Rheinisch Westfälische Technische Hochschule Aachen
Lehrstuhl für Informatik VI
Prof. Dr.-Ing. Hermann Ney

Seminar "speech recognition and language processing SS 2003"

# Wortfehlerminimierung

*Torsten Palm*

Matrikelnummer 219 815

Juli 2003

Betreuer: Dr. Ralf Schlüter

# Inhaltsverzeichnis



# Tabellenverzeichnis

# Abbildungsverzeichnis

# 1 Einführung

In der heutigen Zeit gewinnen Spracherkennungssysteme immer stärker an Bedeutung. Sie sind aus der häufig anzutreffenden Notwendigkeit der Mensch-Maschine-Kommunikation nicht mehr wegzudenken. Um diese Kommunikation möglichst exakt betreiben zu können ist eine möglichst korrekte Zuordnung gesprochener Signale zu einem konkreten Satz nötig. Bei der Findung des treffendsten Satzes wird das Prinzip der Mustererkennung zugrundegelegt.
Die allgemeine Form der Mustererkennung arbeitet nach dem Bayesschen Entscheidungskriterium. In diesem wird berechnet welcher Klasse eine gemachte Beobachtung zugeordnet wird, damit diese Zuordnung lediglich minimal von der zur Beobachtung passenden Klasse abweicht. Das Abweichungsmaß ist dabei anwendungsabhängig frei definierbar. In der Spracherkennung wird die Abweichung, welche auch als Kosten bezeichnet wird, in der Regel durch den Satzfehler bestimmt, d.h. die "richtige" Klasse der Beobachtung (=gesprochener Satz) und die zugeordnete Klasse (=erkannter Satz) werden auf Gleich- bzw. Ungleichheit geprüft. Als erkannten Satz erhalten wir somit den Satz mit der geringsten Erwartung einer Abweichung vom gesprochenen Satz. Wenn der Mensch allerdings zwei Sätze vergleicht so interessiert ihn meist nicht die reine Gleich- bzw. Ungleichheit der Sätze sondern ihn interessiert die gesamte Anzahl an abweichenden Worten zwischen den Sätzen, der Wortfehler. Der dahingehend optimale erkannte Satz ist der mit der minimal zu erwartenden Anzahl an Wortfehlern, was uns zum Thema der Ausarbeitung, der Wortfehlerminimierung führt. Um den Wortfehler zu minimieren müssen neue Entscheidungskriterien erstellt werden, womit sich die folgende Ausarbeitung beschäftigt.
Zu Beginn der Ausarbeitung wird die Grundlage für das Erstellen von Entscheidungskriterien, das Prinzip der Mustererkennung im Allgemeinen und die Einführung eines sogenannten Kostenmaßes, wodurch die Optimierungsrichtung des Kriteriums vorgegeben wird, vorgestellt. Danach werden die beiden für die Spracherkennung wichtigen Entscheidungskriterien zur Satz- bzw. Wortfehlerminimierung hergeleitet und anhand eines Beispieles miteinander auf den jeweils als Ergebnis gelieferten Satz verglichen. Es wird klar, daß die Satzfehlerminimierung nicht zwangsläufig den Satz mit den wenigsten Wortfehlern liefert, daher ist ein explizites Betrachten des Kriteriums zur Wortfehlerminimierung nötig um Wortfehlerminimierung zu betreiben. Dabei werden wir Probleme in der Rechenkomplexität bei der Durchführung des dieses Kriteriums feststellen. Schliesslich werden 3 Ansätze, der n-best-Listen-Ansatz, der word-lattice-consensus-Ansatz und der "time-frame-error"-Ansatz vorgestellt, in welchen durch unterschiedliche Ansätze die Effizienzprobleme gelöst werden. Zum Schluss werden diese 3 Verfahren miteinander verglichen und es wird abschließend ein kurzer Ausblick auf Forschungsansätze gegeben, die die Performance der vorgestellten Ansätze weiter verbessern könnten.

# 2 Mustererkennung im Allgemeinen

In diesem Abschnitt wird das Prinzip der Mustererkennung, welches die Basis für unsere späteren Überlegungen zur Herleitung neuer Entscheidungskriterien

zur Wortfehlerminimierung bildet, erklärt. Zuerst wird das Bayessche Entscheidungskriterium vorgestellt, welches die Grundlage für jede Mustererkennung darstellt. Dabei wird insbesondere darauf eingegangen inwieweit die Wahl des verwandten Kostenmaßes variierbar ist, da diese Möglichkeit der Variation die Basis für unsere späteren Überlegungen bildet.

## 2.1 Bayessches Entscheidungskriterium

Die Grundlage für die Zuordnung einer Beobachtung $x$ zu einer optimalen Klasse $c$ bildet das Bayessche Entscheidungskriterium.
Für jedes Element $c$ einer Hypothesenmenge $C$ der möglichen Zuordnungsklassen werden die zu erwartenden Kosten berechnet die entstehen, wenn man $x$ dieser Klasse $c$ zuordnet. Da man als Vergleichspartner zu $c$ kein konkretes "richtiges" Muster $k$ verwenden kann, da dieses nicht bekannt ist, wird in Bezug auf die Kosten ein Erwartungswert berechnet. Dabei werden alle Klassen $k \in C$ als mögliche richtige Klassen angesehen, jede dieser Klassen wird in Bezug auf entstehende Kosten mit der Klasse c verglichen und die jeweiligen Kosten $L(c,k)$ werden durch $p(k|x)$, die Wahrscheinlichkeit, daß $k$ die richtige Klasse für die Beobachtung $x$ ist, gewichtet und aufsummiert. Die Hypothese $c$ mit den geringsten zu erwartenden Kosten wird als die optimale Zuordnung für $x$ angesehen.

$$E_L(c) = \sum_{k \in C} p(k|x) \cdot L(c,k) \tag{1}$$

$$c_{opt} = \underset{c \in C}{\operatorname{argmin}}\{E_L(c)\} \tag{2}$$

Dabei ist die Kostenfunktion $L(c,k)$ so allgemein gehalten, daß sie anwendungsabhängig in Hinblick darauf, welche Kosten bei der Zuordnung minimiert werden sollen, variiert werden kann. Dieser Gedanke wird im nächsten Abschnitt verdeutlicht.

