keine detaillierte lexikalische Information, wie z.B. Part-Of-Speech-Information benötigt. In [9] wurde Perplexität und eine von den Autoren neu entwickelte Metrik untersucht, welche Informationen verwendet, die Perplexität nicht erfasst. Obwohl diese neue Metrik besser ist, ist die Korrelation mit WER immer noch nicht ausreichend, um damit ein mächtiges Werkzeug zur Bewertung von Sprachmodellen in der Hand zu haben.
Die WER hat den Nachteil, dass sie vom Übersetzungsmodell abhängig ist, Zugang zu diesem benötigt und ihre Berechnung aufwendig ist. Ein gutes Maß, um WER hervorzusagen, ist Trigram-Coverage [19], also der Anteil der Trigramme des Test-Sets, der in den Trainingsdaten vorliegt. Dieses Maß geht allerdings wenig auf das konkrete Sprachmodell ein. Es könnte aber dabei helfen, geeignete Trainingsdaten zur Sprachmodellkonstruktion auszuwählen.

2.5 Dekodierung

Die Dekodierung (Decoding) ist die Suche nach der besten Übersetzungshypothese. Die Dekodierung gliedert sich in zwei Phasen. In der ersten Phase wird eine Datenstruktur aufgebaut, die alle Informationen aus den verwendeten Übersetzungsmodellen enthält. In der zweiten Phase des Dekodierungsprozesses wird diese Datenstruktur dann durchsucht.

Aufbau des Übersetzungsgitters Die Datenstruktur, die im ersten Schritt aufgebaut wird, heisst Übersetzungsgitter (translation lattice). Der Satz der Quellsprache bildet die Ausgangsbasis bei der Konstruktion des Übersetzungsgitters. Es wird eine Pfeilfolge erstellt, wobei jedes Wort des Quellsatzes einem Pfeil zugeordnet wird. Die Reihenfolge der Pfeile entspricht dabei der Reihenfolge der Wörter im Satz. Dann werden sukzessive weitere Pfeile hinzugefügt. Hierzu werden für Wörter und Phrasen des Quellsatzes in den Übersetzungsmodellen passende Phrasen gesucht und deren Übersetzungen mit zugehörigen Übersetzungswahrscheinlichkeiten und weiteren Bewertungen über den Quellwörtern im Übersetzungsgitter als neuer Pfeil eingetragen. Das Übersetzungsgitter speichert dabei Informationen darüber, welche Teile des Satzes schon abgedeckt sind, Backtrackinginformationen und kumulierte Bewertungen.

Suche nach dem besten Pfad im Übersetzungsgitter In der zweiten Phase wird das Übersetzungsgitter nach dem besten Pfad durchsucht. Die Wortfolge dieses Pfades ist dann die beste gefundene Übersetzung. Hierzu werden weitere Modelle mit den Informationen aus dem Übersetzungsgitter kombiniert. Hierzu gehören das Sprachmodell sowie Satzlängenmodelle.
Um die Dekodierungszeit zu verringern, werden während des Dekodierungsprozesses verschiedene Pruningtechniken verwendet. Diese zielen darauf ab, den Suchraum zu verkleinern.

Wortumstellungen In verschiedenen Sprachen treten Wörter in verschiedenen Reihenfolgen auf. Um dieser Tatsache Rechnung zu tragen, müssen beim Übersetzungsprozess Wortumstellungen beachtet werden. Dies kann beim *monotonen Dekodieren* durch die Phrasenzuordnungen des Übersetzungsmodells geschehen. Beim *nichtmonotonen Dekodieren* werden Informationen aus dem Sprachmodell verwendet, um Umordnungen zu bewerten. Es gibt verschiedene Herangehensweisen, um Umordnungen vorzunehmen. Die einfachste Art ist, alle Permutationen der Wörter des Zielsatzes durchzuprobieren. Dies ist allerdings sehr rechenintensiv. Eine weitere Möglichkeit besteht darin, nur Wörter innerhalb eines Fensters umzuordnen, welches über den Zielsatz gleitet.

2.6 Automatische Evaluation der Qualität von maschinellen Übersetzungen

Wie in Abschnitt 2.4.4 für Sprachmodelle bereits erläutert, benötigt man auch für die Evaluierung des gesamten Übersetzungssystems eine automatische Evaluierung der Übersetzungsergebnisse. Dies muss schnell und kostengünstig geschehen und auch praktisch reproduzierbar sein. Die Entwickler neuer MT-Systeme benötigen ein schnelles und günstiges Feedback bezüglich ihrer neuen Ideen. Das Evaluationsproblem stellt den Flaschenhals der automatisierten Übersetzung dar.

In diesem Abschnitt stellen wir die vor allem im Bereich der Spracherkennung erfolgreiche Metrik Word-Error-Rate (WER) sowie die zwei populären Scores (BLEU und NIST) für die Bewertung von Übersetzungen vor. Diese Methoden sind leider nur eine Annäherung an die menschliche Bewertungsleistung. Aus den oben genannten Gründen muss dies aber hingenommen werden.

2.6.1 Maße zur Evaluation

Möchte man die Qualität einer Methode bezüglich einer Aufgabe messen, so ist der Grad an Aufgabenerfüllung das ultimative Maß der Qualität der Methode. Nun gibt es aber Fälle, in denen dieses Maß nicht explizit vorliegt, wie z.B. im Fall der Bewertung einer Übersetzungsleistung.
Menschliche Übersetzer entwickeln leicht ein Gefühl dafür, wie gut oder schlecht eine Übersetzung ist. Es fällt ihnen aber schwer, diese Einschätzung zu formalisieren und in Regeln zu fassen. Deshalb greift man in diesem Fall auf künstliche Ersatzmaße zurück. Ein Beispiel dafür ist die Perplexität, von der man sich erhofft, dass sie mit der Übersetzungsqualität korreliert, auch wenn dies nicht immer der Fall ist. Wir verweisen hier auch noch auf die Evaluationsmaße Precision und Recall aus dem Bereich des Information-Retrieval, welche in Kapitel 3 vorgestellt werden.

2.6.2 Bewertung von Übersetzungsqualität

Korrekte Übersetzungen einer Phrase müssen nicht hundertprozentig genau einer Referenzübersetzung entsprechen. Abweichungen bzgl. einiger Wörter und/oder der Reihenfolge der Wörter von einer Referenzübersetzung müssen keinen Verlust von Übersetzungsqualität darstellen. Deshalb ist es wünschenswert, mehrere Übersetzungen zur Bewertung zu verwenden, da so realistischere Bewertungen möglich sind. Menschliche Bewertungen wägen verschiedene Aspekte einer Übersetzung gegeneinander ab. Dazu gehören:

- Adäquatheit (adequacy): Sagt aus, wie gut die Bedeutung bei der Übersetzung pro Segment erhalten geblieben ist.
- Genauigkeit (informativeness): Gibt an, wieviel der Information über das ganze Dokument erhalten geblieben ist.
- Sprachbeherrschung/Flüssigkeit der Sprache (fluency): Sagt aus, wie flüssig die Sprache ist. Hierzu benötigt man keine Referenzübersetzung. Dies ist allein eine Eigenschaft eines Textes.

Die automatische Bewertung der Übersetzungsqualität eines Systems ist natürlich abhängig von der Qualität der verwendeten Referenzübersetzungen.

2.6.3 Word-Error-Rate

Die Word-Error-Rate (Wortfehlerrate;WER) ist eine Metrik aus dem Bereich der Spracherkennung und misst die Leistung eines Spracherkenners. Sie ist dabei ein Substitutionsmaß für die Erkennungsqualität, die nicht automatisch zu ermitteln ist. Um die WER zu berechnen, werden die minimale Anzahl der Ersetzungen S (substitutions), Löschungen L und Einfügungen E berechnet, die nötig sind, um den gesprochenen Satz (Referenzsatz) in den durch den Spracherkenner erkannten Satz umformen zu

können. Die Summe dieser drei Größen wird dann durch die Länge N des Referenzsatzes geteilt. Damit ergibt sich $WER = \frac{S+L+E}{N}$. Die WER ist ein Maß, das die gesamte Qualität der Sprachübersetzung bewertet. Wenn einzelne Komponenten eines Spracherkenners verbessert werden sollen und die Übersetzungsleistung diesen Komponenten zugeordnet werden soll, erweist sich die WER als nicht geeignet, da sie von der Leistung verschiedener Komponenten abhängt.

2.6.4 BLEU-Score

Die BLEU-Methode [26] (Bilingual Evaluation Understudy) wurde bei IBM entwickelt, um eine Metrik zur Verfügung zu haben, die eine Übersetzung annähernd so gut bewertet wie ein menschlicher Bewerter. Da Menschen Sprache verschieden übersetzen und diese unterschiedlichen Übersetzungen alle als korrekt bewerten, verwendet die BLEU-Methode auch verschiedene Referenzübersetzungen zur Evaluation.
BLEU bewertet die Übersetzungsqualität numerisch mit Hilfe eines Scores, der sich im Intervall [0,1] bewegt. Je näher die Übersetzung an einer professionellen ist, desto besser (höher) der Score. BLEU besteht aus einer Familie von Metriken, von denen hier die Baseline-Metrik vorgestellt wird.
BLEU basiert auf dem Vergleich von N-grammen der Übersetzungshypothese mit N-grammen der Referenzübersetzungen und deren Anzahl, ohne dabei die Position zu beachten. Dieser Vergleich führt zur *Modified-N-gram-Precision*, die zudem der Tatsache Rechnung trägt, dass Wörter höchstens so oft in der Hypothese auftreten dürfen, wie dies jeweils in den Referenzübersetzungen der Fall ist. Sei $refCount(ngram)$ die maximale Anzahl der Vorkommnisse eines N-grams $ngram$ in den einzelnen Referenzübersetzungen und $hypCount(ngram,)$ die Anzahl der Vorkommnisse des N-grams $ngram$ in der Hypothese, so berechnet sich die Modified-N-gram-Precision für einen Satz s nach der Formel

$$mnp_n(s) = \frac{\sum_{\text{n-grams}\in s} \min(\text{refCount(ngram)},\text{hypCount(ngram)})}{\text{Anzahl n-grams der Hypothese}} \tag{8}$$

Für einen ganzen Text wird die Modified-N-gram-Precision nach folgender Formel berechnet: $mnp_n(text) =$

$$\frac{\sum_{\text{satz}\in\text{text}} \sum_{\text{n-gram}\in\text{satz}} \min(\text{refCount(ngram)},\text{hypCount(ngram)})}{\sum_{\text{satz'}\in\text{text}} \sum_{\text{n-gram'}\in\text{satz'}} \text{hypCount(ngram')}} \tag{9}$$

Um diese verschiedenen Modified-N-gram-Precision-Werte zu kombinieren, verwendet BLEU die gewichtete Summe über den $\log mnp_n$.
Das nächste Problem, das BLEU beachtet, ist die Satzlänge. Durch die Modified-N-gram-Precision werden bereits Sätze bestraft, die Wörter enthalten, die nicht in den Referenzen auftreten, sowie das zu häufige Auftreten von Wörtern. Kurze Sätze, die einfach nur ein paar bekannte N-gramme enthalten, bekommen allerdings eine hohe Modified-N-gram-Precision. Um dem entgegenzuwirken, wird ein Faktor eingeführt, der zu kurze Sätze bestraft, die *brevity penalty* BP. Recall, ein sonst häufig verwendetes Mittel um dieses Problem anzugehen, ist hier nicht anzuwenden, da mehrere Referenzübersetzungen vorliegen. Eine gute Übersetzung verwendet jedoch nicht alle Wörter aller Referenzen. Die Brevity-Penalty bezieht sich auf den ganzen Korpus und nicht nur auf einzelne Sätze. Sei r die effektive Länge des Referenzkorpus. Diese berechnet sich als Summe der Längen aller Sätze, wobei für jeden Satz die Länge der Referenzübersetzung mit den höchsten N-gram-Übereinstimmungen verwendet wird.

Sei weiterhin c die Gesamtlänge aller Hypothesen. Dann berechnet sich die Brevity-Penalty nach der Formel

$$BP = \begin{cases} 1 & if c > r \\ e^{1-r/c} & if c \leq r \end{cases} \tag{10}$$

Damit ergibt sich der BLEU-Score zu

$$BLEU = BP * exp(\sum_{n=1}^{N} w_n log p_n,) \tag{11}$$

wobei die Gewichte zu $w_n = 1/N$ gewählt werden.

Stärken und Schwächen von BLEU Die Stärken der BLEU-Methode liegen in einer sprachunabhängigen Bewertung, geringen Kosten, schneller Ausführbarkeit und starker Korrelation mit menschlichen Bewertungen. Nachteilig sind hingegen, dass einzelne isolierte Sätze nicht adäquat bewertet werden können. Die Leistungsfähigkeit von BLEU wird höher, je mehr Referenzübersetzungen es gibt. Deshalb ist es schwierig, Vergleiche zu ziehen zwischen Systemen, die den gleichen Text übersetzen, dabei aber unterschiedlich viele Referenzübersetzungen verwenden. Desweiteren arbeitet der BLEU-Score nicht gut bei Sprachen ohne Wortgrenzen. Es gibt auch keine Garantie dafür, dass eine Erhöhung des BLEU-Scores auch eine Erhöhung der Übersetzungsqualität nach sich zieht.

2.6.5 NIST-Score

Ein zweiter populärer Score für die automatische Evaluation von maschineller Übersetzung ist der NIST-Score (National Institute of Standards and Technology). Der NIST-Score (NIST2002)[12] basiert auf dem BLEU-Score. Er verwendet ebenso wie der BLEU-Score Statistiken über das gleichzeitige Auftreten von Wörtern in der Hypothese und einer oder mehrerer Referenzübersetzungen. Es wird jedoch der arithmetische Mittelwert über die N-gram-Precisions berechnet. Der NIST-Score bestraft zu kurze Übersetzungen nicht so stark wie dies beim BLEU-Score der Fall ist. Der NIST-Score zeichnet sich gegenüber dem BLEU-Score durch den Information-Gain aus. D.h. ein seltenes Wort oder eine seltene Phrase korrekt zu haben wird stärker belohnt, als dies bei einem häufigem Wort oder einer häufigen Phrase der Fall ist. Das führt dazu, dass der NIST-Score mehr Wert auf korrekte Content-Words legt und damit die Adäquatheit stärker beachtet. Dies wird erreicht, indem die N-gram-Precisions gewichtet werden.

3 Information-Retrieval

3.1 Grundlagen

Information Retrieval (IR) beschäftigt sich mit dem Auffinden von Informationen aus Datenbeständen, wobei die Suche auf der inhaltlichen und nicht der rein syntaktischen Ebene stattfindet. Dabei hat der Benutzer ein Informationsbedürfnis, welches er meist in Form von vagen Anfragen stellt, die nicht präzise und formal genau das Informationsbedürfnis beschreiben. Das System muss Antworten auf diese vagen Fragen finden, wobei ihm meist Kenntnisse über die Inhalte der in ihm gespeicherten Dokumente fehlen. Dies kann zu fehlerhaften Antworten führen.
Information-Retrieval-Methoden werden im Rahmen der Sprachmodellierung verwendet, um verschiedene Informationen aus Daten zu extrahieren. Anwendungsbeispiele sind das Erkennen und Verfolgen des Themas eines Textes oder das Clustern eines Textkorpus, um einen großen allgemeinen Korpus sowie mehrere (kleinere) Adaptionskorpora zu erhalten, mit denen der Basiskorpus an verschiedene Themen angepasst werden kann.
In unserem Fall suchen wir aus einem riesigen Korpus (dem World Wide Web) zu einem Satz (der Übersetzungshypothese) passende Dokumente. Wir clustern das WWW also nach Übersetzungen. Diese Cluster können sich natürlich überschneiden. Für unser System brauchen wir eine Möglichkeit, die Ähnlichkeit einer Anfrage mit einem Dokument bestimmen zu können. Die ähnlichsten Dokumente werden dann in die Ergebnismenge oder Ergebnisliste aufgenommen. Wir werden nun häufig im IR verwendete Maße für den Erfolg einer Suche vorstellen sowie Maße für die Ähnlichkeit von Dokumenten.

3.1.1 Anfrageformulierung und Ergebnisberechnung

Unterschieden werden muss zwischen Anfrageformulierung und dem Modell, mit dem die Dokumente als Ergebnismenge berechnet werden (Ergebnisberechnung). Man muss vor allem unterscheiden zwischen der Boolschen Anfrageformulierung und dem Boolschen Modell zur Repräsentation natürlichsprachlicher Dokumente. Als erstes muss jedoch die Frage geklärt werden, was ein Dokument ist. Dies kann z.B. ein einzelner Satz, ein Absatz, eine Seite, ein Kapitel oder auch ein ganzes Buch sein.

Anfrageformulierung Es gibt mehrere Möglichkeiten, Anfragen zu formulieren. Die einfachste Art, Anfragen zu stellen, ist eine Menge an Wörtern anzugeben. Zusätzlich könnte die Anfrage weiter formalisiert werden, indem z.B. Boolsche Operatoren (AND, OR, NOT) verwendet werden oder Forderungen an die Reihenfolge gestellt werden. So könnte z.B. gefordert werden, dass zwei Wörter genau hintereinander im Dokument auftreten müssen „business card". In diesem Fall muss das Wort „card" dem Wort „business" folgen. Weitere Operatoren wären der NEAR-Operator, der angibt, dass ein Wort in der Nähe eines anderen Auftreten muss.

Modelle zur Repräsentation natürlichsprachlicher Dokumente Während der letzten Jahrzehnte sind verschiedene Modelle zur Repräsentation natürlichsprachlicher Dokumente entwickelt worden. Diese lassen sich unterteilen in

- **Mengentheoretische Modelle**, zu denen das (erweiterte) Boolesches-Retrieval-Modell und das Fuzzy-Retrieval-Modell gehören.

- **Algebraische Modelle**, zu denen das bekannte Vektorraummodell (Vector Space Model) gehört (in verschiedenen Ausprägungen) sowie der Latent-Semantic-Index und das Neuronale Netzwerk mit Backpropagation.

- **Probabilistische Retrieval-Modelle**, zu denen das Binary Independence Retrieval (BIR), Inferenz-Netzwek-Modell, Belief-Netzwerk-Modell und Sprachmodelle gehören.

Mengentheoretische Modelle betrachten natürlichsprachliche Dokumente als Mengen. Als Beispiel ist das Boolesches-Retrieval-Modell zu nennen. Jedes Wort, das in der Dokumentensammlung vorkommt, ist ein Boolsches Merkmal. Kommt es in einem Dokument vor, so ist das Merkmal bzgl. des Dokuments wahr, wenn nicht, so ist es falsch. Dies ist ein sehr einfaches Modell. Allerdings ist es für unsere Zwecke nicht geeignet, da es weder auf die Position eines Wortes achtet, noch eine Reihenfolge unter den Dokumenten in der Ergebnismenge erlaubt. Diese Eigenschaft, ein Ranking unter den Dokumenten aufstellen zu können, ist essentiell für unsere Zwecke.
Algebraische Modelle stellen Dokumente und Anfragen als algebraische Objekte dar, also z.B. als Matrizen oder Vektoren. Diese werden zur Berechnung der Ähnlichkeit mit Hilfe eines Ähnlichkeitsmaßes verwendet. Ein Beispiel für ein solches Maß ist TFIDF.
Probabilistische Retrieval-Modelle stützen sich zur Berechnung der Dokumentenähnlichkeit auf Wahrscheinlichkeiten und probabilistische Theoreme ab. Besonders häufig verwendet wird hier das Theorem von Bayes(2.2).

3.1.2 Maße für den Erfolg einer Suche

Den Erfolg einer Suche in einem Dokumentbestand kann man ermitteln, indem man die im Dokumentbestand enthaltenen Dokumente in Gruppen unterteilt und dann bestimmte Verhältnisse der Anzahl der Elemente dieser Gruppen betrachtet. Wir betrachten dabei 2 Dimensionen: Für jedes Dokument stellt sich zum einen die Frage, ob es vom System ausgewählt wurde oder nicht und zum anderen, ob es in der Menge der gesuchten Dokumente liegt oder eben nicht. Durch diese Unterteilung ergeben sich vier Gruppen von Dokumenten. Die vom System ausgewählten und in der Zielgruppe liegenden Dokumente bezeichnet man als „True-Positives" (tp). Die vom System ausgewählten, aber nicht in der Zielgruppe liegenden Dokumente werden als „False-Positives" (fp) bezeichnet. Die vom System nicht gefundenen Dokumente der Zielgruppe bezeichnet man als „False-Negatives" (fn), die vom System nicht ausgewählten, nicht in der Zielgruppe liegenden Dokumente als „True-Negatives" (tn). Dabei sind die „True-Positive" und die „True-Negatives" diejenigen Dokumente, die das System richtig bewertet hat. Basierend auf dieser Unterteilung lassen sich folgende Kenngrößen ableiten:

- Die **Präzision** (Precision) beschreibt den Anteil der vom System zurückgelieferten Dokumente, die richtig bewertet wurden: $precision = P = \frac{tp}{tp+fp}$. Dieses Maß bezieht sich auf die Menge der zurückgelieferten Dokumente. Es gibt an, wie präzise die Ergebnismenge bzgl. der Anfrage ist.

- Die **Ausbeute** (Recall) ist ein Maß, welches sich auf die Menge der Zieldokumente bezieht. Es beschreibt den Anteil der aus dieser Menge gefundenen Dokumente: $recall = R = \frac{tp}{tp+fn}$. Die Ausbeute gibt also die Ergiebigkeit einer Suche an.

- Das **F-Measure** ist ein Maß, welches sowohl Präzision als auch Ausbeute beachtet, indem es beide Maße in einem vereint. Es ist definiert als

$$F = \frac{1}{\alpha \frac{1}{P} + (1 - \alpha) \frac{1}{R}}.$$

  Präzision und Ausbeute sind Maße, welche sich im Intervall $[0, 1]$ bewegen. Da Präzision und Ausbeute nicht immer gleichzeitig maximiert werden können, muss ein Kompromiss in Kauf genommen werden. Das F-Measure stellt einen solchen Kompromiss dar.

In unserem Fall möchten wir vor allem eine hohe Präzision erlangen, also Dokumente finden, die gut auf unsere Übersetzungshypothese passen. Die Ausbeute ist weniger wichtig. Wir müssen nicht alle relevanten Dokumente finden, solange wir genügend Dokumente von ausreichend hoher Qualität für unseren Adaptionskorpus finden.

3.2 Relevanz - Gewichtungsmodelle und Retrievalfunktionen

Suchmaschinen müssen effizient die Relevanz eines Dokumentes bzgl. einer Anfrage bestimmen können. Damit unterscheiden sich Suchmaschinen - als Beispiel eines IR-Systems - von herkömmlichen SQL-Datenbanken, in denen ein Tupel entweder im Ergebnis einer Anfrage liegt oder nicht (binäre Entscheidung). Um dies umsetzen zu können, müssen die Dokumente durch geeignete Parameter beschrieben werden, mit deren Hilfe die Ähnlichkeit zu einer Anfrage beschrieben werden kann. Diese Parameter bilden den Deskriptor des Dokuments. Die Gewichtungsmodelle legen diese Parameter fest. Die Berechnung der Relevanz erfolgt dann über eine Retrievalfunktion. Ziel der Berechnung der Relevanz ist es, die Dokumente in eine Reihenfolge bzgl. ihrer Ähnlichkeit zur Anfrage zu bringen. Die in Suchmaschinen verwendeten Gewichtungsmodelle lassen sich in vektorraumbasierte Gewichtungsmodelle und hypermediabasierte Gewichtungsmodelle unterscheiden.

3.2.1 Vektorraummodell

Im Vektorraummodell (VRM) wird jedes Dokument durch einen n-dimensionalen Vektor beschrieben. Dabei wird jede Dimension durch ein Schlüsselwort aufgespannt, das das Dokument bezüglich seines Inhalts beschreibt. Die Schlüsselwörter werden durch einen Keyword-Relevanzfilter bestimmt. Die Anfrage wird ebenso als ein m-dimensionaler Vektor dargestellt. Man unterscheidet zwischen dem binären und dem gewichteten VRM. Im binären VRM wird jede Dimension entweder mit 1 oder 0 bewertet - je nachdem, ob das Schlüsselwort im Dokument enthalten ist oder nicht. Im mächtigeren gewichteten VRM hingegen kann jede Dimension durch eine positive oder negative Zahl dargestellt werden. Der Wert jeder Dimension gibt die Wichtigkeit, die einem Schlüsselwort zugeordnet wird, an. Der VRM macht keine Aussagen darüber, wie die Dokumentbeschreibung oder die Gewichtung zu erfolgen hat. Das VRM ist das am meisten genutzte Modell in Suchmaschinen. Dies beruht auf seiner Einfachheit und Schnelligkeit beim Retrieval. Es stellt sich nun die Frage, wie die Gewichte der Deskriptoren bestimmt werden. Diese Frage werden wir nun behandeln.

TFIDF Ziel ist es ein Maß zu finden, welches angibt, wie gut ein Term ein gegebenes Dokument charakterisiert. Die einfachste Variante wäre einfach die Anzahl der Vorkommnisse eines Terms in einem Dokument als Schätzwert für seine Wichtigkeit

im Dokument zu nehmen. Dieses Maß lässt aber die Länge des Dokumentes außer Acht und beachtet auch nicht, wie oft ein Term in anderen Dokumenten auftritt. Eine häufig genutzte Alternative ist das TF-IDF-Maß. Bei TF-IDF handelt es sich um eine Familie von Gewichtungsschemata. Ein Bestandteil von TF-IDF ist die Termhäufigkeit (term frequency;tf). Ihm liegt die Annahme zugrunde, dass ein Term ein Dokument umso besser beschreibt, je häufiger er im Dokument auftaucht. Meist wird die Termhäufigkeit von einer Funktion abgeschwächt, da der Zuwachs an Wichtigkeit nicht als linear angesehen wird. Es werden dafür z.B. folgende Formeln verwendet : $f(tf_{i,j}) = \sqrt{tf_{i,j}}$ oder $f(tf_{i,j}) = 1 + \log tf_{i,j}, tf_{i,j} > 0$. Dabei ist i der Term und j das Dokument. Der zweite Bestandteil des tf-idf-Maßes beruht auf der inversen Dokumenthäufigkeit (invers document frequency;idf). Sie ist abgeleitet von der Dokumenthäufigkeit (document frequency;df), welche die Anzahl der Dokumente angibt, in denen ein Term auftritt. Sie ist ein Indikator für die Aussagekraft eines Terms über ein Dokument. Tritt ein Term in vielen Dokumenten auf, charakterisiert er ein Dokument weniger gut als ein Term, der nur in wenigen Dokumenten auftritt. Die inverse Dokumenthäufigkeit berechnet sich nach der Formel $idf_i = \log(\frac{N}{df_i})$, $df_i > 0$. Es gibt unterschiedliche Möglichkeiten, die beiden Bestandteile des TF-IDF-Maßes zu kombinieren. Deshalb handelt es sich um eine Familie von Maßen. Eine Möglichkeit wäre die Verwendung von Logarithmusfunktionen:

$$\text{weight}(i,j) = \begin{cases} (1 + \log tf_{i,j}) \log \frac{N}{df_i} & \text{if } tf_{i,j} \geq 1 \\ 0 & \text{if } tf_{i,j} \geq 0 \end{cases}$$

Die TFIDF-Maße haben sich in der Praxis als effektiv und robust erwiesen. Deshalb werden sie oft benutzt, obwohl sie nicht direkt aus einem mathematischen Termverteilungsmodell abgeleitet sind.

Okapi Die Okapi-TF-Formel ist von einem probabilistischen Modell abgeleitet [27]. Sie basiert auf zwei Parametern k_1 und b. Sei $tf_{i,j}$ die absolute Häufigkeit des Auftretens von term i in Dokument j. Im Okapi-Gewichtungsschema (im Lemur-Toolkit [37]) sind dann die Funktion $td^d_{i,j}$ für Dokumente und die Funktion $td^q_{i,j}$ für die Anfrage folgendermaßen definiert:

$$tf^d_{i,j} = \frac{k_1 tf_{i,j}}{tf_{i,j} + k^d_1(1 - b_d + b_d \frac{l_d}{l_C})} \tag{12}$$

$$tf^q_i = \frac{k_1 tf_i}{tf_i + k^q_1(1 - b_q + b_q \frac{l_q}{l_C})} \tag{13}$$

Hierbei steht l_d für die Länge des Dokumentes d und l_C für die Länge der Dokumentensammlung C. Der Parameter l_q wurde im Lemur-Toolkit eingeführt und beschreibt die durchschnittliche Anfragelänge. Um die Konstanten (die verschiedenen k und b) zu spezifizieren, wird das Bezeichnungsschema $BMxx$ verwendet, wobei xx eine zweistellige Zahl darstellt und BM dabei für „BestMatch" steht.
Der Score für ein Dokument j bzgl. einer Anfrage q ergibt sich dann zu

$$score(j,q) = \sum_{i=1}^{n} tf^d_{i,j} tf^q_i idf^2_i$$

Für weitere Details verweisen wir auf [27] und [37].

3.2.2 Hypermediabasierte Gewichtungsmodelle

Aufgrund der ständig wachsenden weltweiten Verpflechtung von Dokumenten mittels des Internets und des WWW sind Verfahren entstanden, die die spezielle Struktur dieses vernetzten Hyperraumes beachten. Es gibt zwei Verfahrensklassen.
Zur ersten Klasse gehören das PageRank-Verfahren der Suchmaschine Google und das Link-Popularity-Verfahren, das von anderen Suchmaschinen eingesetzt wird. Diese Verfahren bewerten die Anzahl und die Qualität der Links, die auf ein Dokument verweisen. Zur zweiten Klasse der hypermediabasierten Verfahren gehört das Click-Popularity-Verfahren. Es führt eine Bewertung von Dokumenten basierend auf der Häufigkeit, mit der ein Dokument aus der Suchergebnisliste aufgerufen wird und der Verweildauer auf dieser Seite. Obwohl es von einigen Suchmaschinen eingesetzt wurde, hat es sich nie richtig durchgesetzt. Beide Verfahrensklassen zeichnen sich dadurch aus, dass sie aufgrund der Wichtigkeit, auf einer Ergebnisliste möglichst weit oben zu stehen, häufig mit Manipulationsversuchen umgehen müssen. Auch ist für keines dieser Verfahren der Öffentlichkeit der exakte Algorithmus bekannt. Diese Algorithmen sind das Kapital einer Suchmaschine und werden streng geheim gehalten. Die hypermediabasierten Verfahren sind nicht die einzigen Verfahren, die in Suchmaschinen eingesetzt werden. Durch sie wurde eine weitere Dimension für das Information-Retrieval eingeführt. Neben den hypermediabasierten Verfahren kommen auch andere Verfahren, wie z.B. das Vektorraummodell zum Einsatz. Die Verfahren werden außerdem ständig den sich ändernden Bedingungen des WWW angepasst.

Das PageRank-Verfahren von Google Das PageRank-Verfahren ([6][16]) ist das dominierende Verfahren bei Google. Neben ihm werden noch andere Verfahren, wie das Term-Frequency-Verfahren und die differentierte Bewertung der Position von Schlüsselwörtern angewendet.
Das PageRank-Verfahren ist ein iteratives Verfahren. Für jedes Dokument, das ein Suchwort als Deskriptor enthält, wird eine initiale Bewertung mit Hilfe von Verfahren berechnet, die nur mit Charakteristika des Dokumentes arbeiten (wie z.B. dem TF- oder dem IDF-Verfahren). Ausgehend von dieser initialen Bewertung wird der Wert der ausgehenden Verweise eines Dokumentes berechnet. Die Qualität der Links ist ausschlaggebend für die Bewertung der Zieldokumente des Links. Je höher die Qualität des Quelldokuments und je weniger Links von ihm wegführen, desto höher ist der Wert des Links. Auch Links von Quellen, die Google explizit als hochwertig bekannt sind (wie z.B. der Katalog von Yahoo) bekommen ein besonderes Gewicht. Basierend auf diesen Linkwerten wird für jedes Dokument t der PageRank $PR(t)$ nach folgender Formel berechnet:

$$PR(t) = (1-d) + d(\frac{PR(s_1)}{w(s_1)}) + \ldots + \frac{PR(s_n)}{w(s_n)})) \tag{14}$$

Wie man aus der Gleichung erkennt, ist dieses Verfahren ein iteratives Verfahren. Für die Berechnung des PageRanks für das komplette WWW werden von den Erfindern des Algorithmus ca. 100 Iterationen als hinreichend genannt. Das PageRank-Verfahren ist in der Praxis sehr erfolgreich und hat Google einen enormen Vorsprung gegenüber anderen Suchmaschinen eingebracht.

Click-Popularity Das Verfahren der Click-Popularität, das sich nicht richtig durchgesetzt hat, basiert auf der Annahme, dass Dokumente aus Ergebnislisten, die von Benutzern einer Suchmaschine aufgerufen werden, relevanter für die Anfrage sind als

andere. Deshalb werden die Anzahl der Clicks auf eine URL aus einer Ergebnisliste gespeichert. Um neue Dokumente im Datenbestand nicht zu benachteiligen, wird auch die Verweildauer der Daten im Datenbestand berücksichtigt. Da sich durch mehr Clicks die Position des Dokuments in der Ergebnisliste verbessert, ist dieses Verfahren anfällig für Manipulationsversuche. Gegenmaßnahmen bestehen z.B. in einer Identifikation der aufrufenden URLs, um z.B. automatisches Aufrufen der Links aufzudecken und nicht in die Bewertung mit einzubeziehen.

3.2.3 Retrievalfunktionen

Die Retrievalfunktionen geben an, wie die Berechnung der Ähnlichkeit zwischen den Dokumentenvektoren und der Anfrage zu erfolgen hat. Für die Berechnung der Ähnlichkeit zwischen zwei Vektoren wird häufig das Cosinusmaß benutzt.

$$\cos(q,d) = \frac{\sum_{i=1}^{n} q_i d_i}{\sqrt{\sum_{i=1}^{n} q_i^2}\sqrt{\sum_{i=1}^{n} d_i^2}}. \tag{15}$$

Das Cosinusmaß normalisiert die Vektoren. Diesem Maß unterliegt die Annahme, dass semantische Nähe mit räumlicher Nähe korreliert.

3.3 Suchmaschinen

Eine Suchmaschine ist ein Programm zum Finden von Dokumenten innerhalb einer Dokumentensammlung, die bzgl. einer Anfrage eine gewisse Relevanz aufweisen. Die Dokumente können lokal auf einem Rechner gespeichert sein oder auf Rechnern eines Rechnernetzes, oder wie im Falle des Internets über ein Netz von Rechnernetzen verteilt vorliegen. Je nach dem Ort, an dem die Daten vorliegen, unterscheidet man zwischen Internetsuchmaschinen, Intranetsuchmaschinen und Desktopsuchmaschinen. Desktopsuchmaschinen durchsuchen den lokalen Datenbestand eines Computers. Internetsuchmaschinen haben die Aufgabe, relevante Dokumente im Internet zu finden. Da es nicht möglich ist, das Internet für jede Anfrage in Echtzeit zu durchsuchen (zu geringe Bandbreite der Netzverbindungen), muss dies offline geschehen. Dazu durchsuchen sogenannte Webcrawler - auch Spider oder Robots (bots) genannt - das WWW, indem sie von gewissen Einstiegspunkten aus allen Links folgen. Dabei können sie natürlich nur die Zusammenhangskomponenten des Internets durchsuchen, zu denen die Einstiegspunkte gehören. Die Frage, wie man den restlichen Teil des Internets finden und durchsuchen kann, ist ein offenes Forschungsgebiet. Um die Daten aktuell zu halten, muss dieser Prozess von Zeit zu Zeit wiederholt werden. Das Wiederholungsintervall eines solchen Webscans hat dabei entscheidenden Einfluss auf die Aktualität der Daten.
Ein weiteres Unterscheidungsmerkmal für Suchmaschinen ist die Art der Daten, die in der Suchmaschine verarbeitet werden. Hier kann ganz allgemein zwischen Text-, Bild-, Ton- und Videodokumenten unterschieden werden. Einige Suchmaschinen spezialisieren sich auch auf spezielle Anwendungsgebiete, wie z.B. wissenschaftliche Fachartikel, Usenet-Beiträge oder eine Einschränkung auf eine spezielle Zielgruppe, wie z.B. Familien oder Studenten. Viele Suchmaschinen bieten auch eine Einschränkung der betrachteten Dokumente unabhängig von der Anfrage an. Eine solche Einschränkung kann z.B. die Beschränkung auf einen bestimmten Dateityp (pdf, ps, doc, PowerPoint-Folien), auf eine bestimmte Sprache, Daten aus einem bestimmten Land, einen Filter für Daten, die den Benutzer belästigen könnten (z.B. wegen sexuellen Inhalts) oder bei

der Internetsuche der Domainname sein.
Ein Spezialfall der Suchmaschinen sind die Metasuchmaschinen. Metasuchmaschinen setzen ihr Suchergebnis aus den Suchergebnissen mehrerer fremder Suchmaschinen zusammen. Damit decken sie natürlich potentiell eine größere Menge an Dokumenten ab. Es stellt sich allerdings die Frage, wie die Ergebnisse der fremden Suchmaschinen bewertet und relativ zueinander angeordnet werden können. Ein weiterer Nachteil sind die längeren Antwortzeiten. Wenn diese tolerierbar sind, lassen sich Metasuchmaschinen gut bei der Suche nach seltenen Begriffen einsetzen.
Eine andere Form, Webinhalte aufbereitet anzubieten, ist der Katalog, auch Verzeichnis(Directory) genannt. Hier erfolgt eine Anordnung nach Themen, die der Benutzer immer weiter verfeinern kann. Sie sind durch die Ersteller des Katalogs vorgegeben. Eine Aufnahme in den Katalog erfolgt nach Anmeldung und Bewertung der Dokumente durch Lektoren.
Viele Anbieter von Websuchmaschinen bieten auch andere Dienste an, welche häufig auf derselben Website wie die Suchmaschine angeboten werden (z.B. Google, Yahoo). Neben der allgemeinen Suchfunktion der Suchmaschine werden dort die Suche nach Jobs, Nachrichten, die Suche nach Büchern, Einkaufsmöglichkeiten und vielem mehr angeboten.
Die Aufgaben einer Suchmaschine lassen sich in drei Aufgabenbereiche unterteilen. Dies sind der Aufbau einer Datenstruktur zur effizienten Suche nach Dokumenten, die Verarbeitung von Suchanfragen und die Präsentation der Ergebnisse. Wir werden sie in den folgenden drei Abschnitten vorstellen.