## 2.2 Kostenfunktionen

Die Wahl der Kostenfunktion $L(c,k)$ kann beliebig erfolgen und sie ist somit anwendungsabhängig frei definierbar. Bei der Definition der Kostenfunktion ist zu beachten, welche Art von Kosten durch das Entscheidungskriterium minimiert werden soll. Legt man als Kosten die simple Entscheidung zwischen Gleichheit und Ungleichheit zweier Klassen zugrunde, so bietet sich eine Definition mit Hilfe der Kroneckerfunktion $\delta(c,k)$an. Die Kosten sollen bei Gleichheit zwischen $c$ und $k$ 0 sein und bei Ungleichheit 1. Somit definiert sich die Kostenfunktion wie folgt:

$$L(c,k) = 1 - \delta(c,k) = \begin{cases} 0 & f\ddot{u}r \quad c = k \\ 1 & f\ddot{u}r \quad c \neq k \end{cases} \tag{3}$$

Eingesetzt als Kostenfunktion in das Bayessche Entscheidungskriterium (1) ergibt sich daraus

$$\begin{aligned} c_{opt} &= \underset{c\in C}{\operatorname{argmin}}\{\sum_{k\in C} p(k|x)\cdot L(c,k)\} \\ &= \underset{c\in C}{\operatorname{argmin}}\{1-p(c|x)\} \\ &= \underset{c\in C}{\operatorname{argmax}}\{p(c|x)\} \end{aligned} \tag{4}$$

Die Beobachtung $x$ wird somit einer Klasse $c$ zugeordnet, die die Wahrscheinlichkeit auf eine Abweichung jeglicher Art von der "richtigen" Klasse minimiert, die also unter der Beobachtung $x$ die maximale posteriori-Wahrscheinlichkeit besitzt.
In vielen anderen Fällen ist es allerdings nicht Ziel der Mustererkennung zwischen exakter Gleichheit und Ungleichheit zu unterscheiden, sondern andere Kostenaspekte fließen mit ein. Stellt man sich z.B. ein Authentifizierungssystem für einen Hochsicherheitstrakt vor, dann könnten die möglichen Klassenzuteilungen für eine Eingabe von Authentifizierungsdaten wie folgt aussehen:

- $a$ :="Zugang erteilt mit Zugangsberechtigung"
- $b$ :="Zugang nicht erteilt mit Zugangsberechtigung"
- $c$ :="Zugang erteilt ohne Zugangsberechtigung"
- $d$ :="Zugang erteilt mit Zugangsberechtigung"

Bei der Festlegung der Kosten zwischen zwei der Klassenzuteilungen ist es in diesem Falle wichtig, die Dimension des gemachten Fehlers herauszustellen. Die Kosten für den Entscheid für c statt a beispielsweise ist gravierend, da es unberechtigten Zutritt bedeutet. Ein Entscheid für b statt für a bedeutet hingegen lediglich die Notwendigkeit einer neuen Authentifizierung. Entsprechend sollten die Kosten für die erste Fehlzuteilung auch deutlich höher ausfallen als für die zweite. Dieses Beispiel zeigt, daß eine Kostenfunktion durchaus auch komplexerer Art sein kann.

# 3 Entscheidungskriterien in der Spracherkennung

Die unter Abschnitt 2 getätigten Überlegungen zum Thema Mustererkennung werden im folgenden Abschnitt auf die statistische Spracherkennung, insbesondere die beiden Kostenmaße des Satzfehlers bzw. des Wortfehlers spezifiziert. Dies führt uns zu den beiden Entscheidungskriterien zur Satz- bzw. Wortfehlerminimierung, welche am Schluss des Abschnittes anhand eines Beispiels in Bezug auf das jeweils gelieferte Ergebnis verglichen werden.
Um zu verdeutlichen, daß es sich im weiteren Verlaufe der Ausarbeitung bei den gegenübergestellten Klassen um Sätze handelt, die natürlicherweise aus Wortfolgen bestehen, wird ab jetzt die Schreibweise $w_1^N$ bzw. $v_1^M$ statt $c$ und $k$ verwandt um Sätze der Länge $N$ bzw $M$ darzustellen, wobei die $w_i$ und $v_j$ das jeweilige Wort an Position $i$ bzw. $j$ des Satzes angeben. Die verglichenen

Sätze müssen dabei nicht zwangsläufig dieselbe Länge haben. Die Beobachtung $x$, nach der ausgewertet wird, ist eine Folge von akustischen Vektoren $x_1^T$, wobei $t$ angibt zu welchem Zeitpunkt des gesprochenen Satzes die Beobachtung $x_t$ gemacht wurde. Das grundliegende Entscheidungskriterium zur Spracherkennung ist somit

$$\{w_1^N\}_{opt} = \underset{N,w_1^N}{\operatorname{argmin}}\{\sum_{M,v_1^M} p(v_1^M|x_1^T) \cdot L(w_1^N, v_1^M)\} \tag{5}$$

## 3.1 Satzfehlerminimierung

In diesem Abschnitt wird das Entscheidungskriterium zur Satzfehlerminimierung hergeleitet. Wie in Abschnitt 1 bereits erwähnt wird dieses Entscheidungskriterium in den gängigen Spracherkennungssystemen verwandt. Als Kostenfunktion hierzu dient der Satzfehler [1]. Das bedeutet, daß zwei Sätze $w_1^N$ und $v_1^M$ durch die Kostenfunktion $L(w_1^N, v_1^M) = 1 - \delta(w_1^N, v_1^M)$ miteinander auf Gleichheit überprüft werden, wobei $\delta(w_1^N, v_1^M)$ die Kroneckerfunktion darstellt (vgl. Formel (3)). Es bietet sich an, die Kroneckerfunktion für Sätze konkret zu definieren:

$$\delta(w_1^N, v_1^M) = \begin{cases} 1 & \textit{für} \quad N = M \quad \textit{und} \quad \forall 1 \leq i \leq N \quad w_i = v_i \\ 0 & \textit{sonst} \end{cases} \tag{6}$$

Da die Kostenfunktion durch die Wahl des Satzfehlers nun spezifiziert ist kann das Bayessche Entscheidungskriterium (vgl. Formel (5)) für den Fall der Satzfehlerminimierung bei der Spracherkennung ausformuliert werden [1].