3.3.1 Aufbau einer Datenstruktur zur effizienten Suche nach Dokumenten

Um die relevanten Dokumente bzgl. einer Anfrage zu ermitteln, arbeitet die Suchmaschine aus Effizienzgründen nicht direkt auf den Dokumenten, sondern auf einer Datenstruktur, welche Daten über die einzelnen Dokumente zur Verfügung stellt. Die Daten liegen in der Regel in unstrukturierter Form als Texte vor. Es gibt aber auch semistrukturierte Dokumente, wie z.B. HTML- oder XML-Dokumente. Suchmaschinen gehen aber von Dokumenten mit keiner oder nur sehr geringer Struktur aus.

Suchindex und invertierte Datei Die oben erwähnte Datenstruktur heißt Index und enhält alle relevanten Terme der Dokumentensammlung, nach denen gesucht werden kann. Zu jedem Eintrag im Index existiert eine invertierte Datei. Sie gibt für jedes Wort aus der Dokumentensammlung an, in welchem Dokument das Wort auftritt und wie oft (Gewicht).

Gewichte Das Gewicht eines Eintrags ist eine wichtige Information, auf deren Basis die Relevanz eines Dokuments berechnet werden kann. Neben der Häufigkeitsinformation wird in einigen Indizes auch Information über die Position der Wörter in den einzelnen Dokumenten gespeichert. Die Positionsinformation kann dazu verwendet werden, auch Information über Phrasen aus dem Index abzuleiten.

Beispiel für einen Suchindex Der Index enthält die Einträge, nach denen gesucht werden kann, also die einzelnen Wörter und zu jedem Wort einen Verweis auf die zugehörige invertierte Datei. Hier stellen wir den Index und die invertierte Datei für das Suchwort *Sprachmodell* vor:

Wort	Verweis (ID) auf invertierte Datei
Sprachmodell	001
Übersetzungssystem	023

Tabelle 1: Einfacher Index

Invertierte Datei 001			
DocID	Position	Häufigkeit	Gewicht
1234	1,13,75, 200, 343	5	6,7
2345	23, 245, 357, 432	4	3,4
...	...	...	...
3456	2, 456, 567	3	7,8

Tabelle 2: invertierte Datei für Suchbegriff *Sprachmodell*

Phrasen Eine Phrase ist eine Ansammlung von Wörtern, deren Reihenfolge meist wichtig ist. Es kann aber durchaus sinnvoll sein, diese strikte Definition ein wenig aufzuweichen und bei der Suche nach der Phrase „admission fee" auch die Phrase „fee for admission" als äquivalente Ausprägung zuzulassen. In vielen IR-Systemen werden Module zur Identifizierung von Phrasen verwendet und die identifizierten Phrasen dann wie einzelne Wörter in den Index aufgenommen.

Stopwords Bei einigen Suchmaschinen werden bestimmte Wörter - die sogenannten Stopwords - nicht mit in den Index aufgenommen. Stopwords sind Wörter, die relativ häufig in Dokumenten auftreten und das Dokument deshalb nicht besonders stark charakterisieren. Beispiele für Stopwords im Englischen wären: *a, can, have, i, my on, it* und *go*. Der Vorteil liegt darin, das sich die Größe des Indexes stark verkleinern kann. Allerdings sind die Stopwords für die Suche nach exakten Phrasen wichtig. Deshalb verwenden viele Suchmaschinen keine Stopwords.

Stemming Eine andere Technik, die bei Suchmaschinen eingesetzt werden kann, ist das Stemming. Stemming ist eine einfache Form der morphologischen Analyse, bei der die Endungen von Wörtern abgeschnitten werden und so eine Klassenbildung erfolgt. Zum Beispiel würden die Wörter going, goes und go alle in die Klasse „go" fallen. Eine Anfrage nach „go" würde also auch Ergebnisse mit „goes" zurückliefern. Dies kann durchaus im Interesse des Nutzers sein. Probleme treten in morphologisch reichen Sprachen auf, wenn Wörter aus verschiedenen semantischen Klassen auf den gleichen Stamm reduziert werden, wie z.B. die Wörter Bildung und Bild. Ein bekannter und häufig verwendeter Stemmer ist der Porter-Stemmer.
Ein wesentlicher Unterschied von Suchmaschinen zu Datenbanksystemen ist, dass sich die Ergebnisse einer Suchanfrage bei Suchmaschinen bezüglich ihrer Relevanz zur Anfrage unterscheiden. Bei Datenbanksystemen hingegen sind alle Ergebnistupel gleich relevant (vgl. Abschnitt 3.2).
Neben dem Index, der verwendet wird, um die Dokumente effizienter durchsuchen zu können, wird von den Suchmaschinen nicht direkt auf den Dokumenten gesucht, sondern auf einer kompakten Beschreibung der Dokumente, die die Dokumente *inhaltlich* beschreiben. Hierzu werden Keywordrelevanzfilter eingesetzt und nicht unbedingt alle Teile eines Dokumentes verwendet. Ebenfalls werden einige Begriffe stärker bewertet, je nachdem, an welcher Position im Dokument sie auftreten (z.B. im HEAD-Tag

im Falle eines HTML-Dokuments). Die Schwierigkeit besteht darin, automatisch zu entscheiden, welche Elemente eines Dokuments dieses aussagekräftig repräsentieren. Dabei sollten die Deskriptoren so gewählt werden, dass alle zu einem Thema gehörigen Dokumente gefunden werden können (Recall 3.1.2) und möglichst auch nur diese (Präzision 3.1.2). Die Deskriptoren dienen der Relevanzbestimmung eines Dokumentes bezüglich einer Anfrage und der Verknüpfung von Dokumenten, die thematisch ähnlich sind.

3.3.2 Verarbeitung von Suchanfragen

Um ein Suchergebnis zu einer Anfrage liefern zu können, muss die Relevanz der Dokumente der Dokumentensammlung bezüglich der Anfrage von dem Query-Processor der Suchmaschine bestimmt werden. Der Query-Processor ist die eigentliche Suchkomponente der Suchmaschine. Mit Hilfe der Retrievalfunktionen bestimmt er die Relevanz der in einem oder mehreren Gewichtungsmodellen dargestellten Dokumenten. Dies kann sowohl auf rein syntaktischer Ebene, als auch auf semantischer Ebene geschehen. So sollten z.B. zu einer Anfrage nach „Fahrrad" auch Dokumente mit dem Wort „Zweirad" zurückgeliefert werden. Neben dem Auffinden von Synonymen kann die Anfrage auch einer Rechtschreibkorrektur unterzogen werden. Bei der Anfrageauswertung kann auch Metainformation der Dokumente verwendet werden. Die theoretischen Grundlagen bezüglich Gewichtungsmodellen (Abschnitt 3.2) und Retrievalfunktionen (Abschnitt 3.2.3) wurden bereits dargestellt. Bei Internetsuchmaschinen wird häufig eine Kombination aus verschiedenen Gewichtungsmodellen verwendet. Dabei gehen sowohl Vektorraummodelle (Abschnitt 3.2.1) als auch hypermediabasierte Gewichtungsmodelle (Abschnitt 3.2.2) in die Berechnung des Ranges eines Dokumentes bezüglich einer Anfrage mit ein.

Boolsche Operatoren Die Internetsuchmaschinen bieten dem Anwender verschiedene Operatoren zur Anfrageformulierung an. Darunter befinden sich die Boolschen Operatoren. Diese bestehen aus dem AND-Operator, der Dokumente zurückliefert, für die seine beiden Operanden als wahr evaluiert werden sowie der OR-Operator, bei dem einer seiner beiden Operanden zu wahr evaluiert werden muss. Mit dem NOT-Operator kann definiert werden, dass eine Bedingung nicht erfüllt sein darf. Diese Operatoren können miteinander kombiniert werden. Auf unterster Ebene entscheidet die Existenz bzw. Nichtexistenz eines Wortes über die Evaluierung zu wahr oder falsch.

Weitere Operatoren und Optionen Weitere Operatoren erlauben die Suche nach genauen Phrasen. Es werden Optionen angeboten, mit deren Hilfe die Suche weiter eingeschränkt werden kann, z.B. kann sie auf bestimmte Bereiche der Dokumente, auf bestimmte URLs oder Domains, Dokumentformate, Sprachen und Datumszeiträume beschränkt werden.
Das Ergebnis einer Suchanfrage entsteht also aus dem Zusammenspiel von mehreren Komponenten. Durch das Information-Retrieval-System einer Suchmaschine werden die Dokumente dargestellt und ihnen werden Gewichte zugeordnet. Durch Retrievalfunktionen wird die Berechnung von Relevanzwerten definiert. Der Query-Processor einer Suchmaschine führt dann die Suche nach den relevanten Dokumenten basierend auf Dokumentbeschreibungen und Retrievalfunktionen (und eventueller Kombination dieser) durch.

3.3.3 Präsentation der Ergebnisse

Die Dokumente der Ergebnisliste können auf unterschiedliche Weise dargestellt werden. Der einfachste Fall wäre, die Dokumente als eine nach dem Rang der Dokumente sortierte Liste darzustellen. Die meisten Internet-Suchmaschinen stellen ihre Ergebnisse als geordnete Liste dar. Einige Suchmaschinen erweitern diese Art der Darstellung, indem sie Dokumente, die zu den Dokumenten in der Liste ähnlich sind, nur bei Bedarf anzeigen. Dies gibt dem Benutzer eine bessere Übersicht. Eine andere Art der Darstellung ist die Baumdarstellung, in der Dokumente entweder in vordefinierte Klassen eingeordnet werden (Classification) oder durch ein automatisches Verfahren Dokumentenklassen zugeordnet werden (Clustering). Von nahezu allen Internetsuchmaschinen wird für die gesamte Liste die Anzahl der gefundenen relevanten Dokumente angegeben. Eine Angabe des von der Suchmaschine berechneten Rangwerts je Dokument gibt es jedoch nicht. Lediglich die Rangposition ist dem Nutzer bekannt.

Abbildung 2: Ergebnisliste von Google

Beispielantwortseite von Google In Abbildung 2 wird eine Antwortseite der Suchmaschine Google dargestellt. Hier werden die gefundenen Dokumente als geordnete Liste dargestellt, wobei das Ordnungskriterium der Relevanzwert ist. Dieser wird jedoch nicht mit angegeben. Für jedes Dokument wird eine Überschrift und ein Auszug aus dem Dokument angezeigt, in dem die gesuchten Wörter hervorgehoben sind. Falls das Dokumentformat nicht HTML ist, wird dieses angegeben und eine Konvertierung nach HTML angeboten. Abschliessend wird die URL angezeigt und ein Links zu ähnlichen Seiten angeboten. Für HTML-Seiten wird zusätzlich die Größe der Datei angegeben, sowie ein Link zu einer gecachten Kopie der Datei. Auf der rechten Seite werden noch zusätzlich Links angeboten, für deren Position bezahlt wurde. Als Zusatzinformation über alle Dokumente werden die Anzahl der gefundenen Dokumente angegeben, sowie die Zeit, die zur Erzeugung des Anfrageergebnisses benötigt wurde.

4 Sprachmodelladaption mit Hilfe von Information Retrieval und WWW

Nach der Darstellung der theoretischen Grundlagen in den vorangegangenen Kapiteln werden wir in diesem Kapitel noch in die Sprachmodelladaption einführen (Abschnitt 4.1), unserer Arbeit vorausgegangene Arbeiten vorstellen (Abschnitt 4.2) und dann unsere Adaptionsmethode beschreiben (Abschnitt 4.3). Dabei soll hier die Methode beschrieben werden. Die Experimente, sowie die Ergebnisse werden dann in Kapitel 5 präsentiert.

4.1 Sprachmodelladaption

Im Laufe der Forschungen zur Verarbeitung natürlicher Sprache, besonders in den Bereichen Spracherkennung und maschineller Übersetzung, hat sich gezeigt, dass ein sprachverarbeitendes System umso besser arbeitet, je stärker es auf den Anwendungsbereich zugeschnitten ist. In einer bestimmten Domäne trainierte Systeme übersetzen Texte aus dieser Domäne besser als dies ein System kann, welches auf allgemeinen Daten trainiert wurde. Das auf allgemeinen Daten trainierte System bringt jedoch eine bessere Übersetzungsleistung als das spezifische System, wenn es sich um einen out-of-domain-Text bzgl. des spezifischen Systems handelt.
Sprachmodelladaption wird durch folgende Tatsachen motiviert: Die oft verwendeten n-gram-Sprachmodelle behalten keine langfristigen Kontextinformationen. Dies ist ungünstig, da sich Eigenschaften eines Textes wie Thema, Stil, und Domäne im Laufe eines Textes ändern können. Um diesem Problem entgegenzuwirken, kann man das Sprachmodell dynamisch anpassen. Wir unterscheiden zwei Arten, Kontextinformation zu extrahieren. Zum einen kann Kontextinformation aus der Umgebung der zu übersetzenden Phrase ermittelt werden, zum anderen aus einer bereits vorliegenden Übersetzungshypothese. Bei der letzten Art besteht die Gefahr, Übersetzungsfehlern zu starken Einfluss zu gewähren. Auf Basis von Kontextinformationen können aus einem großen Textkorpus geeignete Adaptionsdaten ausgewählt werden.
Die Sprachmodelladaption kann in zwei Kernaufgaben unterteilt werden. Zum einen muss zuverlässige Kontetxinformation ermittelt werden. Zum anderen muss das Sprachmodell basierend auf den ermittelten Kontextinformationen angepasst werden.

4.1.1 Ermittlung von Kontextinformation und Adaption des Sprachmodells

Zur Extraktion von Kontextinformation können Methoden des Information-Retrievals genutzt werden. Es existieren Algorithmen zur Detektion von verschiedenen Themen und deren Verfolgung (Tracking) im Text. Dabei kann zwischen Algorithmen unterschieden werden, die eine fest vorgegebene Menge an Themen kennen und jeden zu klassifizierenden Text einem Thema oder einer Kombination von Themen zuordnen (Klassifikation), und denen, die zu klassifizierende Texte gruppieren, ohne eine fest vorgegebene Menge an Themen zu kennen (Clustering). Folgende Gegebenheiten machen die Sprachadaption schwer: Zum einen kann ein Text nicht immer genau einem Thema zugeordnet werden. Zum anderen kann man nicht wissen, wie die Testdaten aussehen. Es können bis dato unbekannte Domänen auftreten, weshalb Adaptionsdaten online ermittelt werden müssen. Desweiteren kann es große Unterschiede zwischen den Trainings- und den Testdaten geben.

Mixture-basierte Sprachmodelle Bei mixturebasierten Sprachmodellen wird der Trainingskorpus auf k verschiedene Themen aufgeteilt. Dabei kann eine Phrase durchaus mehreren Themen zugeordnet werden. Für jedes Thema wird dann ein Sprachmodell L_j erzeugt und diese Sprachmodelle werden später interpoliert. Die Gewichte λ_j können mit Hilfe des EM-Algorithmus ermittelt werden.

$$P(t_i|h_i) = \sum_{j=1}^{k} \lambda_j P_{L_j}(t_i|h_i) \tag{16}$$

k ist ein Parameter, der vorgegeben werden muss. Ein großes k ermöglicht dabei eine Vielzahl an Themen und damit eine bessere Adaption. Es muss aber darauf geachtet werden, dass genügend Daten vorliegen, um die Parameter der einzelnen Sprachmodelle L_j robust schätzen zu können. Ein zu kleines k lässt nur schwammige Themen entstehen und verhindert damit eine gute Einteilung und Übersetzung.

4.2 Vorausgegangene Arbeiten

4.2.1 Überblick

Es gibt verschiedene Ansätze, das Sprachmodell an sich ändernde Gegebenheiten wie z.B. Kontext oder Stil anzupassen. Es wurden hierzu bereits Arbeiten in den Bereichen automatische Spracherkennung und maschinelles Übersetzen durchgeführt. Dabei wurden verschiedene Techniken entwickelt, die aber je nach Bereich unterschiedlich gut funktionieren. Janiszek [20] (nach [41]) gibt folgenden Überblick über Sprachmodelladaptionsansätze:

- Lineare Interpolation eines allgemeinen und eines domänenspezifischen Sprachmodells [30].
- Back-Off von domänenspezifischen Wahrscheinlickeiten mit denen eines spezifischen Sprachmodells [4].
- Retrieval von Dokumenten einer geänderten Domäne und Onlinekonstruktion eines neuen Sprachmodell mit Hilfe der gefundenen Dokumente [18] [24].
- Maximum-Entropie, Minimum-Discrimination-Adaption [8]
- Adaption durch lineare Transformation von Bigramm-Counts in einen reduzierten Raum [11].
- Smoothing in einem dualen Raum durch Latent-Semantic-Analysis, Modellierung von langfristigen semantischen Abhängigkeiten und Triggerkombinationen [3].

4.2.2 Sprachmodelladaption via Information-Retrieval

Eck, Vogel und Waibel [14] verfolgen den Ansatz, die Sprachmodelladaption für statistische Übersetzungssysteme auf Satzbasis durchführen. Die Autoren übertragen den für Spracherkennungssysteme erfolgreichen Ansatz auf statistische Übersetzungssysteme und arbeiten dabei auf Satz- statt auf Story-Level. Ähnliche Sätze wurden dabei mit Hilfe des Gewichtungsschemas TFIDF und dem Kosinusähnlichkeitsmaß ermittelt. Dabei konnte die Perplexität deutlich verringert werden und in einem Fall auch die

Übersetzungsqualität verbessert werden.
Eine darauf aufbauende Arbeit von Zhao, Eck und Vogel [41] stellt Bag-Of-Words-Anfragen und strukturierte Anfragen an den Basiskorpus, um auf Satzbasis einen Adaptionskorpus aufzubauen. Hierbei werden verschiedene Strategien zur Generierung von Anfragen verwendet. Dabei wird die Nähe von Termen beachtet, Terme unterschiedlich gewichtet oder die Informationen aus N-Best-Liste und Übersetzungsmodell verwendet. Für automatische Spracherkenner stellen Mahajan, Beeferman und Huang [24] eine Methode vor, die mit Techniken des Information-Retrievals vorhandene Kontextinformationen generalisiert und zeigen, dass sie mit ihrer Methode die Perplexität vermindern können. Dabei verwenden sie das Gewichtungsschema TFIDF, um Dokumente zu bewerten und das Kosinusmaß um Ähnlichkeit zwischen den Dokumenten und der Anfrage zu definieren. Es werden immer eine feste Anzahl an Dokumenten als Basis für den Adaptionskorpus ausgewählt, aus dem dann das Adaptionssprachmodell erzeugt wird.

4.2.3 Sprachmodelladaption via WWW

Es existieren verschiedene Ansätze, das WWW zur Sprachmodelladaption zu nutzen. Der Vorteil des WWW gegenüber einem lokal vorliegenden Korpus besteht darin, dass das WWW wie die natürliche Sprache dynamisch ist und Änderungen in der Sprache aufnimmt. Weiterhin gibt es für viele Spezialgebiete genügend Ressourcen im Netz, was in einem lokalen Korpus nicht in dem Ausmaß der Fall ist. Zhu und Rosenfeld [42] verwenden das WWW, um die Wahrscheinlichkeiten für ein Trigramsprachmodell eines automatischen Spracherkenners zu schätzen. Dabei stellen sie Trigramanfragen an eine Suchmaschine und verwenden die Anzahl an gefundenen Webseiten/Dokumenten $c_{web}(w_1 w_2 w_3)$ als Schätzwert der Häufigkeit des Trigramme im WWW. Der Schätzwert berechnet sich dann nach folgender Formel:

$$p_{web}(w_3|w_1 w_2) = \frac{c_{web}(w_1 w_2 w_3)}{c_{web}(w_1 w_2)}. \tag{17}$$

Die Autoren interpolieren dieses Websprachmodell mit einem Sprachmodell aus einem lokal vorliegenden Korpus und berichten über signifikante Verbesserungen der WER bei Spracherkennungsaufgaben. Ghani, Jones und Mladenic [15] beschreiben einen Ansatz für die automatische Generierung von Anfragen an Websuchmaschinen, um Dokumente in Sprachen mit wenig verfügbaren Ressourcen (minority languages) zu finden und somit Korpora in diesen Sprachen aufzubauen. Le et al. in [21] erweitern diesen Ansatz und erstellen Sprachmodelle für Sprachen mit wenig verfügbaren Ressourcen im Rahmen von automatischer Spracherkennung.
O'Neil und French [25] beschreiben den Umgang mit Webdaten und Internetsuchmaschinen im Rahmen ihres *Language-Model-Builders*, wobei sie unter Sprachmodell allgemein eine Einheit verstehen, die eine Sammlung von Texten zusammenfasst und beschreibt. Dieses Sprachmodell wird verwendet, um unter verschiedenen Datenbanken die geeignetste auszuwählen.

4.3 Adaptionsprozess

Die Untersuchungen in der vorliegenden Arbeit beschäftigen sich mit der Frage, wie durch eine geeignete Anpassung des Sprachmodells an einem zu übersetzenden Satz das Übersetzungsergebnis verbessert werden kann. Dazu übersetzen wir den zu übersetzenden Satz mit dem CMU-SMT-Toolkit unter Verwendung eines Baselinesprach-

modells und suchen dann zu einer aus dieser ersten Übersetzung (first pass translation; 1PT) erzeugten Anfrage passende Texte. Aus diesen Texten erzeugen wir ein Sprachmodell (Adaptionssprachmodell). Das Adaptionssprachmodell verwenden wir als weiteres Modell im Dekoder. Der Dekoder und das Übersetzungsmodell werden dabei nicht verändert. Wir betrachten das Übersetzungssystem also als Black Box, zu dem wir nur das Adaptionssprachmodell hinzufügen.
Der entscheidende Schritt bei diesem Vorgehen ist die Auswahl der Daten für das Adaptionssprachmodell. Dazu verwenden wir IR-Methoden.

Die sechs Schritte des Adaptionsprozesses

1. Im **Übersetzungsschritt** wird mit Hilfe des Baseline-Systems eine erste Übersetzung des Satzes der Quellsprache erzeugt.

2. Mit Hilfe dieser Übersetzung wird dann eine Anfrage (Query) für eine Suchmaschine im Internet erzeugt **(Query-Generierung)**.

3. Im **Preprocessingschritt** werden die von der Suchmaschine gelieferten Dokumente aus dem Internet heruntergeladen und dann einer Vorverarbeitung unterzogen. Hierbei werden nicht gewünschte Bestandteile innerhalb der Dokumente entfernt.

4. Diese gesäuberten Dokumente werden dann im **Bewertungs- und Auswahlschritt** bewertet und auf Grund der Bewertung wird eine Auswahl aus der Menge der Dokumente getroffen, welche die Basis für das Adaptionssprachmodell bildet.

5. Im **Adaptionsschritt** wird ein neues Sprachmodell aus den Adaptionsdaten - das Adaptionssprachmodell - gebildet oder alternativ ein Sprachmodell aus verschiedenen Adaptionssprachmodellen interpoliert.

6. Danach wird im letzten Schritt (Erzeugen der finalen Übersetzung) der zu übersetzende Satz erneut übersetzt (second pass translation;2PT). Diese Übersetzung sollte besser sein als die erste, da das Sprachmodell besser auf den zu übersetzenden Satz zugeschnitten ist.

Im folgenden werden die einzelnen Schritte detailliert beschrieben:

4.3.1 Erzeugung der ersten und der finalen Übersetzung

Für den ersten und letzten Schritt wird das CMU-SMT-Toolkit verwendet. Dessen Funktionsweise und Modelle werden im Rahmen dieser Arbeit nicht verändert. Es wird lediglich ein zusätzliches Sprachmodell hinzugefügt. In einem zusätzlichen Versuch optimieren wir auch die Gewichte der einzelnen Modelle, inklusive des Gewichts des zusätzlichen Sprachmodells.

4.3.2 Query-Generierung

Zur Query-Generierung verwenden wir die erste Übersetzung (1PT) und alternativ die N-Best-Liste. Dabei erzeugen wir pro Satz verschiedene Anfragen. Unter diesen Anfragetypen befinden sich zwei Bag-of-Words-Anfragen (**BoW**) und eine Anfrage, bei der die Reihenfolge der Wörter eine Rolle spielt (**Ngram**). In beiden Fällen erzeugen wir

Anfragen, die Stopwörter enthalten (+S) und solche ohne Stopwörter (-S). Stopwörter sind Wörter, die so häufig in Dokumenten auftreten, dass sie das Dokument nicht besonders stark charakterisieren. Als Beispiele sind in der deutschen Sprache die Wörter „und", die bestimmten und unbestimmten Artikel oder das Wort „ist" zu nennen. In der ersten Bag-of-Words-Anfrage (**BoW**) werden alle Wörter der 1PT verwendet. Die zweite BoW-Anfrage (**BoW-Maj**) verwendet als Basis die n besten Hypothesen (N-Best-Liste) und benutzt alle Wörter, die in den n Hypothesen mindestens k-mal auftreten. Für den dritten Anfragetyp (**Ngram**) benötigen wir eine Methode, um die 1PT in N-gramme zu unterteilen, aus denen die Anfrage besteht. Dazu spalten wir die erste Übersetzung an den Stopwörtern auf und verlangen, dass für die zwischen ihnen enthaltenen Phrasen (N-gramme) die Reihenfolge der Wörter relevant ist. Wir erhalten also pro zu übersetzendem Satz (z.B. *i would like to contact the japanese embassy*) die folgenden Anfragetypen:

- **BoW**: Bag-of-Words-Anfrage: Verwendung aller Wörter der 1PT. Zu diesem Anfragetyp werden sowohl Anfragen mit Stopwörtern (BoW+S) als auch ohne Stopwörter (BoW-S) erzeugt.
  Bsp. BoW+S: *i would like to contact the japanese embassy*
  Bsp. BoW-S: *contact japanese embassy*

- **BoW-Maj**: Bag-of-Words-Anfrage basierend auf den n besten Hypothesen der ersten Übersetzung. Hierbei werden alle Wörter verwendet, die in der N-Best-Liste mindestens k-mal auftreten.
  Bsp. BoW-Maj: *embassy japanese call contact*

- **Ngram**: Anfrage unter Beachtung der Reihenfolge von Teilsequenzen. Teilsequenzen werden unter Verwendung von Stopwords ermittelt. Die Stopwords sind dabei die Stellen, an denen der 1PT-String in Teilsequenzen aufgespalten wird. Wir betrachten hier auch den Fall mit (Ngram+S) und ohne (Ngram-S) Stopwords.
  Bsp. Ngram-S: *„contact" „japanese embassy"*
  Bsp. Ngram+S: *i would like to contact the „japanese embassy"*

Bei der Generierung der Anfragen stellt sich noch die Frage, wieviele Dokumente von der Suchmachine geliefert werden sollen. Dies ist aber nicht so einfach zu beantworten, da man die Qualität der gelieferten Dokumente im voraus nicht kennt. Im Rahmen dieser Arbeit laden wir standardmässig eine feste Anzahl an Dokumenten aus dem Internet (falls vorhanden) und wenden dann unsere Strategien darauf an. Wir beschränken uns hier zwar auf HTML-Dokumente, doch der Ansatz kann grundsätzlich auf alle Arten von Textdokumenten (pdf, ps, doc, rtf, ...) angewendet werden.

4.3.3 Download und Säubern der Dokumente (Preprocessing)

In diesem Schritt werden die durch die Suchmaschine ermittelten Dokumente aus dem Internet heruntergeladen und vorverarbeitet. Hierbei werden störende Bestandteile der Dokumente entfernt und Fehler im Text der Webseite (Informationselemente) behoben. Störende Bestandteile können z.B. Code-Stücke aus Skriptsprache, Werbung, Orientierungs- und Navigationselemente, z.B. Links sein. Ein gutes Preprocessing ist eine entscheidende Voraussetzung für die Generierung eines guten Sprachmodells und für eine gute Übersetzungsqualität.

In dieser Arbeit wird keine Link-Information verwendet. Wir versuchen, aus den Webseiten den Text mit einfachen Mitteln zu ermitteln. Es werden also insbesondere keine HTML-Metadaten ausgewertet. Diese Tatsache sorgt für eine einfache Übertragung unseres Ansatzes auf andere Arten von Textdokumenten.

4.3.4 Download der Dokumente und Preprocessing

Die durch die Suchmaschine gefundenen Dokumente sind von der Suchmaschine mit einem Score bewertet und in dieser Reihenfolge ausgegeben. Die Anfragesprache der Suchmaschine unterscheidet sich in ihrer Ausdrucksmächtigkeit von Anfragesprachen für allgemeinere Information-Retrieval-Systeme wie z.B. das Lemur-Toolkit[1]. Zwar können neben Bag-Of-Words-Anfragen auch Anfragen gestellt werden, die die Reihenfolge der Wörter beachten, allerdings kann hier nur das unmittelbare Aufeinanderfolgen von Worten gefordert werden. Weitere Konstrukte, wie z.B. geordnete und ungeordnete Fenster (ordered/unordered windows) (Nachbarschaftsterme) oder stärkere Gewichtung von bestimmten Wörtern oder Phrasen (belief operators) sind nicht möglich. Ein weiterer Nachteil der Suchmaschinen ist, dass sie nur ganze HTML-Dokumente zurückliefern. Es könnte z.B. sein, dass ein Dokument im ersten Teil sehr gut auf die Anfrage passt, im zweiten Teil jedoch überhaupt nicht, weil z.B. ein Domänenwechsel vorliegt. Ein weiterer Nachteil besteht darin, dass die Relevanz zur Anfrage von der Suchmaschine (bis auf die Reihenfolgeangabe) nicht quantisiert dargestellt wird. So kann der Benutzer beispielsweise nicht erkennen, an welcher Rangposition in der Ergebnisliste die Relevanz stark abnimmt oder einen bestimmten Wert unterschreitet.
Aufgrund all dieser Überlegungen nehmen wir die Suchergebnisse der Suchmaschine nur als Vorauswahl aus all den Dokumenten des WWW und führen eine lokale Auswahl auf den Dokumenten als Basis für einen Adaptionskorpus durch. Hierbei betrachten wir verschiedene Fälle, indem wir den Begriff des Dokumentes jeweils unterschiedlich definieren. Zum einen betrachten wir jeweils das **gesamte (HTML-)Dokument** als Dokument, welches zur Auswahl steht. In weiteren Versuchen arbeiten wir dann auf **Satzebene**, wobei wir die durch die Suchmaschine vorgegebene Ordnung (auf den Dokumenten) vernachlässigen. In beiden Fällen müssen wir die Dokumente zuerst bewerten und dann basierend auf dieser Bewertung eine Auswahl treffen.

Bewertung Werden die ganzen Texte (HTML-Dokument) als Dokumente betrachtet, so erstellen wir für jeden Text ein Sprachmodell und berechnen die Perplexität der 1PT bzgl. dieses Sprachmodells. Es werden dann die n Texte verwendet, die die niedrigsten Perplexitätswerte aufweisen. n ist ein konstanter Parameter, der also unabhängig von den gefundenen Dokumenten ist.

Zur satzbasierten Auswahl mit Hilfe von IR-Methoden benutzen wir die Gewichtungsschemata TFIDF, OKAPI, sowie die Konstrukte der strukturierten Anfragesprache *Indri Query Language*. Wir verwenden hierzu das Lemur-Toolkit[1]. Hierbei kann die gewünschte Anzahl Zeilen als Parameter angegeben werden. Wir führen hierzu Versuche mit verschiedenen Werten durch. Weiterhin führen wir auch Auswahlen durch, bei denen wir einen Mindestscorewert für die Aufnahme in das Anfrageergebnis vorgeben.

Konstrukte der strukturierten Anfragesprache INDRI Bei der strukturierten Anfragesprache *Indri Query Language* erstellen wir Korpora für verschiedene Zeilenanzahlen. Wir stellen verschiedene Typen strukturierter Anfragen. Ziel strukturierter An-

fragen ist es, Informationen, welche über die des bloßen Auftretens von Wörtern hinausgehen, zu verwenden. Hierzu gehören z.B. die Wichtigkeit von Wörtern (Überzeugungsoperatoren; Belief-Operatoren), die Zuordnung von Gewichten zu Wörtern/Phrasen, sowie die (relative) Position der Wörter (Nachbarschaftsbedingungen).

Bag-of-Words-Anfragen Wir stellen zum einen einfache Bag-of-Words-Anfragen mit Hilfe des *#combine()*-Konstruktes und des *#syn()*-Konstruktes.

- Das *#combine()*-Konstrukt (*INDRI.combine*) erfordert ein Auftreten aller Wörter. Fehlt eines der Wörter, so ist der Wert nicht sehr hoch.
  Bsp.: *#combine(i would like to contact the japanese embassy).*

- Wir verwenden auch das *#syn()*-Konstrukt (*INDRI.syn*), welches alle Wörter als Synonyme, also als gleichwertig behandelt:
  Bsp.: *#syn(i would like to contact the japanese embassy).*
  Allerdings verwenden wir das *#syn()*-Konstrukt zusammen mit dem Ordered-Window-Operator *#N*, wobei wir beide Werte gewichtet mit den Gewichten w_1, w_2 in die Gesamtbewertung eingehen lassen:
  Bsp.: *#weight(* w_1 *#syn (i would like to contact the japanese embassy)* w_2 *#2 (i would like to contact the japanese embassy)).*
  Das *#syn()*-Konstrukt ist nicht so anfällig für fehlende Wörter wie das *#combine()*-Konstrukt.

Beachtung der Häufigkeiten in n-Best-List Der zweite und dritte Typ strukturierter Anfragen verwenden den *#wsum()*-Operator. Mit dessen Hilfe kann verschiedenen Wörtern oder N-grammen eine unterschiedliche Wichtigkeit zugeordnet werden. Wir generieren diese Anfrage auf Basis der N-Best-Liste, um so auf verschiedene Übersetzungsalternativen eingehen zu können. Die Häufigkeit des Auftretens eines Wortes oder N-grams in der N-Best-Liste bestimmt dabei seine Wichtigkeit. Dadurch, dass mit mehreren Übersetzungshypothesen gearbeitet wird, wird zum einen mehr Information zur Generierung des Adaptions-LMs verwendet, zum anderen wird der negative Einfluss einer schlechten Übersetzungsalternative abgeschwächt und somit das Rauschen gemindert.

- Der zweite Typ strukturierter Anfragen verwendet nur die gewichteten Wörter (*INDRI.simpleWsum*) bzw. die gewichteten N-gramme (*INDRI.ngramWsum*) der n-Best-List.
  Bsp. *INDRI.simpleWsum*: *#wsum(109 i 103 would 107 like 108 to 78 contact 100 the 90 japanese 100 embassy 22 call 18 want 6 is 13 you 4 for 10 a 4 make 13 have 3 my 4 some 2 we 2 it 1 do)*
  Bsp. *INDRI.ngramWsum*: *#wsum(90 i 82 would 99 like 86 to 76 contact 89 #1(japanese embassy) 22 call 18 want 9 embassy 11 you 4 make 8 a 11 have 1 #1(contact japanese embassy) 2 some 2 it 1 #1(contact embassy) 1 do)*

- Hingegen verwendet der dritte Typ strukturierter Anfragen (*INDRI.ngramWsumUO*) den *Unordered-Window-Operator #uw()*, bei dem alle Terme auftreten müssen, wobei die Reihenfolge des Auftretens jedoch keine Rolle spielt:
  Bsp.: *#wsum(90 i 82 would 99 like 86 to 76 contact 89 #uw1(japanese embassy) 22 call 18 want 9 embassy 11 you 4 make 8 a 11 have 1 #uw1(contact japanese embassy) 2 some 2 it 1 #uw1(contact embassy) 1 do).*

4.3.5 Adaption des Sprachmodells

Wir erstellen für jeden Satz pro Anfragetyp ein Adaptionssprachmodell, das wir neben dem Baseline-Sprachmodell verwenden. Wir verwenden in verschiedenen Versuchen unterschiedliche Gewichte für das Adaptionssprachmodell, wobei als Gewicht für das Baselinesprachmodell das Gewicht der bei der ersten Übersetzung gefundenen optimalen Parameter verwendet wird.

4.4 Implementierungsdetails

4.4.1 Generierung der Übersetzungen

Für die erste Übersetzung wird das System bzgl. des Testsets nach BLEU-Score oder NIST-Score optimiert. Für die zweite Übersetzung wird keine weitere Optimierung durchgeführt, sondern es werden die bei der ersten Übersetzung ermittelten optimalen Parameter für die einzelnen Modelle verwendet.

4.4.2 Preprocessing

Im Preprocessing-Schritt müssen die erhaltenen HTML-Dokumente in Textdokumente transformiert und störende Elemente entfernt werden. Wir verwenden hierzu den Lynx-Browser mit der dump-Option. Daran schliesst sich ein weiterer Schritt an, in dem übriggebliebene Elemente entfernt werden. Dies sind z.B. Links, die vom Lynx-Browser angezeigt werden oder Elemente aus Skriptsprachen wie z.B. Java-Script oder auch HTML-Tags, die vom Lynx-Browser nicht als solche erkannt werden. Hierzu werden das *grep*-Tool und das *sed*-Tool verwendet.
Nach dem Entfernen störender Elemente wird von einem weiteren Tool nach Dokumenten gesucht, die offensichtlich nicht zu den gewünschten Texten gehören, wie z.B. Wortlisten für Textkorpora und andere Listen. Hierzu werden die Anzahl der Wörter, der Zahlen und Satzzeichen bestimmt und in Relation zueinander gesetzt, so dass die meisten Listen ausgeschlossen werden können. Ebenso werden alle im Logfile von *wget* als Applikationen gekennzeichneten Dateien entfernt.
Danach werden die Texte in Sätze unterteilt, um die Daten so in das gewünschte Format zu transformieren. Als letzter Schritt werden die Sätze in eine normalisierte Form überführt.

5 Experimente

5.1 Übersetzungssystem

Übersetzungsmodell (TM) und Decoder In den Versuchen wird das CMU-Statistical-Machine-Translation-System (Vogel et al. [33], Eck et al. [13]) verwendet. Das System verwendet die PESA-Phrasenextraktion (Vogel - PESA-Phrasen-Extraktion [35]), die Phrasen aufgrund von IBM1-Scores bewertet. Die Phrasen werden dynamisch, d.h. während der Üebersetzung extrahiert und sind damit beliebig lang. Weitere Informationen über das System gibt es zum Dekoder von Vogel [34] und zum Phrase-to-Phrase-Alignment-Modell [38].