$$\begin{aligned}
\{w_1^N\}_{opt} &= \underset{N,w_1^N}{\operatorname{argmin}}\{\textstyle\sum_{M,v_1^M} L(w_1^N, v_1^M) \cdot p(v_1^M|x_1^T)\} \\
&= \underset{N,w_1^N}{\operatorname{argmin}}\{\textstyle\sum_{M,v_1^M} (1 - \delta(w_1^N, v_1^M)) \cdot p(v_1^M|x_1^T)\} \\
&= \underset{N,w_1^N}{\operatorname{argmin}}\{1 - p(w_1^N|x_1^T)\} \\
&= \underset{N,w_1^N}{\operatorname{argmax}}\{p(w_1^N|x_1^T)\} \\
&= \underset{N,w_1^N}{\operatorname{argmax}}\{\frac{p(x_1^T|w_1^N) \cdot p(w_1^N)}{p(x_1^T)}\} \\
&= \underset{N,w_1^N}{\operatorname{argmax}}\{p(x_1^T|w_1^N) \cdot p(w_1^N)\}
\end{aligned} \tag{7}$$

$p(x_1^T|w_1^N)$ ist hierbei das Ergebnis des optimalen Durchlaufpfades der Beobachtungen $x_1^T$ durch das HMM des Satzes $w_1^N$ (akkustisches Modell) und $p(w_1^N)$ gibt an, wie wahrscheinlich die Abfolge der Worte $p(w_1^N)$ ist (Sprachmodell). Als Ergebnis wird derjenige Satz ausgegeben, bei dem die Wahrscheinlichkeit daß im Vergleich zum gesprochenen Satz ein Satzfehler vorliegt am geringsten ist, d.h. dessen Satz-posteriori-Wahrscheinlichkeit maximal ist. Dieser Ansatz

hat den Vorteil daß er sehr leicht berechenbar ist und somit einfach umzusetzen ist.
Die Findung des optimalen Satzes erfolgt also mit Hilfe des Bayesschen Entscheidungskriteriums durch die Optimierung des erwarteten Satzfehlers über alle möglichen Sätze.
Eine zweite Art von einem möglichen Kostenmaß in der Spracherkennung, der Wortfehler, wird im nächsten Abschnitt eingeführt.

## 3.2 Wortfehlerminimierung

Wenn der Mensch zwei Sätze auf Ähnlichkeit überprüft, so betrachtet er in der Regel die beiden Sätze nicht nur in Hinblick auf das Auftreten eines Fehlers, sondern ihn interessiert an wievielen Positionen die beiden Sätze nicht übereinstimmen. Eine Minimierung der Anzahl dieser Fehler bei der Findung des Satzes, dem die akustische Beobachtung optimalerweise zugeordnet wird, ist somit ebenfalls ein sinnvolles Kriterium welches in der Spracherkennung angewandt wird. In diesem Abschnitt wird somit ein Entscheidungskriterium zur Wortfehlerminimierung hergeleitet. Als Kostenfunktion wird hierbei die Anzahl an Wortfehlern nach Levensthein, deren Berechnung kurz vorgestellt wird, herangezogen.

### 3.2.1 Levensthein-Anpassung

Um die Anzahl an Wortfehlern zwischen zwei Sätzen zu bestimmen interessiert die Anzahl der Positionen an denen sich die Sätze unterscheiden. Da der Begriff der Position im Satz aber problematisch ist, weil die verglichenen Sätze z.B. nicht unbedingt gleich lang sind, ist dieser Vergleich nicht so trivial wie es sich zuerst anhören mag [4]. Die genaue Bestimmung des Wortfehlers zwischen 2 Sätzen ist in der sogenannten Levensthein-Anpassung definiert. Diese funktioniert nach folgendem Prinzip: Die einzelnen Worte in den zu vergleichenden Sätzen werden zuerst einander so zugeordnet, daß die Reihenfolge der Worte in $w_1^N$ genau der Reihenfolge der jeweils zugeordneten Worte in $v_1^M$ entspricht. Dabei werden alle dieser möglichen Zuordnungen durchgeführt und diejenige, bei der die Anzahl an insertions/deletions (d.h. Anzahl der nicht zugeteilten Worte in $w_1^N$ bzw. $v_1^M$) und substitutions (zwei unterschiedliche Worte sind einander zugeteilt) minimal ist wird durchgeführt. Die Anzahl der bei der durchgeführten Zuordnung auftretenden Fehlern wird als die Anzahl der Wortfehler verstanden. [4].

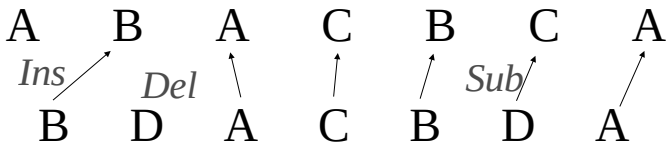


Abbildung 1: Prinzip der Levensthein-Anpassung (optimale Zuordnung)

In dem Beispiel beträgt der Wortfehler also 3. Allgemein wird in der folgenden Ausarbeitung die Anzahl an Wortfehlern nach Levensthein zwischen zwei Sätzen $w_1^N$ und $v_1^M$ mit $\mathcal{L}(w_1^N, v_1^M)$ bezeichnet.
Explizit durchgeführt wird in dem Algorithmus jede Anpassungsmöglichkeit berechnet und die Anpassung mit minimalem Fehler wird durchgeführt. Dieses Vorgehen hat eine exponentielle Rechenkomplexität, welche durch dynamische Programmierung auf die Komplexitätsklasse $O(M \cdot N)$ reduziert werden kann.

### 3.2.2 Entscheidungskriterium zur Wortfehlerminimierung

In diesem Abschnitt wird der unter Abschnitt 3.2.1 vorgestellte Begriff des Wortfehlers in die Grundform des Bayesschen Entscheidungskriteriums (Formel (5)) eingesetzt und somit das Bayessche Entscheidungskriterium zur Wortfehlerminimierung hergeleitet.
Wenn wir nun davon ausgehen, daß $L(w_1^N, v_1^M) = \mathcal{L}(w_1^N, v_1^M)$, dann ergibt sich folgendes Entscheidungskriterium:

$$\{w_1^N\}_{opt} = \underset{N, w_1^N}{\operatorname{argmin}}\{ \sum_{M, v_1^M} p(v_1^M | x_1^T) \cdot \mathcal{L}(w_1^N, v_1^M)\} \tag{8}$$

Eine weitere Vereinfachung des Kriteriums ist bisher nicht bekannt, da der Wortfehler zwischen zwei Sätzen bisher nur durch ein explizites Durchführen der Levensthein-Anpassung berechnet werden kann. Ein Anwenden des Kriteriums setzt voraus, daß für alle Satzhypothesen eine paarweise Anpassung durchgeführt wird und für jede Hypothese die so erhaltenen Wortfehler gewichtet und aufsummiert werden.