Sprachmodell (LM) Wir benutzen das SRILM-Toolkit [31], um Sprachmodelle zu erzeugen, Perplexitäten zu berechnen und Sprachmodelle zu interpolieren. Wir erzeugen Trigramm-Sprachmodelle mit Good-Turing-Discounting und Katz-Backoff fürs Smoothing, wobei nicht zwischen Groß- und Kleinschreibung unterschieden wird.
Zur Benutzung im Dekoder interpolieren wir das Baselinesprachmodell nicht mit dem Adaptionssprachmodell, sondern verwenden das Adaptionssprachmodell als weiteres Modell neben dem Baselinesprachmodell. Hier findet also keine Interpolation der Sprachmodelle statt. Lediglich zur Erzeugung eines Adaptionssprachmodells aus den Texten verschiedener HTML-Dokumente interpolieren wir die Sprachmodelle dieser einzelnen Dokumente. Das daraus resultierende Sprachmodell wird allerdings ebenfalls separat neben dem Baselinesprachmodell im Dekoder benutzt.

Evaluationsmetriken Zur Evaluation werden die bekannten Metriken BLEU (Abschnitt 2.6.4 und [26]) und NIST (Abschnitt 2.6.5 und [12]) verwendet. Um die besten Scores zu ermitteln, wird die Übersetzung mit verschiedenen Gewichten (Parametern) für das Adaptionssprachmodell durchgeführt und in einem zweiten Schritt werden die Gewichte basierend auf der N-Best-List aller Sätze extern optimiert.

5.2 Szenarien

In den Experimenten zu dieser Studienarbeit wird mit zwei Sprachpaaren gearbeitet. Als Zielsprache verwenden wir in beiden Fällen Englisch, da das WWW reich an Webseiten in englischer Sprache ist. Die Quellsprachen sind Spanisch und Japanisch.

5.2.1 Szenario 1: BTEC+medical-SpaEng

Trainings- und Testdaten Bei den Daten handelt es sich um bilinguale Phrasenpaare (eine Referenzübersetzung). Die Daten aus dem medical-Korpus entstammen einer medizinischen Datenbank, bestehend aus Mitschnitten von Gesprächen zwischen Arzt und Patient. Der BTEC-Korpus (Basic Travel Expression Corpus) [32] ist ein mehrsprachiger Korpus, bestehend aus touristischen Phrasen. Die beiden Korpora sind sich bezüglich des Stils ähnlich, die Thematik ist aber eine andere. Wir verwenden für unsere Untersuchungen diese beiden Korpora zusammen (Gesamttestdaten und Gesamttrainingsdaten). Die durchschnittliche Satzlänge ist 8,03 Wörter für Englisch und 7,15 Wörter für Spanisch. Es handelt sich also um recht kurze Sätze. Das aus den Trainingsdaten erzeugte Sprachmodell bezeichnen wir im folgenden als Baselinesprachmodell (BLM).

Bezeichnung	# Sätze	# Wörter Spanisch	# Wörter Englisch
medical-TEST	329	3065	3399
BTEC-TEST	245	1581	1667
Gesamttestdaten	574	4646	5066
medical-TRAIN	24619	205402	221543
BTEC-TRAIN	123171	850781	901858
Gesamttrainingsdaten	147790	1056183	1123401

Tabelle 3: Verwendete Korpora in Szenario 1 (BTEC+medical-SpaEng)

Beispielsätze

Satz 1: yes i would like an information regarding muscles
Satz 2: that is right
Satz 3: all right so you have a neuropathy and now we come to the art of medicine 'cause we don't know what to do
Satz 4: what kind of sightseeing bus tours do you have
Satz 5: i'd like you to help me please
Satz 6: do you have something with darker colors

5.2.2 Szenario 2: BTEC-JapEng

Trainings- und Testdaten Dieser Korpus (BTEC-Korpus [32]) besteht aus touristischen Phrasen. Die Quellsprache ist Japanisch und Zielsprache ist Englisch. Für die Zielsprache liegen 16 Referenzen vor. Die englischen Sätze haben eine durchschnittliche Länge von 6,18 Wörtern. Damit handelt es sich ebenfalls um recht kurze Sätze.

Bezeichnung	# Sätze	# Wörter Japanisch	# Wörter Englisch
Testdaten	500	3773	3713 (57347 [16 Ref])
Trainingsdaten	162318	1188106	1003785

Tabelle 4: Verwendete Korpora in Szenario 2 (BTEC-JapEng)

Beispielsätze

Satz 1: where can i find the foreign currency exchange
Satz 2: could i get a different room that faces the ocean
Satz 3: can anyone speak in japanese
Satz 4: i must apologize since i just ran out of business cards and do not have 1 to give to you at the moment
Satz 5: i got it
Satz 6: my hair is thin here could you fill it out more here

5.2.3 Baselinesysteme

Das Baselinesystem wird mit den Trainingsdaten trainiert und bezüglich der zu übersetzenden Daten (Testdaten) zum einen mit der BLEU-Metrik, zum anderen auch mit der NIST-Metrik optimiert. Beim BTEC-JapEng-System werden dazu 16 Referenzen verwendet, beim BTEC+medical-SpaEng-System nur eine Referenz. Die berechneten Scores lassen sich in Tabelle 5 ablesen. Die Systeme sind bezüglich der Testdaten optimiert. In der Praxis ist dies natürlich nicht möglich, da keine Referenzübersetzungen vorliegen.

System	Optimierungsmetrik	BLEU-Score	NIST-Score
BTEC+medical-SpaEng	BLEU	0.3449	6.4985
BTEC+medical-SpaEng	NIST	0.3367	6.6242
BTEC-JapEng	BLEU	0.5726	10.0373
BTEC-JapEng	NIST	0.5686	10.7408

Tabelle 5: Baseline-Scores

5.2.4 Adaptionssysteme

Die für das Baselinesytem ermittelten optimalen Parameter übernehmen wir unverändert für den zweiten Übersetzungsschritt. Lediglich den Parameter für das Adaptionssprachmodell fügen wir hinzu. Wir benutzen verschiedene Gewichte (Parameter) für das Adaptionssprachmodell und vergleichen die Übersetzungsleistungen. Die für das Baselinesystem ermittelten Parameter müssen bei Hinzunahme des Adaptionssprachmodell natürlich nicht mehr optimal sein. Deshalb führen wir nachträglich eine externe Optimierung auf den N-Best-Listen aller Sätze gleichzeitig durch.

5.3 Retrieval mit Hilfe des WWW

Wir erzeugen Anfragen für verschiedene Anfragetypen (BoW-S, BoW+S, Ngram-S, Ngram+S, BoW.Maj) und stellen diese an die Suchmaschine. Hier vergleichen wir die von der Suchmaschine je Anfragetyp gelieferten URLs bzw. die zugehörigen Dokumente. Wir untersuchen zum einen die Auswirkung der Verwendung von BLEU-optimalen bzw. NIST-optimalen Parametern. Eine zweite Untersuchung vergleicht die Anfragegenerierung aus der 1PT mit der auf der Referenz (REF). Schließlich vergleichen wir die Anfragetypen untereinander bei Verwendung der BLEU-Metrik. Wir untersuchen dabei jeweils, wieviele Dokumente für die Adaption (Adaptiondokumente; AD) zur Verfügung stehen und vergleichen die Anfragetypen paarweise, indem wir für die Ergebnismenge eines Anfragetyps den Anteil der Dokumente ermitteln, die in beiden untersuchten Mengen enthalten sind (doppelte Dokumente; DD). Tabelle 6 fasst die untersuchten Größen zusammen.

5.3.1 Allgemeine Beobachtungen

Aus den Daten der Tabellen 7 und 8 erkennt man, dass die Systeme ohne Stopwords weniger Adaptionsdokumente finden als die Anfragetypen mit Stopwords. Dies ist auf den ersten Blick überraschend, da die Ergebnismenge durch die Stopwords ja eigentlich weiter eingeschränkt wird. Dieser Effekt ist aber dadurch zu erklären, dass einige Sätze nur aus Stopwords bestehen und beim Entfernen dieser keine Wörter mehr

Kürzel	Bedeutung
AD0	Anteil der Sätze (in Prozent), für die kein Adaptionskorpus erzeugt werden konnte.
AD10	Anteil der Sätze (in Prozent), für die weniger als 10 Dokumente gefunden wurden. Sätze ohne Adaptionskorpus sind hierin enthalten.
#AD	Durchschnittliche Anzahl gefundener Dokumente für die Sätze, für die überhaupt Adaptionsdokumente gefunden wurden.
DD	Anteil der doppelten Dokumente (in Prozent).
DD90	Anteil der Sätze, die mindestens 90% doppelte Dokumente enthalten.
DD10	Anteil der Sätze, die höchstens 10% doppelte Dokumente enthalten.

Tabelle 6: Untersuchte Größen

für die Anfrage übrigbleiben. Werden Adaptionsdokumente gefunden, so sind dies recht viele: 91,7-97,1 Dokumente im BTEC+medical-SpaEng-System und 91,9-98,0 im BTEC-JapEng-System. Während die übrigen Werte im BTEC+medical-SpaEng-System und BTEC-JapEng-System sehr ähnlich sind, findet die BOW.Maj-Anfrage im BTEC-JapEng-System deutlich mehr Dokumente.

5.3.2 Optimierung nach BLEU vs. Optimierung nach NIST

Bei Verwendung von NIST-optimalen Parametern ist zu beobachten, dass die Hypothesen länger werden. Alle Maße sind bei beiden Optimierungsmaßen sehr ähnlich. Es werden annähernd gleich viele Dokumente gefunden und auch bei den Sätzen mit Adaptionsdokumenten ist die durchschnittliche Anzahl gefundener Sätze sehr ähnlich. Der Anteil der doppelten Dokumente liegt zwischen 78,0% und 90,8% für BLEU und 79,3% und 89,8% für NIST im BTEC+medical-SpaEng-System (Tablle 7). Beim BTEC-JapEng-System (Tablle 8) sind die #AD-Werte nur leicht höher, lediglich beim BOW.Maj-System werden 4,5% mehr Dokumente gefunden als beim BTEC+medical-SpaEng-System. Beim BTEC-JapEng-System gibt es jedoch deutlich weniger doppelte Dokumente: Der DD-Wert liegt 12,5%-19,4% unter dem Wert für das BTEC+medical-SpaEng-System.

Anfragetyp	BoW-S	BoW+S	Ngram-S	Ngram+S	BOW.Maj
AD0 (in %)	8,0 / 7,8	0,6 / 0,4	16,0 / 15,0	8,4 / 7,4	23,6 / 24,4
AD10 (in %)	9,6 / 9,2	1,4 / 1,0	18,2 / 17,0	12,0 / 10,8	25,8 / 26,2
#AD	96,1 / 96,0	97,1 / 96,8	92,5 / 92,2	91,7 / 91,6	92,8 / 92,8
DD (in %)	89,7 / 89,5	86,3 / 86,1	90,8 / 89,8	87,0 / 85,8	78,0 / 79,3
DD90 (in %)	86,1 / 85,7	79,1 / 78,9	90,2 / 89,2	81,9 / 80,8	66,0 / 67,7
DD10 (in %)	5,9 / 6,1	5,4 / 5,6	7,9 / 8,5	7,2 / 8,4	10,2 / 9,0

Tabelle 7: BTEC+medical-SpaEng-System: Vergleich gefundener Dokumente bei Anfragegenerierung aus Hypothesen erzeugt mit BLEU- bzw. NIST-optimalen Parametern. (Einträge: BLEU / NIST)

5.3.3 1PT-Anfragen vs. Referenzanfragen

Bei diesem Orakelexperiment werden die Dokumentenlisten von Anfragen, basierend auf der 1PT mit den Dokumentenlisten, die mit der Referenzübersetzung REF (1REF

Anfragetyp	BoW-S	BoW+S	Ngram-S	Ngram+S	BOW.Maj
AD0 (in %)	9,4 / 8,6	0,0 / 0,0	16,4 / 15,6	8,2 / 8,2	4,6 / 4,8
AD10 (in %)	9,8 / 9,2	0,4 / 0,8	19,4 / 18,2	11,2 / 11,2	6,0 / 5,8
#AD	98,0 / 97,9	97,9 / 97,8	92,2 / 93,2	91,9 / 92,5	96,9 / 97,0
DD (in %)	78,5 / 77,9	70,9 / 71,1	77,6 / 77,0	70,1 / 70,1	64,8 / 64,9
DD90 (in %)	76,2 / 75,5	63,8 / 63,8	76,6 / 76,1	64,9 / 64,9	56,6 / 56,9
DD10 (in %)	17,9 / 18,8	19,2 / 19,2	20,6 / 21,3	23,5 / 23,5	22,6 / 22,3

Tabelle 8: BTEC-JapEng-System: Vergleich gefundener Dokumente bei Anfragegenerierung aus Hypothesen erzeugt mit BLEU- bzw. NIST-optimalen Parametern. (Einträge: BLEU / NIST)

im Falle des BTEC-JapEng-Systems) ermittelt wurden, verglichen. Beim BTEC+medical-SpaEng-System werden annähernd ähnlich viele Dokumente gefunden. Auffallend ist die geringe Anzahl doppelter Dokumente. Bei den BoW-S- und Ngram-S-Systemen sind dies rund ein Drittel der Dokumente, bei den BoW+S- und Ngram+S-System jedoch nur knapp ein Viertel. Deutliche Unterschiede treten beim BOW.Maj-System auf: Das BOW.Maj-REF-System findet für nahezu alle Dokumente (99,4%) Adaptionsdokumente. Beim BOW.Maj-System ist dies nur für 76,4% der Dokumente der Fall. Dies ist dadurch zu erklären, dass die Anfrage beim BOW.Maj-System restriktiver ist. Die Anzahl der doppelten Dokumente ist hier mit unter 10% sehr gering.

Anfragetyp	BoW-S	BoW+S	Ngram-S	Ngram+S	BOW.Maj
AD0 (in %)	8,0 / 4,8	0,6 / 0,8	16,0 / 12,8	8,4 / 8,4	23,6 / 0,6
AD10 (in %)	9,6 / 5,8	1,4 / 2,2	18,2 / 15,0	12,0 / 12,4	25,8 / 1,8
#AD	96,1 / 95,8	97,1 / 96,1	92,5 / 91,0	91,7 / 89,1	92,8 / 96,0
DD (in %)	34,7 / 33,4	26,8 / 26,8	35,5 / 34,7	26,8 / 27,2	8,2 / 5,9
DD90 (in %)	30,2 / 29,4	19,9 / 19,8	33,3 / 32,6	21,4 / 21,8	4,7 / 3,4
DD10 (in %)	59,3 / 61,6	59,2 / 59,3	61,7 / 62,4	63,1 / 62,4	86,1 / 89,7

Tabelle 9: Untersuchung 1PT-Anfragen vs. Referenzanfragen (BTEC+medical-SpaEng-System) (Einträge: 1PT / Referenz)

Wie das BTEC+medical-SpaEng-System, hat das BTEC-JapEng-System einen hohen #AD-Wert. Die Anzahl der doppelten Dokumente ist allerdings deutlich geringer. Wie beim BTEC+medical-SpaEng-System, ist der DD-Wert für die BoW-S- und Ngram-S-Systeme (16,2%-17,5%) deutlich höher als der für die BoW+S- und Ngram+S-Systeme (6,9%-7,9%). Die Abweichungen beim BOW.Maj-System treten hier nicht auf, da hier 16 Referenzen verwendet werden. Der DD-Wert beträgt hier allerdings nur 4,6% bzw. 4,3%. Die deutlich geringeren Übereinstimmungen sind dadurch zu erklären, dass für das REF-System nur aus einer Übersetzung Anfragen erstellt wurden, wohingegen die 1PT auf einem mit 16 Referenzen trainierten System erstellt wurde.

5.3.4 Vergleich der Anfragetypen untereinander

Beim Vergleich der Anfragetypen untereinander fällt auf, dass die Anfragetyppaare ohne Stopwords (BoW-S/Ngram-S) sowie die Anfragetyppaare mit Stopwords (BoW+S/Ngram+S) die höchste Anzahl an doppelten Dokumenten haben. Hier stimmen im BTEC+medical-SpaEng-System 71,0%-80,2% der Dokumente überein, im BTEC-JapEng-System sind

Anfragetyp	BoW-S	BoW+S	Ngram-S	Ngram+S	BOW.Maj
AD0 (in %)	9,4 / 7,0	0,0 / 0,2	16,6 / 13,2	8,2 / 7,6	4,6 / 0,2
AD10 (in %)	9,8 / 7,6	0,4 / 1,2	19,4 / 17,4	11,2 / 12,2	6,0 / 0,8
#AD	98,0 / 97,6	97,9 / 97,5	92,5 / 91,2	91,9 / 91,0	96,9 / 98,0
DD (in %)	17,5 / 16,8	7,5 / 7,3	16,9 / 16,2	6,9 / 6,9	4,6 / 4,3
DD90 (in %)	14,6 / 14,0	2,4 / 2,0	15,6 / 15,0	2,6 / 2,4	1,9 / 1,6
DD10 (in %)	78,8 / 79,4	83,2 / 83,4	81,8 / 82,5	86,5 / 86,4	91,4 / 91,4

Tabelle 10: Untersuchung 1PT-Anfragen vs. Referenzanfragen (BTEC-JapEng-System) (Einträge: 1PT / Referenz)

es nur 57,4%-75,7%, wobei es zwischen den Anfragetypen ohne Stopwords weniger Übereinstimmungen gibt. Die verwandten Anfragetypen Ngram+/-S sowie BoW+/-S stimmen nur zu rund einem Viertel überein. Am wenigsten Übereinstimmungen gibt es zwischen den BOW.Maj-Anfragen und den anderen Typen, da die BOW.Maj-Anfragen einschränkender sind. Beim BTEC-JapEng-System sind es nur 1,8% bzw. 2,0% zwischen Ngram+S und BOW.Maj, beim BTEC+medical-SpaEng-System 3,2% bzw. 3,8%.

Adaptionstyp	BoW-S	BoW+S	Ngram-S	Ngram+S	BOW.Maj
BoW-S	-	24,6	71,7	15,6	12,5
BoW+S	22,7	-	13,7	71,0	5,3
Ngram-S	80,2	16,9	-	27,4	9,8
Ngram+S	17,3	78,5	27,2	-	3,2
BOW.Maj	16,3	8,6	10,8	3,8	-

Tabelle 11: Vergleich der Anfragentypen (BTEC+medical-SpaEng-System) untereinander. In der Matrix gibt der Eintrag (i,j) den Anteil der in den Antwortlisten von Querygenerierungstyp i und j gemeinsam enthaltenen Dokumenten in der Menge von Typ i an.

Adaptionstyp	BoW-S	BoW+S	Ngram-S	Ngram+S	BOW.Maj
BoW-S	-	12,8	57,4	6,9	8,8
BoW+S	11,3	-	5,9	69,0	4,1
Ngram-S	63,7	7,3	-	15,2	5,7
Ngram+S	8,0	75,7	16,7	-	2,0
BOW.Maj	8,4	4,7	4,9	1,8	-

Tabelle 12: Vergleich der Anfragentypen (BTEC-JapEng-System) untereinander. In der Matrix gibt der Eintrag (i,j) den Anteil der in den Antwortlisten von Querygenerierungstyp i und j gemeinsam enthaltenen Dokumenten in der Menge von Typ i an.

5.4 Untersuchung der Übersetzungsleistung

Wir untersuchen im Folgenden die Übersetzungsleistung des BTEC+medical-SpaEng-Systems mit Bag-of-Words-Anfrage und unter Verwendung von Stopwords (BoW+S). In Abschnitt 5.8 werden dann noch ausgewählte Adaptionstypen für das BTEC-JapEng-

Systems mit Bag-of-Words-Anfragen unter Verwendung von Stopwords (BoW+S) untersucht.

5.4.1 Übersetzungsergebnisse

Wir stellen die Ergebnisse der einzelnen Adaptionstypen für das BTEC+medical-SpaEng-Systems dar und untersuchen, welche Kombination aus Adaptionskorpusgröße und Gewicht das beste Resultat geliefert hat.

Dokumentbasierte Auswahl Hier wird eine feste Anzahl von gefundenen Dokumenten als Korpus für das Adaptionssprachmodell verwendet. Wir untersuchen die Größen $N = 10, 20, 30$. Für die N-First-Korpora werden die *ersten* N von der Suchmaschine gefundenen Dokumente zur Konstruktion des Adaptionssprachmodells verwendet. Für die N-Best-Korpora werden die N von der Suchmaschine gefundenen Dokumente verwendet, die die *geringste Perplexität* bezüglich der 1PT aufweisen.

Satzbasierte Auswahl Für diese Adaptionskorpora wurden einzelne Sätze ausgewählt. Diese wurden bestimmt, indem die Sätze durch einen Score bewertet und dann die N besten ausgewählt wurden. Für die *TFIDF*- und *OKAPI*-Korpora geschah dies mit dem TFIDF- und dem OKAPI-Score. *INDRI.combine* fordert ein Auftreten aller Wörter. *INDRI.syn* setzt sich aus zwei Komponenten zusammen, die gewichtet einen Gesamtwert ergeben. Zum einen behandelt der #syn-Operator alle Wörter gleich (als Synonyme). Zum anderen wird als zweiter Bestandteil der Orderd-Window-Operator #N verwendet. Der Score für *INDRI.simpleWsum* berechnet sich aus der Häufigkeit der Wörter des Satzes. *INDRI.ngramWsum* ebenso, beachtet aber auch N-gramme. *INDRI.ngramWsumUO* basiert auch auf der Häufigkeit der Wörter eines Satzes. Statt aber das Auftreten von N-grammen (wie *INDRI.ngramWsum*) zu fordern, müssen hier die Wörter eines N-grams nur hintereinander auftreten. Die Reihenfolge des Auftretens ist egal (Bsp.: *japanese embassy* oder *embassy japanese* für das N-gram *japanese embassy*). Für weitere Details siehe Abschnitt 4.3.4.

Anzahl Adaptionskorpora Für die verschiedenen Anfragetypen konnten nur für einen Teil der Sätze Adaptionskorpora gefunden werden. Tabelle 13 gibt eine Übersicht über die Anzahl der verwendeten Adaptionskorpora je Anfragetyp. Obwohl mit den Wsum-Anfragen (INDRI.ngramWsum-, INDRI.ngramWsumUO- und INDRI.simpleWsum) die besten Adaptions-BLEU-Scores erreicht wurden, konnten mit diesen Adaptionstypen nur rund zwei Fünftel der Sätze adaptiert werden. Mit den anderen Adaptionstypen konnten bis auf die TFIDF.score-Typen für fast alle Sätze Adaptionskorpora gefunden werden.

Vergleich der Ergebnisse der Adaptionstypen Beim BTEC+medical-SpaEng-System (BoW+S mit Optimierung nach BLEU) schneiden alle adaptierten Systeme schlechter ab als das Baseline-System, dessen 1PT mit einem BLEU-Score von 0.3449 bewertet wurde. Der beste BLEU-Score der Adaptionsmodelle ist 0.3413. Obwohl sich der BLEU-Score bei allen Arten der Datenauswahl verschlechtert hat, gibt es jedoch Unterschiede zwischen den einzelnen Auswahltypen.
Bei allen Adaptionstypen lässt sich erkennen, dass der BLEU-Score tendenziell umso besser wird, je größer der Einfluss des Adaptionssprachmodells ist (größeres Gewicht).

Adaptionstyp	Anzahl Adaptionskorpora
N-First	494
N-Best	493
INDRI.combine	493
INDRI.syn	493
INDRI.ngramWsum	207
INDRI.ngramWsumUO	206
INDRI.simpleWsum	207
OKAPI	493
TFIDF	493
TFIDF.score-10	454
TFIDF.score-15	393
TFIDF.score-20	336
TFIDF.score-25	265

Tabelle 13: Anzahl Adaptionskorpora je Adaptionstyp

In den meisten Fällen sind auch die Adaptionen besser, die mehr Zeilen für den Adaptionskorpus verwenden.

N-First- und N-Best-Anfragen Für die dokumentenbasierten Korpora konnte für fast alle Sätze (494 bzw. 493 Sätze) ein Adaptionskorpus erzeugt werden. Die BLEU-Scores sind jedoch im Vergleich zu denen der satzbasierten Adaptionsmodelle niedriger.

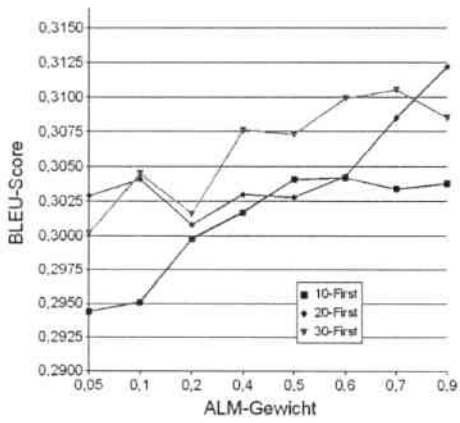

Abbildung 3: BLEU-Scores der N-First-Anfragen

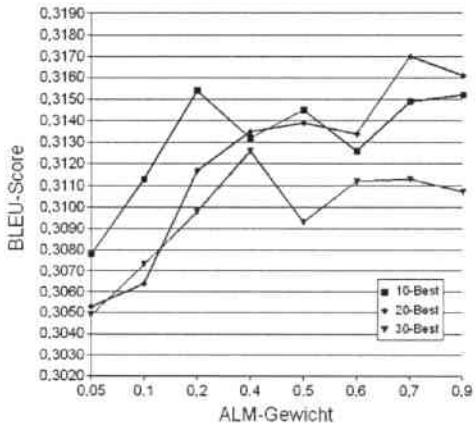

Abbildung 4: BLEU-Scores der N-Best-Anfragen

Bei den N-First-Adaptionstypen lässt sich beobachten, dass der BLEU-Score besser wird, je mehr Daten verwendet werden (Abbildung 3). Bei den N-Best-Adaptionstypen ist das Gegenteil der Fall: Je weniger, aber besser die Daten sind (geringere Perplexität), desto höher der Score (Abbildung 4). Wie in Tabelle 5 zu erkennen ist, stellen die N-Best-BLEU-Scores bzgl. der N-First-BLEU-Scores eine leichte Verbesserung dar (0.0058-0.0116, im Mittel 0.0077). Besonders bei $N = 10$ ist der Abstand von 0.0123

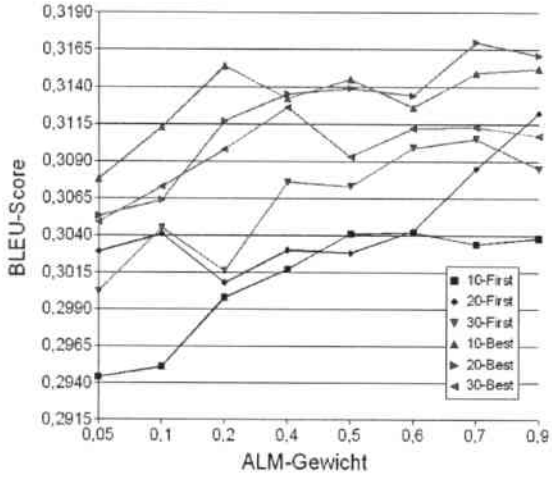

Abbildung 5: N-First vs. N-Best

hervorzuheben. Bei gleicher Anzahl an Dokumenten wird ein besserer BLEU-Score erreicht.

INDRI.combine- und INDRI.syn-Anfragen Bei beiden Anfragetypen schneiden die Adaptionsmodelle mit höherem Einfluss des Adaptionsprachmodells besser ab. Während der Anstieg von Gewichten von 0.05-0.4 stark ist, schwächt er sich bei höheren Gewichten eher ab. Besonders schlecht schneiden die Adaptionsmodelle mit kleinem Korpus und kleinem Gewicht ab. Bei den INDRI.syn-Anfragen schneiden die Modelle mit Korpusgrößen 50 und 100 über alle gewählten Gewichte schlechter ab als die anderen Modelle. Hier wirkt sich die strengere Auswahl bei den INDRI.combine-Anfragen positiv auf den BLEU-Score aus.

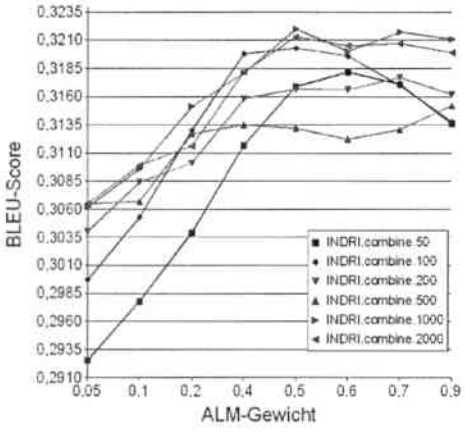

Abbildung 6: BLEU-Scores der INDRI.combine-Anfragen

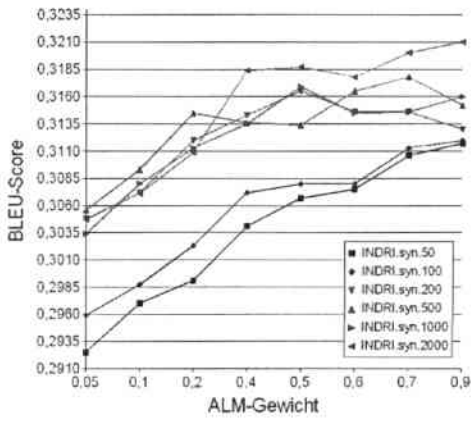

Abbildung 7: BLEU-Scores der INDRI.syn-Anfragen

OKAPI- und TFIDF-Anfragen Auch hier schneiden bei beiden Anfragetypen die Adaptionsmodelle mit höherem Einfluss des Adaptionsprachmodells besser ab. Während der Anstieg von Gewichten von 0.05-0.4 stark ist, schwächt er sich bei höheren Gewichten eher ab. Besonders schlecht schneiden die Adaptionsmodelle mit einm Korpus von 50 Zeilen ab. Hingegen erreichen die Adaptionssprachmodelle mit einer Korpusgröße von 100 Zeilen ähnlich gute Scores, wie die Adaptionssprachmodelle mit größerem Korpus. Bei den OKAPI-Sprachmodellen werden die höchsten BLEU-Scores sogar mit den OKAPI.100-Adaptionssprachmodellen erreicht.

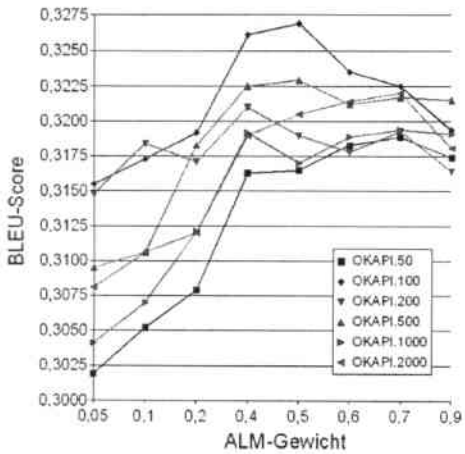

Abbildung 8: BLEU-Scores der OKAPI-Anfragen

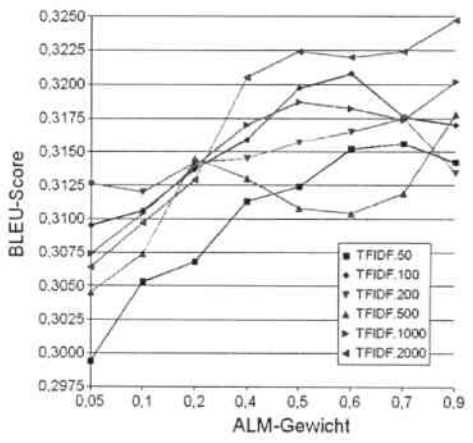

Abbildung 9: BLEU-Scores der TFIDF-Anfragen

TFIDF.score- und INDRI.ngramWsum-Anfragen Bei den INDRI.ngramWsum-Anfragen lässt sich erkennen, dass mit größeren Adaptionskorpora ($\geq$ 500 Zeilen) bessere Ergebnisse erreicht werden können. Hier können sogar schon mit Adaptionsmodellen mit kleinem Einfluss gute Ergebnisse erzielt werden.
Bei den TFIDF.score-Modellen sieht man deutlich, dass mindestens ein Gewicht von 0.4 benötigt wird, um bessere Ergebnisse zu erlangen. Weiterhin kann man erkennen, dass mehr Daten den Score verbessern, dieser aber deutlich sinkt, wenn die Qualität der Daten nachlässt (TFIDF.score-10.00).

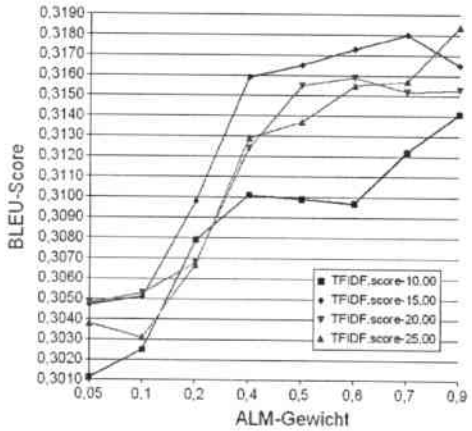

Abbildung 10: BLEU-Scores der TFIDF.score-Anfragen

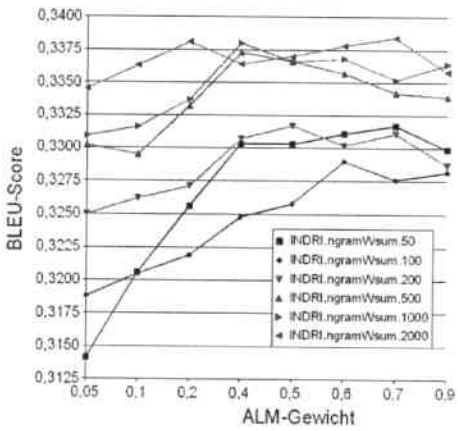

Abbildung 11: BLEU-Scores der INDRI.ngramWsum-Anfragen

INDRI.ngramWsumUO- und INDRI.simpleWsum-Anfragen Aus den INDRI.ngramWsumUO- und INDRI.simpleWsum-Anfragen kann man ablesen, dass mehr Daten den Score verbessern. Der Score hängt besonders bei Modellen mit größerem Korpus nicht so stark von der Wahl des Gewichtes ab. Die INDRI.ngramWsum-, INDRI.ngramWsumUO- und INDRI.simpleWsum-Anfragen erzeugen die besten BLEU-Scores, allerdings sind diese nicht besser als der BLEU-Score der 1PT des Baseline-Systems.

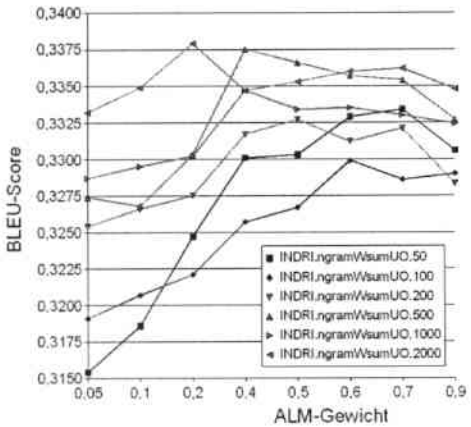

Abbildung 12: BLEU-Scores der INDRI.ngramWsumUO-Anfragen

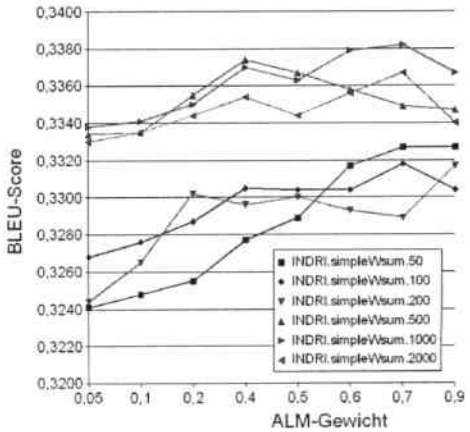

Abbildung 13: BLEU-Scores der INDRI.simpleWsum-Anfragen

Vergleich der satzbasierten Adaptionstypen Die Adaptionstypen *INDRI.simpleWsum*, *INDRI.ngramWsum* und *INDRI.ngramWsumUO* schneiden wesentlich besser ab als die anderen Adaptionstypen.
Die *OKAPI*- und *TFIDF*-Typen sind in etwa gleich gut. *INDRI.combine* und *INDRI.syn* erreichen bei guten Gewichten fast so gute BLEU-Scores wie *OKAPI*- und *TFIDF*, können aber bei geringen Gewichten wesentlich schlechter werden.
Die *TFIDF.score*-BLEU-Scores sind ein wenig schlechter als die *TFIDF-Scores*, da weniger Daten für die Korpora zur Verfügung stehen und die Schätzwerte für die Parameter somit unzuverlässiger sind. Tabelle 14 gibt eine Übersicht über die durchschnittliche Anzahl verwendeter Zeilen für die Adaptionskorpora.
Dass die BLEU-Scores besser werden, wenn mehr Daten für die Adaptionskorpora

Adaptionskorpus	Durchschnittliche Zeilenanzahl
TFIDF.score-25	90.8
TFIDF.score-20	154.08
TFIDF.score-15	296.52
TFIDF.score-10	650.08

Tabelle 14: Durchschnittliche Anzahl Zeilen für TFIDF.score-Adaptionskorpora

verwendet werden, ist besonders bei den Typen *INDRI.syn*, *INDRI.ngamWsum* und *INDRI.simpleWsum* zu beobachten.

In Tabelle 15 sind die besten BLEU-Scores der Adaptionstypen aufgeführt. Auch hier sieht man noch einmal, dass die Anfragen mit gewichteten Summen (INDRI.ngramWsum, INDRI.simpleWsum und INDRI.ngramWsumUO) am besten abschneiden. Danach folgen mit Abstand die OKAPI- und TFIDF-Adaptionstypen. Die dokumentbasierten Anfragetypen schneiden am schlechtesten ab. Die satzbasierte Adaption ist also besser als die dokumentbasierte.