## 3.3 Satzfehlerminimierung vs. Wortfehlerminimierung

In diesem Abschnitt werden die bisher vorgestellten Entscheidungskriterien der Satz- bzw. Wortfehlerminimierung anhand eines Beispieles angewandt und in Hinblick auf das jeweils gelieferte Ergebnis verglichen. Die Frage die es zu klären gilt ist, ob die Satzfehlerminimierung automatisch eine Wortfehlerminimierung impliziert oder ob zum Erreichen eines minimalen Wortfehlers explizit das Kriterium zur Wortfehlerminimierung angewandt werden muss.
Als Beispiel dient uns ein Hypothesenraum über einem Vokabular $\Sigma = \{a, b\}$,

der die möglichen Sätze der Länge 1 bzw. 2 enthält. Diese Hypothesen werden zuerst paarweise durch die Levensthein-Anpassung auf Wortfehler überprüft.

Tabelle 1: Wortfehler zwischen zwei Satzhypothesen nach Levensthein

| $\mathcal{L}(v_1^M, w_1^N)$ | ab | ba | a | b |
|---|---|---|---|---|
| ab | 0 | 2 | 1 | 1 |
| ba | 2 | 0 | 1 | 1 |
| a | 1 | 1 | 0 | 1 |
| b | 1 | 1 | 1 | 0 |

Mit Hilfe dieser Werte und den Satz-posteriori-Wahrscheinlichkeiten der Hypothesen lassen sich nun die beiden Entscheidungskriterien durchführen:

Tabelle 2: Durchführung des Satz- bzw. Wortfehlerminimierung-Kriteriums

| $w_1$ | $w_2$ | $p(w_1w_2\|x_1^T)$ | $1 - p(w_1w_2\|x_1^T)$ | $\sum_{v_1^M} p(v_1^M\|x_1^T) \cdot \mathcal{L}(v_1^M, w_1w_2)$ |
|---|---|---|---|---|
| a | b | 0.35 | 0.65 | $0 \cdot 0.35 + 1 \cdot 0.2 + 1 \cdot 0.15 + 2 \cdot 0.3 = 0.95$ |
| a | □ | 0.2 | 0.8 | $1 \cdot 0.35 + 0 \cdot 0.2 + 1 \cdot 0.15 + 1 \cdot 0.3 = 0.8$ |
| □ | b | 0.15 | 0.85 | $1 \cdot 0.35 + 1 \cdot 0.2 + 0 \cdot 0.15 + 1 \cdot 0.13 = 0.85$ |
| b | a | 0.3 | 0.7 | $2 \cdot 0.35 + 1 \cdot 0.2 + 1 \cdot 0.15 + 0 \cdot 0.3 = 1.05$ |

Die vierte Spalte liefert uns die jeweils mittlere Zahl an Satzfehlern, gleichbedeutend mit der Wahrscheinlichkeit eines Satzfehlers und die fünfte Spalte liefert die jeweils mittlere Anzahl an Wortfehlern. Demnach ergeben sich folgende Resultate:

- Das Kriterium zur Satzfehlerminimierung liefert den Satz "ab" als Ergebnis mit der Wahrscheinlichkeit eines Satzfehlers von $0,65$
- Das Kriterium zur Wortfehlerminimierung liefert den Satz "a" als Ergebnis mit einer mittleren Zahl an Wortfehlern von $0,8$

Es wird also deutlich, daß die Satzfehlerminimierung nicht zwangsläufig die Anzahl an Wortfehlern minimiert. Um unserem Ziel der Wortfehlerminimierung gerecht zu werden müssen wir somit explizit Wortfehlerminimierung betreiben. Im nächsten Abschnitt wird daher das bereits vorgestellte Entscheidungskriterium (Formel (8)) zur Wortfehlerminimierung genauer untersucht und es werden weitere Ideen zur Wortfehlerminimierung vorgestellt.

# 4 Explizite Wortfehlerminimierung

In Abschnitt 3.3 haben wir gesehen, daß es um Wortfehlerminimierung zu betreiben notwendig ist, das Entscheidungskriterium zur Wortfehlerminimie-

rung explizit anzuwenden. In diesem Abschnitt werden wir daher das in Abschnitt 3.2.2 vorgestellte Entscheidungskriterium zur Wortfehlerminimierung genauer untersuchen. Dabei werden wir Probleme in der Rechenkomplexität bei Durchführung des Kriteriums feststellen. Um diese Probleme zu lösen werden drei Ansätze entwickelt, mit denen Wortfehlerminimierung effizient durchführbar ist. Dabei handelt es sich um den n-best-Listen-Ansatz, bei dem durch Einschränkung des Hypothesenraumes die Rechenzeit begrenzt wird und um den word-lattice-consensus-Ansatz bzw. den "time frame error"-Ansatz, welche jeweils durch Annäherung des Wortfehlers diesen mit geringerer Komplexität berechnen und somit die Rechenzeit in Grenzen halten.

## 4.1 Entscheidungskriterium zur Wortfehlerminimierung

Bei der Wortfehlerminimierung in der Spracherkennung liegt wie in Abschnitt 3.2.2 hergeleitet wurde folgende Formel zugrunde:

$$\{w_1^N\}_{opt} = \underset{N,w_1^N}{\operatorname{argmin}}\{\sum_{M,v_1^M} p(v_1^M|x_1^T) \cdot \mathcal{L}(w_1^N, v_1^M)\}$$

$\mathcal{L}(w_1^N, v_1^M)$ gibt hierbei den Wortfehler an, der durch die Levensthein-Anpassung berechnet wird. Wie in Abschnitt 3.2.1 erklärt liegt die Komplexität der Berechnung des Wortfehlers nach Levensthein in $O(M \cdot N)$ wenn die beiden gegenübergestellten Hypothesen die Länge $M$ bzw. $N$ haben, im Entscheidungskriterium kommt bei der Bildung der erwarteten Kosten für eine Hypothese ein Aufsummieren über den gesamten Hypothesenraum hinzu. Insgesamt erhält man also bei einem Betrachten von Sätzen der Länge 5 über einer Anzahl von 10.000 Worten einen Rechenaufwand von $10.000^5 \cdot 5 \cdot 5$ für das Entscheidungskriterium. Dieser Rechenaufwand ist in der Regel inakzeptabel, so daß Überlegungen angestellt werden müssen die Komplexität zu verringern. Erster und einfachster Ansatz ist die Limitierung des Hypothesenraumes, ähnlich wie in unserem Beispiel in Abschnitt 3.3 die Beschränkung auf Sätze der Länge 1 bzw. 2. Dieser Ansatz wird im n-best-Listen-Ansatz verfolgt.