Adaptionstyp	Bester Wert	optimale Korpusgröße	optimales ALM-Gewicht
INDRI.ngramWsum	0.3384	2000	0.7
INDRI.simpleWsum	0.3382	1000	0.7
INDRI.ngramWsumUO	0.3379	2000	0.9
OKAPI	0.3269	100	0.5
TFIDF	0.3247	2000	0.9
INDRI.combine	0.3220	1000	0.5
INDRI.syn	0.3210	2000	0.9
TFIDF.score	0.3184	score ≥ 25	0.9
N-Best	0.3170	20 Dokumente	0.7
N-First	0.3122	20 Dokumente	0.9

Tabelle 15: Bester Wert je Adaptionstyp

5.4.2 Scores der Orakelanfragen

Wir haben Orakelexperimente durchgeführt, indem wir die Referenzübersetzung anstelle der 1PT zur Anfragegenerierung verwendet haben. Dies haben wir sowohl auf der ersten Stufe (Suchmaschinenanfrage), also auch auf der zweiten Stufe (lokale Auswahl: dokumentenbasierte Auswahl und Auswahl mit Lemur) durchgeführt.
Schreibweise: Wir schreiben im Folgenden X-Y-Adaption, wenn auf der ersten Stufe X zur Anfragegenerierung verwendet wurde und Y auf der zweiten Stufe ($X, Y \in \{1PT, REF\}$).
Beispiel: 1PT-1PT-Adaption steht für die bisher untersuchte Adaption.
Bei Verwendung der 1PT zur WWW-Auswahl und der Referenz auf der zweiten Stufe (1PT-REF-Adaption) konnte der beste BLEU-Score der 1PT-1PT-Adaption von 0.3384 bei Verwendung der INDRI.ngramWsum.FirstPT.2000-Anfrage nur von OKAPI-Anfragen der Größe 50-200 übertroffen werden. Der beste BLEU-Score liegt bei 0.3413 und ist damit schlechter als der Baseline-BLEU-Score von 0.3449. Neben den OKAPI-Anfragen schnitten auch die INDRI.simpleWsum-, INDRI.ngramWsum- und INDRI.ngramWsumUO-Anfragen gut ab. Mit der 1PT auf der 1. Stufe konnte also keine Verbesserung der Übersetzungsqualität erreicht werden.
Bei der Verwendung der Referenz zur Anfragegenerierung auf der 1. Stufe konnte der BLEU-Score im Vergleich zum Baselinesystem (1PT) sogar verbessert werden. Bei der REF-1PT-Adaption konnten mit OKAPI-Anfragen, die einen Korpus der Größe 50-200 Zeilen erzeugen sollten, BLEU-Scores von 0.3459 bis 0.3522 erreicht werden. Somit konnte die Übersetzungsleistung verbessert werden. Aus Tabelle 16 können weitere Details entnommen werden. Mit der REF-REF-Adaption konnte sogar ein BLEU-Score von 0.3539 erreicht werden. Hier lieferten die INDRI.ngramWsum-, INDRI.ngramWsumUO-, INDRI.simpleWsum- und OKAPI-Systeme die besten Ergebnisse. Dabei waren bei den INDRI-Systemen die Systeme mit mehr Zeilen (200-2000) erfolgreicher, während bei den OKAPI-Systemen diejenigen mit weniger Zeilen (50-200) am erfolgreichsten waren.

Diese Ergebnisse zeigen also, dass der Erfolg der Adaption stark von der Qualität der Daten aus dem WWW abhängt. Die durch die 1PT-Auswahl auf der 1. Stufe gefundenen Dokumente waren in unseren Versuchen nicht gut genug. Bei geeigneten Dokumenten aus dem WWW ist eine Adaption aber dennoch erfolgreich machbar, wie

Adaptionstyp	ALM-Gewicht	BLEU-Score
OKAPI.200	0.7	0.3459
OKAPI.100	0.7	0.3460
OKAPI.100	0.2	0.3462
OKAPI.50	0.4	0.3486
OKAPI.100	0.6	0.3487
OKAPI.50	0.6	0.3493
OKAPI.50	0.5	0.3508
OKAPI.50	0.7	0.3512
OKAPI.100	0.5	0.3522

Tabelle 16: Beste BLEU-Scores der REF-1PT-Adaption

die REF-1PT- und REF-REF-Versuche gezeigt haben.

5.5 Ergebnisse der externen Optimierung

In diesem Versuch werden die Gewichte für die einzelnen im Dekoder verwendeten Modelle optimiert. Dies geschieht basierend auf den N-Best-Listen der einzelnen Sätze, wobei extern über alle Sätze gleichzeitig optimiert wird. Dann wird aus den N-Best-Listen jeweils die mit den neuen Modellgewichten beste Übersetzungshypothese ausgewählt. In den Tabellen 17 und 18 werden für jedes Adaptionsmodell die besten BLEU-Scores (die mit optimalem Gewicht aus der Übersetzung mit festem Gewicht für das Adaptionsmodell) mit denen der Adaptionsmodelle nach der externen Optimierung verglichen. Die höchste Verbesserung des BLEU-Scores konnte für INDRI.combine.50 mit Gewicht 0.055 erreicht werden. Hier ergibt sich eine Verbesserung von 0.2925 auf 0.3290 (0.0365). Das ist eine Verbesserung um 12,48%. Im Durchschnitt konnten die BLEU-Scores um 0.0068 (2,19%) verbessert werden. Im schlechtesten Fall ergab sich keine Verbesserung. Der beste BLEU-Score nach externer Optimierung beträgt 0.3431 und liegt damit unter dem BLEU-Score der Übersetzung des Baseline-Systems (0.3449).

Adaptionstyp	Bester 2PT-BLEU-Score	Bester Score der externen Optimierung
groupTo.10	0.3042	0.3268
groupTo.20	0.3122	0.3308
groupTo.30	0.3105	0.3271
INDRI.combine.100	0.3203	0.3259
INDRI.combine.1000	0.3220	0.3236
INDRI.combine.200	0.3177	0.3342
INDRI.combine.2000	0.3213	0.3201
INDRI.combine.50	0.3182	0.3340
INDRI.combine.500	0.3152	0.3252

Tabelle 17: Verbesserung durch externe Optimierung (Teil 1)

Adaptionstyp	Bester 2PT-BLEU-Score	Bester Score der externen Optimierung
INDRI.syn.100	0.3120	0.3299
INDRI.syn.1000	0.3169	0.3341
INDRI.syn.200	0.3165	0.3236
INDRI.syn.2000	0.3210	0.3241
INDRI.syn.50	0.3117	0.3331
INDRI.syn.500	0.3178	0.3278
INDRI.ngramWsum.100	0.3290	0.3400
INDRI.ngramWsum.1000	0.3380	0.3388
INDRI.ngramWsum.200	0.3317	0.3402
INDRI.ngramWsum.2000	0.3384	0.3410
INDRI.ngramWsum.50	0.3317	0.3409
INDRI.ngramWsum.500	0.3373	0.3409
INDRI.ngramWsumUO.FirstPT.100	0.3299	0.3431
INDRI.ngramWsumUO.FirstPT.1000	0.3347	0.3410
INDRI.ngramWsumUO.FirstPT.200	0.3327	0.3378
INDRI.ngramWsumUO.FirstPT.2000	0.3379	0.3399
INDRI.ngramWsumUO.FirstPT.50	0.3334	0.3412
INDRI.ngramWsumUO.FirstPT.500	0.3375	0.3376
INDRI.simpleWsum.FirstPT.100	0.3318	0.3372
INDRI.simpleWsum.FirstPT.1000	0.3382	0.3422
INDRI.simpleWsum.FirstPT.200	0.3317	0.3375
INDRI.simpleWsum.FirstPT.2000	0.3367	0.3363
INDRI.simpleWsum.FirstPT.50	0.3327	0.3377
INDRI.simpleWsum.FirstPT.500	0.3374	0.3363
interpolated-10	0.3154	0.3319
interpolated-20	0.3170	0.3301
interpolated-30	0.3126	0.3251
OKAPI.100	0.3269	0.3343
OKAPI.1000	0.3194	0.3295
OKAPI.200	0.3210	0.3293
OKAPI.2000	0.3220	0.3274
OKAPI.50	0.3189	0.3376
OKAPI.500	0.3229	0.3319
TFIDF.100	0.3208	0.3305
TFIDF.1000	0.3202	0.3265
TFIDF.200	0.3175	0.3233
TFIDF.2000	0.3247	0.3311
TFIDF.score-10.00	0.3141	0.3200
TFIDF.score-15.00	0.3180	0.3296
TFIDF.score-20.00	0.3159	0.3330
TFIDF.score-25.00	0.3184	0.3348
TFIDF.50	0.3156	0.3358
TFIDF.500	0.3178	0.3225

Tabelle 18: Verbesserung durch externe Optimierung (Teil 2)

5.6 Orakelexperiment *Beste Sätze auswählen*

Mit den verschiedenen Adaptionstypen generieren wir verschiedene 2PT-Hypothesen. Mit diesem Experiment möchten wir untersuchen, wie gut die Übersetzung durch die generierten 2PT-Hypothesen verbessert werden könnte.
Dazu ersetzen wir sukzessive einen Satz aus der 1PT durch die entsprechende Hypothese aus den 2PT und evaluieren den gesamten Text. Auf diese Art können wir für jede 2PT-Hypothese die Auswirkung auf die Übersetzungsqualität ermitteln. Dadurch lassen sich die besten 2PT-Hypothesen bestimmen und somit die mit unserer Methode zu erzielende Verbesserung, wenn wir eine geeignete Auswahlregel für die 2PT-Hypothesen hätten. Für die Sätze, für die alle 2PT-Hypothesen schlechter sind als die 1PT, verwenden wir die 1PT als Übersetzung.
Das Baseline-System hat einen BLEU-Score der 1PT von 0.3449. Unter Verwendung der 1PT zur Datenauswahl auf der 1. Stufe (WWW) und der 1PT zur Datenauswahl auf der 2. Stufe (Lemur- und dokumentbasierte Auswahl) ergibt sich ein Oracle-BLEU-Score von 0.3778 (192 adaptierte Sätze), also eine Verbesserung des BLEU-Scores um 9,5%. Bei der Verwendung der Referenz zur Datenauswahl auf der zweiten Stufe ergibt sich sogar ein Oracle-BLEU-Score von 0.4034 (230 adaptierte Sätze). Dies entspricht einer Verbesserung um 17,0%.
Auffallend ist, dass die Orakelübersetzung kürzer ist als die 1PT. Im Durchschnitt waren die adaptierten Sätze um rund 1 Wort kürzer (1,04 Wörter bei 1PT-Auswahl, 0.97 Wörter bei REF-Auswahl). Somit haben sich die 2PT-Übersetzungen von der Länge der Referenzübersetzung entfernt. Die 1PT hat mit 3766 Wörtern fast genauso viele Wörter wie die Referenz mit 3782 Wörtern.
Wird die Referenz zur Datenauswahl auf der ersten Stufe verwendet, so ergibt sich eine geringere Verbesserung des BLEU-Scores (auf 0.3547 und 0.3564, vgl. Tabelle 19) als dies bei Verwendung der 1PT der Fall war. Es werden allerdings auch nur 51 bzw. 53 Sätze adaptiert.

Datenauswahl WWW	Datenauswahl 2. Stufe	Oracle-BLEU-Score	Verbesserung in %	Anzahl adaptierter Sätze	Länge der Übersetzung
1PT	1PT	0.3778	9,5%	192	3567
1PT	REF	0.4034	17,0%	230	3544
REF	1PT	0.3547	2,8%	51	3713
REF	REF	0.3564	3,3%	53	3709

Tabelle 19: Orakelexperiment *Beste Ergebnisse auswählen*: Ergebnisse

Aus den Daten konnte keine Regel zur Auswahl von guten 2PT-Hypothesen ermittelt werden.

5.7 Untersuchung der Sätze

Hier untersuchen wir die Sätze per Hand um zu sehen, was bei der Adaption passiert. Das Ziel dieser Untersuchung ist festzustellen, ob eine positive oder negative Veränderung im BLEU-Score auch subjektiv eine positive oder negative Veränderung der Übersetzungsqualität mit sich bringt.

Beispiel 1 In diesem Beispiel wird der Übersetzungsfehler von *lower leg* zu *calf* erfolgreich korrigiert und das 2-gram *i have* gefunden. Mit dieser Verbesserung geht auch eine Erhöhung des BLEU-Scores einher.

HYPOTHESE (2PT): i have easily you to be my best stiff the muscles of the calf

BASELINE-ÜBERSETZUNG (1PT): i do you bleed easily it the best stiff the muscles of the lower leg

REFERENZ: i have the tendency of getting stiff muscles in my calf

ÄNDERUNG IM BLEU-SCORE: 0.3449->0.3452

Beispiel 2 Wir schauen uns den Satz mit der höchsten Verbesserung an. Nicht alle Fehler konnten verbessert werden, aber der Satz wird verständlicher.

HYPOTHESE (2PT): if it hurts to touch any area and half an inch there it doesn't hurt it's a broken bones and need examination it a bone scan to make the diagnosis

BASELINE-ÜBERSETZUNG (1PT): if it hurts even if it touches the any area and half an inch don't it hurts it's a broken bone and need examination what we call a ct scan of bone to make the diagnosis

REFERENZ: if it hurts to touch one spot and you move a half an inch away and it doesn't hurt that is a crack bone and you need something called a bone scan to make that diagnosis

ÄNDERUNG IM BLEU-SCORE: 0.3449->0.3470

Beispiel 3 Der Satz mit der stärksten Verschlechterung ergibt auch subjektiv eine starke Verschlechterung.

HYPOTHESE (2PT): especially when i feel or when i am a tight

BASELINE-ÜBERSETZUNG (1PT): especially when i sit down or when i'm in a position tight

REFERENZ: especially when i sit down or when i'm in a cramped position

ÄNDERUNG IM BLEU-SCORE: 0.3449->0.3424

Beispiel 4 In diesem Beispiel konnte die Wortreihenfolge verbessert werden, der Übersetzungsfehler jedoch nicht.

HYPOTHESE (2PT): or two winding

BASELINE-ÜBERSETZUNG (1PT): two or winding

REFERENZ: or two calves

ÄNDERUNG IM BLEU-SCORE: 0.3449->0.3450

Beispiel 5 Häufig wird einfach ein Wort weggelassen und dadurch verbessert sich der BLEU-Score. Der Satz wird dadurch aber nicht immer besser.

HYPOTHESE (2PT): i know that i should have to the hospital

BASELINE-ÜBERSETZUNG (1PT): i know that i should have gone to the hospital

REFERENZ: i knew i should go to the hospital

ÄNDERUNG IM BLEU-SCORE: 0.3449->0.3450

Beispiel 6 Hier ein Negativbeispiel: Der BLEU-Score dieses Satzes ist höher, aber der adaptierte Satz kann nicht mehr verstanden werden, obwohl dies bei der 1PT noch halbwegs der Fall ist. Dies liegt daran, dass Scores zur automatischen Bewertung von maschineller Übersetzung nur Annäherungen an die Bewertung eines Satzes durch einen Menschen sind.

HYPOTHESE (2PT): temperature fluids moisture

BASELINE-ÜBERSETZUNG (1PT): take only room temperature fluids provide some moisture

REFERENZ: what you drink should be room temperature

ÄNDERUNG IM BLEU-SCORE: 0.3449->0.3450

5.8 Diskussion der BTEC-JapEng-System-Ergebnisse

Wir haben für die erfolgreichsten Adaptionstypen im BTEC+medical-SpaEng-System auch eine Adaption auf dem BTEC-JapEng-System durchgeführt und präsentieren hier die Ergebnisse.

INDRI.ngramWsum- und INDRI.simpleWsum-Adaptionstypen Sowohl bei den INDRI.ngramWsum- als auch bei den INDRI.simpleWsum-Adaptionstypen schneiden die Adaptionsmodelle mit kleiner Korpusgröße schlechter ab, als die mit größerem Korpus (vgl. Abbildung 14 und 15). Je stärker der Einfluss des Adaptionssprachmodells jedoch wird, desto mehr nähern sich die BLEU-Scores an. Die BLEU-Scores nehmen ihren besten Wert an, wenn das ALM-Gewicht gleich groß oder größer ist als das Gewicht des Baselinesprachmodells. Bei den INDRI.ngramWsum-Adaptionstypen ist dieser Punkt früher erreicht (ALM-Gewicht ca. 1,0) als bei den INDRI.simpleWsum-Adaptionstypen (ALM-Gewicht ca. 1,3-1,4). Beste Werte werden also erreicht, wenn der Einfluss des Adaptionssprachmodells größer ist als der des Baselinesprachmodells.

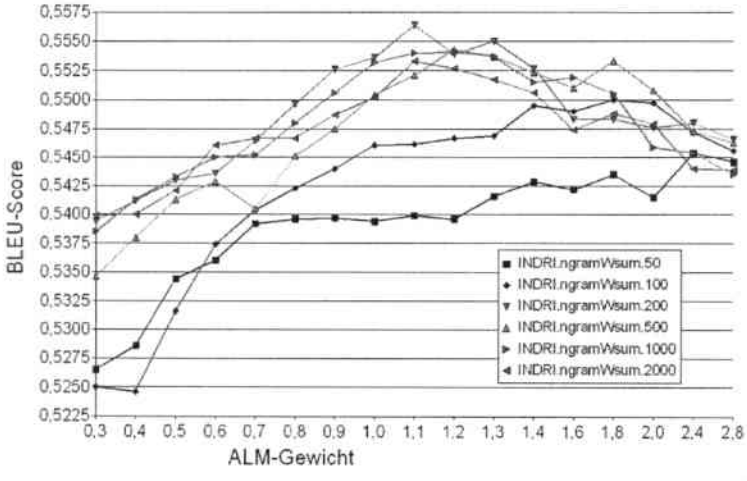

Abbildung 14: BLEU-Scores der INDRI.ngramWsum-Anfragen

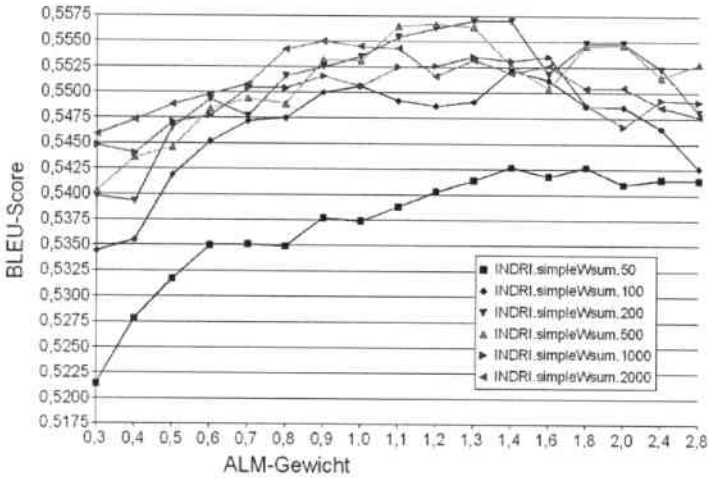

Abbildung 15: BLEU-Scores der INDRI.simpleWsum-Anfragen

TFIDF-und OKAPI-Adaptionstypen Bei den TFIDF-und OKAPI-Adaptionsmodellen sind es eher die Adaptionsmodelle mit größerem Korpus, die schlechter abschneiden (vgl. Abbildung 16 und 17). Eine Zunahme des BLEU-Scores bei höherem Einfluss des Adaptionssprachmodells vor Erreichen des Spitzenwertes ist zu erkennen.Die besten BLEU-Scores werden erreicht, wenn der Einfluss des Adaptionssprachmodells 1,5- 2-mal so groß ist, wie der des Baselinesprachmodells.