## 4.2 Wortfehlerminimierung über n-best-Listen

Eine n-best-Liste ist eine Liste von den n Sätzen, die die maximale Satzposteriori-Wahrscheinlichkeiten in Bezug auf die gemachte Beobachtung $x_1..x_T$ haben. Die Idee ist, daß die Einschränkung des Hypothesenraumes eine Durchführbarkeit der Levensthein-Anpassung mit anschliessender Bildung der erwarteten Wortfehler für jede Hypothese ermöglicht [1]. Dies liegt daran, daß mit der Einschränkung der Größe des Hypothesenraumes auch der Rechenaufwand des Kriteriums (8) eingeschränkt wird und es somit durchführbar ist.

### 4.2.1 Herleitung des Entscheidungskriteriums

Die Idee des n-best-Listen-Ansatzes basiert auf dem bereits bekannten Entscheidungskriterium (8) zur Wortfehlerminimierung. Neu ist nur, daß sowohl Minimierung wie auch Erwartungswertbildung ausschliesslich über die Elemente

der n-best-Liste erfolgen. Wenn man diesen Aspekt im Entscheidungskriterium berücksichtigt entsteht [1]

$$w_1^N{}_{opt} = \underset{N,w_1^N \in n-best}{\operatorname{argmin}} \{ \sum_{M,v_1^M \in n-best} p(v_1^M|x_1^T) \cdot \mathcal{L}(w_1^N, v_1^M) \} \tag{9}$$

Das explizite Durchführen der paarweisen Satzanpassung sowie das explizite Aufsummieren der einzelnen Kosten ist hierbei weiterhin notwendig. Ausserdem müssen die Satz-posteriori-Wahrscheinlichkeiten $p(w_1^N|x)$ der Hypothesen $w_1^N$ neu berechnet werden, weil die Zahl der konkurrierenden Hypothesen geschrumpft ist und somit die einzelnen Hypothesen wahrscheinlicher geworden sind. Dies geschieht wie bei der Herleitung des Kriteriums zur Satzfehlerminimierung unter Abschnitt 3.1 durch die Zerlegung in akustisches Modell bzw. Sprachmodell.

$$p(w_1^N|x_1^T) = \frac{p(x_1^T|w_1^N)^{1/\lambda} \cdot p(w_1^N)}{\sum_{M,v_1^M \in n-best} p(x_1^T|v_1^M)^{1/\lambda} \cdot p(v_1^M)} \tag{10}$$

Neu hierbei sind die normalisierende Division durch $p(x)$sowie die Einführung des Exponenten $1/\lambda$, die bei der Herleitung von (7) der Einfachheit halber vernachlässigt wurden [1]. Durch die Normalisierung erreicht man, daß der berechnete Wert eine Wahrscheinlichkeit darstellt, die durch ihre Normierung mit jeglichen anderen Wahrscheinlichkeiten vergleichbar ist und nicht nur beim Vergleich mit in gleichem Maße unnormierten Wahrscheinlichkeiten aussagekräftig ist. Die Relativierung des akkustischen Modells durch den Exponenten $1/\lambda$ ist sinnvoll, da es sich dabei lediglich um einen Schätzwert handelt, der verglichen mit der Schätzung des Sprachmodells in der Regel schlechter ausfällt. Man erreicht dadurch erfahrungsgemäß bessere Ergebnisse.

### 4.2.2 Performance

Die in der folgenden Tabelle enthaltenen Ergebnisse entstammen einer von A. Stolcke, Y. Knig und M. Weintraub durchgeführten Versuchsreihe [1]. Die darin verwendeten corpora Switchboard und CallHome Spanish sind wie folgt aufgebaut:

- **Switchboard** ist ein corpus der 2400 Telefongespräche zwischen 543 Amerikanern über 70 verschiedene Themen, von denen 50 regelmäßig angewandt wurden, umfasst.

- **CallHome Spanish** ist ein corpus der 120 spontan geführte Telefongespräche zwischen spanischen Verwandten enthält.

Tabelle 3: Performance des n-best-list-Ansatzes zur Wortfehlerminimierung

| | | WER [%] | SER [%] |
|---|---|---|---|
| Switchboard | Standardansatz | 52.7 | 84.0 |
| | n-best-Ansatz | 52.2 | 84.4 |
| CallHome Spanish | Standardansatz | 68.4 | 80.9 |
| | n-best-Ansatz | 67.8 | 81.2 |

Man erkennt, daß der n-best-Listen-Ansatz im Vergleich zum Standardansatz der Satzfehlerminimierung in Bezug auf die Wortfehler wie gewünscht besser abschneidet. Dabei erreicht er auf dem Switchboard-corpus eine absolute Verbesserung der Wortfehlerrate von 0,5% und auf dem CallHome Spanish-corpus eine absolute Verbesserung von 0,6%, jeweils verglichen mit der Wortfehlerrate nach Satzfehlerminimierung.