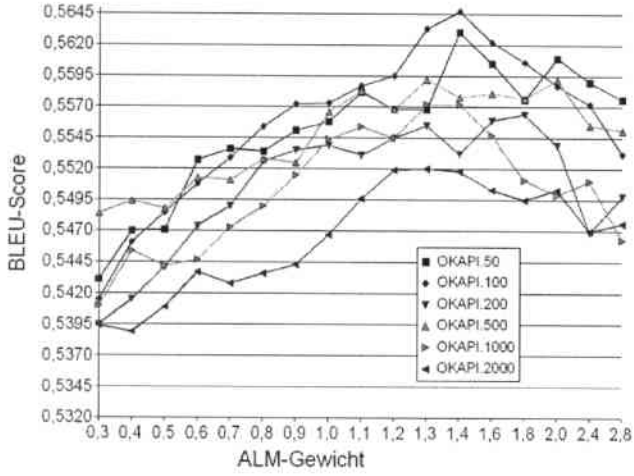

Abbildung 16: BLEU-Scores der OKAPI-Anfragen

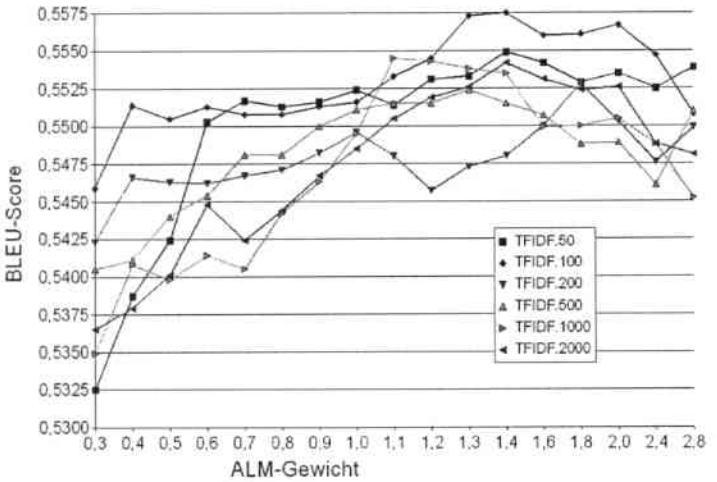

Abbildung 17: BLEU-Scores der TFIDF-Anfragen

TFIDF.score-Adaptionstypen Bei den TFIDF.score-Adaptionsmodellen lässt sich erkennen, dass die BLEU-Scores für die Adaptionsmodelle mit einem Mindest-TFDIF-Score von 20 bzw. 25 bei niedrigeren ALM-Gewichten schlechter sind als die anderen (vgl. Abbildung 18). Wird der Einfluss des Adaptionssprachmodells allerdings stärker als der des Baselinesprachmodells, so nähern sich die BLEU-Scores der TFIDFscore-25/20/15-Systeme an. Das TFIDFscore-10-System liefert hier die besten BLEU-Scores. Wie auch bei den anderen Adaptionstypen erhöht sich der BLEU-Score mit zunehmendem Einfluss des Adaptionssprachmodells bis zum Erreichen des Spitzenwertes des BLEU-Scores.

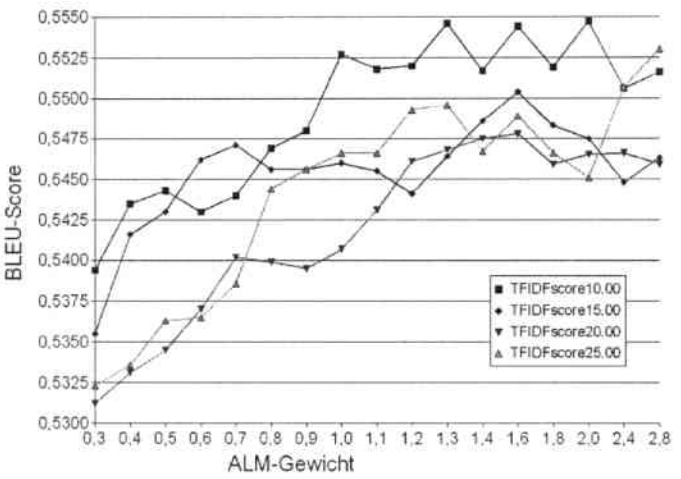

Abbildung 18: BLEU-Scores der TFIDFscore-Anfragen

Tabelle 20 gibt für jeden Adaptionstyp die besten BLEU-Scores an. Zusätzlich sind die Größe des zugrundeliegenden Korpus (bzw. der Mindest-Score bei dem TFIDFscore-Typ) und das ALM-Gewicht des Systems mit dem besten BLEU-Score angegeben. Keiner der Adaptionstypen ist besser als das Baselinesystem, das einen BLEU-Score von 0.5726 erreicht hat. Der Orakel-BLEU-Score, der sich aus den besten erzeugten Hypothesen errechnet, beträgt hier 0.6253. Zu beachten ist, das hier nur die TFIDF-, OKAPI-, INDRI.ngramWsum-, INDRI.simpleWsum- und TFIDF.score-Hypothesen beachtet wurden.

Adaptionstyp	Bester BLEU-Score	Korpusgröße (Zeilen)	ALM-Gewicht
OKAPI	0.5554	100	1.4
INDRI.simpleWsum	0.5542	200	1.3 und 1.4
TFIDF	0.5517	100	1.4
INDRI.ngramWsum	0.5496	200	1.1
TFIDFscore	0.5471	TFIDF-Score $\geq$ 10.00	2.0

Tabelle 20: Beste BLEU-Scores für das BTEC-JapEng-System

6 Zusammenfassung und Ausblick

6.1 Zusammenfassung

In dieser Studienarbeit haben wir die Möglichkeiten einer satzbasierten Sprachmodelladaption mit Hilfe des WWW untersucht. Dabei konnte mit unseren Systemen keine Verbesserung des BLEU-Scores erreicht werden. In den Orakelexperimenten unter Verwendung der Referenzübersetzung zur Anfragegenerierung hat sich aber gezeigt, dass eine webbasierte Sprachmodelladaption möglich ist, wenn die Daten aus dem WWW gut genug sind. Auch mit der 1PT-basierten Auswahl könnte der Score verbessert werden, wenn eine geeignete Regel zur Auswahl der guten Hypothesen bekannt wäere. Im Fall einer Orakelauswahl der besten Hypothesen konnte der BLEU-Score um 9,5% verbessert werden. Unsere Ergebnisse widersprechen also nicht denen aus [41] und [14]. Dort wird nur eine Auswahl auf unserer 2. Stufe durchgeführt und die verwendeten Korpora für die Satzauswahl sind den zu übersetzenden Sätzen ähnlich. Das WWW als Korpus enthält wesentlich mehr schlechte Daten, als dies in dem lokalen Korpus der Fall ist. Durch unsere Korpusgenerierung auf der 1. Stufe konnten wir keine Texte mit der 1PT finden, die genügend ähnliche sind. Es hat sich in unseren Experimenten gezeigt, dass die satzbasierte Auswahl besser abschneidet, als die dokumentbasierte.

6.1.1 Vor- und Nachteile des Ansatzes

Die Vorteile dieses Ansatzes bestehen darin, dass der Ansatz eine Anpassung des Sprachmodells auf Satzebene erlaubt. Dadurch kann das Sprachmodell spezifischer angepasst werden. Des weiteren wird das Test-Set zur Adaption benutzt. Somit ist kein Overfitting bezüglich anderer zum Training verwendeter Daten möglich. Ein weiterer wichtiger Aspekt ist, dass der Ansatz es erlaubt, neues Vokabular bei geeigneter Unterstützung durch das Übersetzungsmodell zu integrieren. Könnten z.B. mögliche Übersetzungen für OOV-Wörter ermittelt werden (etwa in Form eines Lexikons), die den Kontext aber nicht beachten, so kann diese Information durch das Adaptionssprachmodell mit Hilfe des WWW hinzugefügt werden. Dadurch kann unter den möglichen Übersetzungsalternativen ausgewählt werden.
Nachteilig ist, dass die Adaption erst bei Vorliegen des Test-Sets möglich ist. Dadurch wird die Zeit zu einem kritischen Faktor. Ob die Adaption in Echtzeit vorgenommen werden kann ist fraglich. Netzverzögerungen, beschränkte Bandbreite sowie die Verfügbarkeit des Netzes und der Suchmaschine können das Zeitverhalten negativ beeinflussen. Zudem ist dieser Ansatz zur Sprachmodelladaption nur bei Übersetzungssystemen mit zusätzlicher Infrastruktur möglich: Eine Internetanbindung mit hoher Bandbreite, bei zeitkritischen Anwendungen eine hohe Verfügbarkeit und eine hochwertige Suchmaschine sind nötig. Daher ist der Ansatz nicht oder nur eingeschränkt für mobile Geräte geeignet. Hinzu kommen die Kosten für den Internetzugang. Weiterhin können Fehler in der 1PT eine korrekte Anpassung verhindern und sogar die Ergebnisqualität verschlechtern.

6.2 Ausblick

Mit folgenden Arbeiten könnten weitere Verbesserungsmöglichkeiten untersucht werden:

- Es sind verschiedene Methoden entwickelt worden, wie durch webbasierte Ansätze Übersetzungen für OOV-Wörter ermittelt werden können [17], [22]. Auch

im Bereich des sprachübergreifenden IR sind dazu Arbeiten entstanden [10], [23]. Durch die Suche im WWW nach Texten mit den gefundenen Übersetzungen der OOV-Wörter kann das neue Vokabular auch in das Sprachmodell aufgenommen und so der Übersetzungsprozess in die richtige Richtung dirigiert werden.

- Auch für Wörter, für die die Übersetzung bekannt ist, die übersetzten Wörter aber nicht oft genug im Korpus vorkommen, kann das Sprachmodell mit Hilfe des WWW verbessert werden. Dies ist z.B. dann der Fall, wenn Wort-zu-Wort-Übersetzungen z.B. aus einem Lexikon vorliegen, aber keine Phrasen mit den OOV-Übersetzungen existieren. Weiterhin könnte untersucht werden, wie unbekannte Wörter bzw. Wörter, die nur ein Mal bzw. in geringer Anzahl im Trainingskorpus vorkommen (keine zuverlässige Schätzung), im Zusammenspiel mit dem Dekoder dennoch übersetzt werden können. Das Sprachmodell entscheidet dann, welche Übersetzung wahrscheinlicher ist.

- Die Anfrage an die Suchmaschine könnte mit verschiedenen Techniken erweitert werden, z.B. durch IR-Pseudo-Relevance-Feedback. Hierdurch wird eine spezifischere Anfrage generiert, so dass die Dokumente des Anfrageergebnisses noch besser zu dem zu übersetzenden Satz passen.

- Man könnte versuchen, den Fehler durch die 1PT zu vermeiden. Mit Hilfe eines bilingual vorliegenden Korpus könnten ähnliche Texte in der Quellsprache ermittelt und aus deren Pendant in der Zielsprache dann eine Anfrage an das WWW generiert werden. Der Vorteil besteht hier darin, dass die Übersetzung korrekt ist. Allerdings kann in diesem Fall nicht mehr auf OOV-Wörter eingegangen werden.

- Das Preprocessing der WWW-Dokumente könnte verbessert werden, um störende Bestandteile, wie z.B. Werbung, und unerwünschte Dokumente, z.B. Seiten eines Online-Shops, zu eliminieren.

- Neben den HTML-Seiten, die in dieser Studienarbeit als WWW-Dokumente genutzt wurden, könnte man auch Daten in anderen Dateiformaten, wie z.B. PDF- oder MS-Word-Dateien verwenden. PDF-Dateien haben häufiger bessere Daten und könnten insbesondere in speziellen Fällen, wie z.B. Vorlesungsübersetzungen erfolgreich eingesetzt werden. Allerdings kann es schwieriger sein diese zu finden und eine gute Textextraktion wird benötigt.

- Man könnte die Linkstruktur in HTML-Dokumenten ausnutzen und so noch mehr ähnliche Seiten zu bereits gefundenen Seiten ermitteln. Dies könnte dann gut funktionieren, wenn von Webseiten auf ähnliche Seiten verlinkt würde.

Literatur

[1] Das lemur-toolkit. *http://www.lemurproject.org/*. 4.3.4, 4.3.4

[2] L. Bahl, P. Brown, P. de Souza, and R. Mercer. A tree-based statistical language model for natural language speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 37(7). 2.4.3

[3] J. Bellegarda. Exploiting latent semantic information in statistical language modeling. *Proceedings of the IEEE, 88(8)*, pages 1279–1296, 2000. 4.2.1

[4] S. Besling and H.G. Meier. Language model speaker adaptation. *Eurospeech 1995, Madrid, Spain*, 1995. 4.2.1

[5] T.L. Booth. Probabilistic representation of formal languages. *IEEE Conference Record of the Tenth Annual Symposium on Switching and Automata Theory, Waterloo, Ontario*, pages 74–81, 1969. 2.4.3

[6] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 1998. 3.2.2

[7] P.F. Brown, S.A.D. Pietra, V.J.D. Pietra, and R.L. Mercer. The mathematics of machine translation: Parameter estimation. *Computational Linguistics*, pages 263–311, 1993. 2.3

[8] S.F Chen., K. Seymore, and R. Rosenfeld. Adaptation for language modeling using unnormalized exponential models. *International Conference on Acoustics, Speech and Signal Processing 1998, Seattle WA*, 1998. 4.2.1

[9] Stanley Chen, Douglas Beeferman, and Ronald Rosenfeld. Evaluation metrics for language models. 2.4.4

[10] P.-J. Cheng, J.-W. Teng, R.-C. Chen, J.-H. Wang, W.-H. Lu, and L.-F. Chien. Translating unknown queries with web corpora for cross-language information retrieval. *SIGIR '04*, pages 146–153, 2004. 6.2

[11] Renato DeMori and Marcello Federico. Language model adaptation. *Computational Models of Speech Pattern Processing, Keith Pointing (ed.), NATO ASI Series, Springer Verlag*, 1999. 4.2.1

[12] George Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. *Proceedings of the Human Language Technology Conference (HLT2002), SanDiego, USA*, March 2002. 2.6.5, 5.1

[13] Matthias Eck, Ian Lane, Nguyen Bach, Sanjika Hewavitharana, Muntsin Kolss, Bing Zhao, Almut Silja Hildebrand, Stephan Vogel, and Alex Waibel. The uka/cmu statistical machine translation system for iwslt. *Proceedings of IWSLT 2006, Kyoto, Japan*, 2006. 5.1

[14] Matthias Eck, Stephan Vogel, and Alex Waibel. Language model adaptation for statistical machine translation based on information retrieval. *LREC*, 2004. (document), 4.2.2, 6.1

[15] Rayid Ghani, Rosie Jones, and Dunja Mladenic. Mining the web to create minority language corpora. 4.2.3

[16] Michael Glöggler. *Suchmaschinen im Internet*. Springer Verlag, 2003. 3.2.2

[17] Fei Huang, Ying Zhang, and Stephan Vogel. Mining key phrase translations from web corpora. *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP), Vancouver, October 2005*, pages 483–490, 2005. 6.2

[18] R. Iyer and M. Ostendorf. Modeling long distance dependence in language: Topic mixtures vs. dynamic cache models. *IEEE Transactions on Speech and Audio Processing, SAP-7(1)*, pages 30–39, 1999. 4.2.1

[19] R. Iyer, M. Ostendorf, and M. Meteer. Analyzing and predicting language model improvements. *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 1997. 2.4.4

[20] David Janiszek, Renato DeMori, and Frederic Bechet. Data augmentation and language model adaptation. *IEEE International Conference on Acoustics, Speech and Signal Processing 2001, Salt Lake City, UT*, 2001. 4.2.1

[21] Viet Bac Le, Brigitte Bigi, Laurent Besacier, and Eric Castelli. Using the web for fast language model construction in minority languages. 4.2.3

[22] Chengye Lu, Yue Xu, and Shlomo Geva. Improving translation accuracy in web-based translation extraction. *Proceedings of NTCIR-6 Workshop Meeting, May 15-18, 2007, Tokyo, Japan*, 2007. 6.2

[23] W.-H. Lu, L.-F. Chien, and H.-J. Lee. Translation of web queries using anchor text mining. pages 159–172, 2002. 6.2

[24] M. Mahajan, D. Beeferman, and X. Huang. Improved topic-dependent language modeling using information retrieval techniques. *Proc. of the Int. Conf. on Acoustics, Speech, and Signal Processing. Phoenix, Mar., 1999.* 4.2.1, 4.2.2

[25] Edward K. O'Neil and James C. French. A description of the lamb web-derived language model builder. *Technical Report CS-2000-31*, 2000. 4.2.3

[26] Kishore Papineni, Salim Roukos, Tod Ward, and Wei-Jing Zhu. Bleu: a methos for automatic evaluation of machine translation. *Proceedings of the 40th Annual Meeting of the ACL, Philadelphia*, pages 311–318, July 2002. 2.6.4, 5.1

[27] Stephen E. Robertson, Steve Walker, Micheline Hancock-Beaulieu, Aarron Gull, and Marianna Lau. Okapi at TREC. In *Text REtrieval Conference*, pages 21–30, 1992. 3.2.1, 3.2.1

[28] Stuart Russell and Peter Norvig. *Artificial Intelligence - A Modern Approach (2nd edition)*. Prentice Hall, 2003. 2.4.3

[29] A. Salomaa. Probabilistic and weighted grammars. *Information and Control*, 15:529–544, 1969. 2.4.3

[30] Kristie Seymore and Ronald Rosenfeld. Using story topics for language model adaptation. *Proc. Eurospeech 1997, Rhodes, Greece*, 1997. 4.2.1

[31] A. Stolcke. Srilm – an extensible language modeling toolkit, 2002. 5.1

[32] Toshiyuki Takezawa, Eiichiro Sumita, Fumiaki Sugaya, Hirofumi Yamamoto, and Seiichi Yamamoto. Toward a broad-coverage bilingual corpus for speech translation of travel conversation in the real world. *LREC 2002 Third International Conference on Language Resources and Evaluation*, 1:147/152, 2002. 5.2.1, 5.2.2

[33] S. Vogel, Y. Zhang, F. Huang, A. Tribble, A. Venugopal, B. Zhao, and A. Waibel. The cmu statistical machine translation system, 2003. 5.1

[34] Stephan Vogel. Smt decoder dissected: Word reordering. *Proceedings of NLP-KE 2003, Beijing, China*, 2003. 5.1

[35] Stephan Vogel. Pesa: Phrase pair extraction as sentence splitting. *Proceedings of MTSummit X, Phuket, Thailand*, 2005. 2.3, 5.1

[36] Stephan Vogel, Hermann Ney, and Christoph Tillmannn. Hmm-based word alignment in statistical translation. *The Proceedings of the 16th International Conference on Computational Linguistics*, pages 836–841, 1996. 2.3

[37] C. Zhai. Notes on the lemur tfidf model. *Unpublished report*, 2001. 3.2.1, 3.2.1

[38] Ying Zhang and Stephan Vogel. An efficient phrase-to-phrase alignment model for arbitrarily long phrases and large corpora. *Proceedings of EAMT 2005, Budapest, Hungary*, 2005. 5.1

[39] Ying Zhang and Stephan Vogel. Suffix array and its applications in empirical natural language processing. Technical Report CMU-LTI-06-010, Language Technologies Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, Dec. 2006. 2.4.1

[40] Ying Zhang, Stephan Vogel, and Alex Waibel. Integrated phrase segmentation and alignment algorithm for statistical machine translation. 2003. 2.3

[41] Bing Zhao, Matthias Eck, and Stephan Vogel. Language model adaptation for statistical machine translation with structured query models. *Coling 2004*. 4.2.1, 4.2.2, 6.1

[42] Xiaojin Zhu and Ronald Rosenfeld. Improving trigram language modeling with the world wide web. 4.2.3