## 4.3 Wortfehlerminimierung über word-lattice-consensus-Ansatz

Beim in Abschnitt 4.2 vorgestellten Ansatz über n-best-Listen ist man dem Problem der enormen Rechenkomplexität des Entscheidungskriteriums (8) begegnet, indem der Hypothesenraum deutlich eingeschränkt wurde. Diese Einschränkung kann zur Folge haben, daß der in Bezug auf den Wortfehler optimale Satz bei der Vorauswahl über die Satz-posteriori-Wahrscheinlichkeiten der Hypothesen herausfällt und somit bei der eigentlichen Wortfehlerminimierung keine Rolle mehr spielt. Deshalb gab es Überlegungen wie man das Entscheidungskriterium (8) so modifizieren kann, daß der Hypothesenraum uneingeschränkt bleibt. Diese Überlegungen führten zu einer Modifizierung der Anpassungsart zwischen zwei Sätzen, der 1:1-Anpassung nach Positionen. Diese neue Anpassungsart wird im folgenden Kapitel vorgestellt und Voraussetzungen gezeigt, die für eine Durchführung der Anpassung notwendig sind. Darauf basierend wird ein neues Entscheidungskriterium hergeleitet und gezeigt wie man den Hypothesenraum transformiert um die Voraussetzungen für die Anwendung des neuen Kriteriums der Wortfehlerminimierung über den word-lattice-consensus-Ansatz herzustellen. Schließlich wir die Performance des Ansatzes an verschiedenen corpora getestet.

### 4.3.1 1:1-Anpassung nach Wortposition

Der Rechenaufwand der Levensthein-Anpassung entsteht, da bei der Anpassung zwischen zwei Sätzen alle möglichen Anpassungen durchprobiert werden und die mit den minimalen Kosten durchgeführt wird. Wenn man die Einschränkung macht, daß bei der Anpassung keine deletions / insertions auftreten dürfen, sondern nur substitutions dann gibt es nur noch eine mögliche Anpassungsmöglichkeit. Diese ordnet jedem Wort $w_i$ eines Satzes $w_1^N$ das entsprechende Wort $v_i$ eines Satzes $v_1^N$ zu. Wie man an den einzelnen Indices erkennen kann müssen hierbei die Sätze gleicher Länge sein. Dieser Zustand kann leicht

hergestellt werden, indem kürzere Hypothesen mit dem leeren Wort aufgefüllt werden. Wie dieser Vorgang genau funktioniert werden wir später im Abschnitt der lattice-Transformation kennenlernen. Die bei dieser Anpassungsart entstehenden Kosten betragen

$$\mathcal{L}'(w_1^N, v_1^N) = \sum_{i=1}^{N} (1 - \delta(w_i, v_i)) \tag{11}$$

Hierbei kommt wiederum die Kroneckerfunktion $\delta$ zum Einsatz, die an jeder Position feststellt ob die Worte der beiden verglichenen Sätze voneinander abweichen oder nicht.
Konkret ist die Idee, um den Zustand der Längengleichheit aller Hypothesen herzustellen, daß man alle Hypothesen des Ausgangslattices in einem lattice, welches multiple alignment genannet wird, vereinigt. In diesem müssen alle Knoten genau einmal durchlaufen werden, was bedeutet, daß alle Sätze dieselbe Länge haben, da alle Pfade durchs lattice gleich lang sind. Bei der Anpassung zweier Sätze interessiert dann ob beide Sätze zwischen zwei Knoten dieselbe Kante durchlaufen (identisch) oder unterschiedliche (abweichend). Ein Beispiel für ein multiple alignment sieht wie folgt aus:

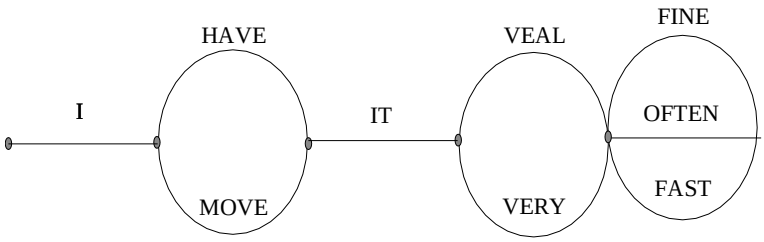


Abbildung 2: lattice in gewünschter Gestalt (multiple alignment)

### 4.3.2 Herleitung des Entscheidungskriteriums

Mit Hilfe von $L(w_1^N, v_1^M) = \mathcal{L}'(w_1^N, v_1^N)$ und dem Bayesschen Entscheidungskriterium (5), wobei $M = N = const.$ kann jetzt ein alternatives Entscheidungs-

kriterium zur Wortfehlerminimierung hergeleitet werden:

$$
\begin{aligned}
\{w_1^N\}_{opt} &= \underset{w_1^N}{\operatorname{argmin}}\{\sum_{v_1^N} p(v_1^N|x_1^T)\cdot\mathcal{L}'(v_1^N,w_1^N)\} \\
&= \underset{w_1^N}{\operatorname{argmin}}\{\sum_{v_1^N} p(v_1^N|x_1^T)\cdot\sum_{i=1}^{N}(1-\delta(w_i,v_i)) \\
&= \underset{w_1^N}{\operatorname{argmin}}\{\sum_{v_1^N} p(v_1^N|x_1^T)\cdot\sum_{i=1}^{N}1-\sum_{v_1^N} p(v_1^N|x_1^T)\cdot\sum_{i=1}^{N}\delta(w_i,v_i) \\
&= \underset{w_1^N}{\operatorname{argmin}}\{N-\sum_{n=1}^{N} p(w_n|x_1^T)\} \\
&= \underset{w_1^N}{\operatorname{argmax}}\{\sum_{n=1}^{N} p(w_n|x_1^T)\} \qquad (12)
\end{aligned}
$$

Umgangssprachlich formuliert bedeutet dies, daß derjenige Satz, dessen Worte an ihrer Position im Satz jeweils die maximale Wort-posteriori-Wahrscheinlichkeit besitzen der mit dem minimal zu erwartenden Wortfehler ist. Um das gerade hergeleitete Kriterium allerdings anwenden zu können muss das Ausgangslattice zuerst in das multiple alignment transformiert werden. Dies wird im nächsten Abschnitt beschrieben.

### 4.3.3 Latticetransformation

In diesem Abschnitt wird beschrieben wie die für Anwendung des Kriteriums (12) beschriebenen Voraussetzungen geschaffen werden. Dies geschieht durch eine Ausgangslatticetransformation, die zum multiple alignment führt.

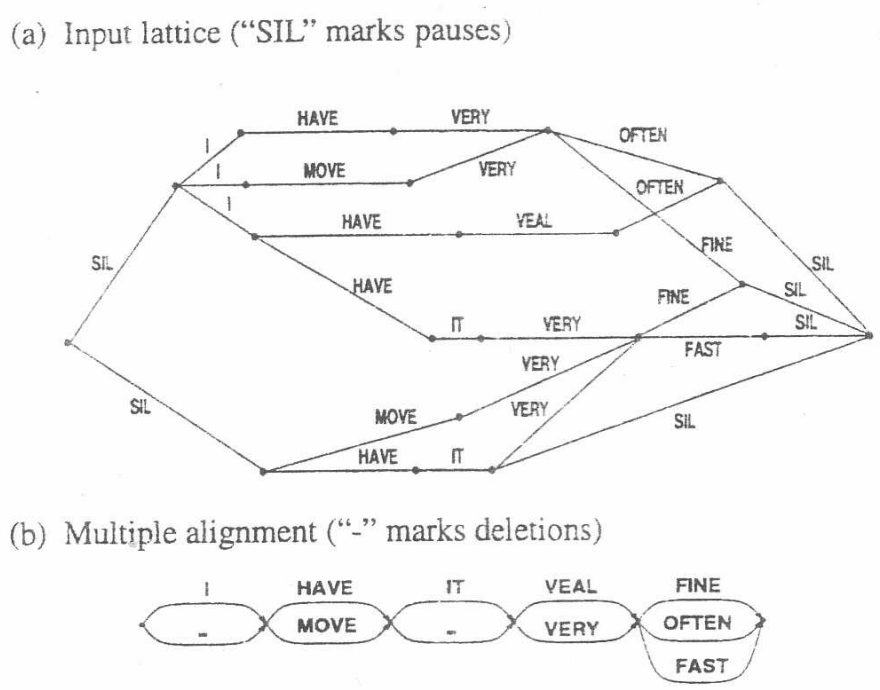


Abbildung 3: Ziel der Latticetransformation [2]

Um die in der Folge besprochenen Umformungen besser erläutern zu können wird das Ausgangslattice zuerst formalisiert (vgl. [2]).
Das lattice besteht aus einer Menge von Kanten E, wobei jeder Kante ein Wort Word(e), ein Startknoten Inode(e) mit Startzeitpunkt Itime(e), ein Endknoten Fnode(e) mit Endzeitpunkt Ftime(e) und ein Gewicht $p([e])$ zugeordnet ist. Sei $F \subset E$. Dann sei $Words(F) = \{f | \exists e \in F : Word(e) = f\}$ und $P(F) = \sum_{e \in F} P(e)$.
Ziel ist es durch Umformungen des lattice ein neues lattice zu erhalten, bei dem jeder Knoten genau einmal durchlaufen wird, d.h. zwischen dem ersten und zweiten Knoten verlaufen die Kanten der möglichen Worte für Position 1 im Satz, zwischen Knoten 2 und 3 die Kandidaten für Position 2 etc. Ziel der Transformation ist offensichtlich, die Kanten zu Äquivalenzrelationen $[e_i]$ zu vereinigen, wobei einer Äquivalenzrelation $[e_i]$ die Kanten, deren Worte Kandidaten für Position $i$ im Satz darstellen angehören. Dabei sind einige Einschränkungen zu beachten, damit die Hypothesen um Ausgangslattice bei der Latticetransformation nicht zerstört werden. Die Kanten des Ausgangslattice stehen in zeitlicher $\leq$-Relation zueinander, d.h. für ein $e, f \in E$ gilt $e \leq f$ gdw.

$$e = f \quad \vee \quad Fnode(e) = Inode(f) \quad \vee \quad \exists e' \in E \quad : \quad e \leq e' \quad \wedge \quad e' \leq f$$

Diese Relationen sind zu wahren damit die Satzhypothesen des Ausgangslattice auch nach der Transformation Bestand haben. Es muss also nach der Transfor-

mation gelten:

$$e \leq f \quad \wedge \quad e \in [e_i] \quad \wedge \quad f \in [e_j] \quad \Rightarrow \quad i \leq j \tag{13}$$

Anzumerken hierbei ist, daß Anfangs- bzw. Endknoten von vereinigten Kanten miteinander verschmelzen, während der Transformation des lattice entstehen somit auch neue, zusätzliche Hypothesen und somit auch neue, zusätzliche $\leq$-Relationen. Ausserdem ist beim Vereinigen der Kanten ein weiterer Gedanke zu beachten. Die akustischen Beobachtungen, die den einzelnen Positionen im Satz zugrundeliegen sind für alle Hypothesen, abgesehen von leichten Verschiebungen durch variierende Wortgrenzen, annähernd identisch. Somit ist es offensichtlich sinnvoll primär solche Kanten zu vereinigen, die dasselbe Wort repräsentieren (intra-word-clustering) und danach Kanten, die phonetisch ähnliche Worte repräsentieren (inter-word-clustering).
Beim intra-word-clustering werden iterativ diejenigen Kantenmengen vereinigt, die in Bezug auf ein Ähnlichkeitsmaß den maximalen Wert erzielen. Dieses Ähnlichkeitsmaß ist wie folgt definiert [2]:

$$SIM(E_1, E_2) = \max_{e_1 \in E_1, e_2 \in E_2} \quad overlap(e_1, e_2) \cdot p(e_1) \cdot p(e_2) \tag{14}$$

Die overlap-Funktion gibt hierbei das Ausmaß der zeitlichen Überlappung der Kanten an, normalisiert durch die Länge der Kantenlängen. Erreicht wird damit, daß primär Kanten vereinigt werden die sich stark überlappen, da dies neben phonetischer Gleichheit ein weiterer Hinweis darauf ist, daß sie sehr wahrscheinlich Kandidaten für dieselbe Position im Satz darstellen. Dieser iterative Schritt erfolgt solange, bis kein intra-word-clustering unter den Einschränkungen (13) mehr möglich ist. Vereinigte Kanten werden während des Prozesses des intra-word-clustering weiterhin als separate Kanten betrachtet. Nach Abschluss des intra-word-clustering werden die einzelnen Kantenmengen im lattice jeweils durch eine einzelne Kante repräsentiert. Deren Wahrscheinlichkeit ist die Summe der Einzelwahrscheinlichkeiten der in ihr vereinigten Kanten.
Auf das intra-word-clustering folgt das inter-word-clustering. Dabei geht es bei der Ähnlichkeit der Kantenmengen nicht mehr um das zeitliche Überlappen der Kanten sondern um phonetische Ähnlichkeit der jeweils repräsentierten Worte. Dieses Ähnlichkeitsmaß sieht wie folgt aus [2]:

$$\begin{aligned} SIM(F_1, F_2) = \operatorname*{avg}_{w_1 \in Words(F_1), w_2 \in Words(F_2)} \quad & sim(w_1, w_2) \cdot \\ & p(\{e \in F_1 : Word(e) = w_1\}) \cdot \\ & p(\{e \in F_2 : Word(e) = w_2\}) \end{aligned} \tag{15}$$

$sim(w_1, w_2)$ gibt die phonetische Ähnlichkeit zwischen zwei Worten an. Diese wird beispielsweise über die die Formel $1 - editdistance(w_1, w_2)$ angegeben, wobei die editdistance vergleichbar ist mit der Anzahl der Wortfehler die die Levensthein-Anpassung beim Vergleich zweier Sätze ausgibt, allerdings auf phoneme-Ebene beim Vergleich zweier Worte. Hierbei sind im Gegensatz zum intra-word-clustering, wo die Kantenmengen dasselbe Wort repräsentieren

müssen grundsätzlich alle Kantenmengen die zueinander in keinem $\leq$-Verhältnis stehen, Kandidaten miteinander vereinigt zu werden. Das intra-word-clustering wird wiederum iterativ solange vollzogen bis keine Vereinigung unter Wahrung der Einschränkungen (13) mehr möglich ist. Die vereinigten Kanten werden beim inter-word-clustering sowohl während wie auch nach Abschluss des clusterings weiterhin als separate Kanten mit separaten Wahrscheinlichkeiten betrachtet.
Wenn durch eine bestehende $\leq$-Relation, die aus einer Hypothese die sehr unwahrscheinlich ist resultiert, eine Vereinigung von Kantenmengen blockiert wird, die allerdings in Hinblick auf die anderen Hypothesen sinnvoll erscheint ist dies ungewollt. Hierbei schafft das sogenannte **pruning** Abhilfe, welches erlaubt blockierende Relationen, wenn sie aus einer unwahrscheinlichen Hypothese resultieren, einfach zu ignorieren und somit die Blockierung des clusterings aufzuheben. Dieser pruning-Schritt wird vor den beiden clustering-Schritten durchgeführt um einen reibungslosen Ablauf des clusterings zu ermöglichen.

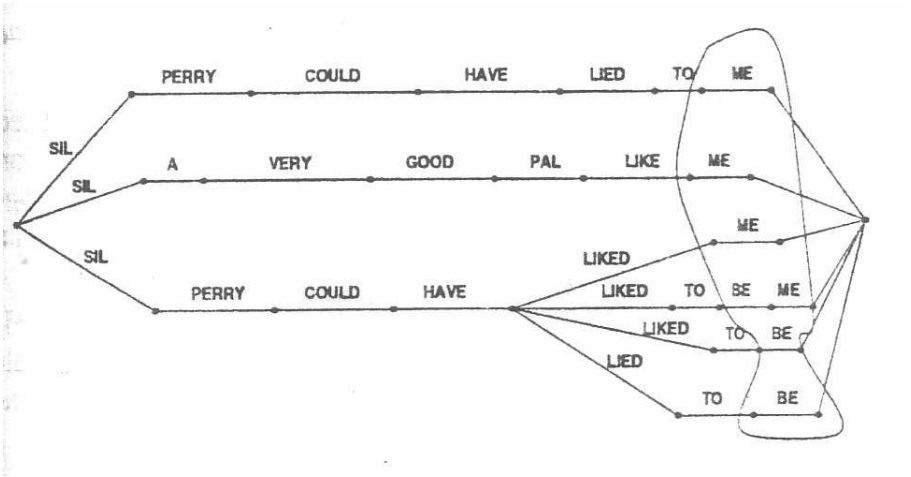


Abbildung 4: Beispiel für die Notwendigkeit des prunings [2]

Nach Durchführung all dieser Schritte liegt ein word lattice vor, was dem multiple alignment entspricht. D.h. zwischen zwei Knoten, wobei jeder Knoten genau einmal durchlaufen werden muss, liegen die Kantenalternativen für die jeweilige Position im Satz, gekoppelt mit der entsprechenden Wahrscheinlichkeit, daß diese Kante die treffende ist. Die Wahrscheinlichkeiten der jeweiligen Wortalternativen für eine Position müssen sich hierbei offensichtlich zu 1 addieren, da logischerweise irgendein Wort an der Position stehen muss. Oft ist dies allerdings nicht der Fall, da beim clustering die Kanten einer Hypothese des Ausgangslattices durchaus so zu Äquivalenzrelationen $[e_i]$ zugeteilt werden können daß zwei Kanten, die ursprünglich verbunden waren jetzt Äquivalenzrelationen $[e_i]$ angehören, die nicht direkt aufeinander folgen. Dieser Hypothese "fehlt" offensichtlich ein Wort. Gelöst wird das Problem indem man derarti-

ge Lücken in jeder Menge $[e_i]$ durch eine Kante ergänzt, die das leere Wort repräsentiert; die neue Kante erhält die Wahrscheinlichkeit $1 - p([e_i])$ [2].

### 4.3.4 Findung der consensus hypothese

Wenn das multiple alignment vorliegt so ist es ein einfacherer Schritt daraus die consensus hypothese (optimale Wortfolge) abzulesen. In Erinnerung an das Entscheidungskriterium (12) suchen wir den Pfad durchs lattice, dessen Kantenwahrscheinlichkeiten sich zur maximalen Wahrscheinlichkeit aufaddieren. Da alle Knoten nach Definition genau einmal durchlaufen werden genügt es zwischen zwei Knoten die jeweils wahrscheinlichste Verbindung zu wählen [2]. Hierbei ist zu beachten, daß durchaus ein Satz entstehen kann, der im Ausgangslattice nicht vorhanden war, jedoch durch die Transformation und dadurch bedingte neue Kantenverbindungen entstanden ist. Werden solche Sätze als Lösung zugelassen spricht man von lattice-consensus, anderenfalls von n-best-consensus [2].

### 4.3.5 Performance

Die Ergebnisse in der folgenden Tabelle entstammen einer Versuchsreihe von L. Mangu, E. Brill und A. Stolcke [2] bzw. Versuchsergebnissen aus Lidia Mangus Dissertation [3]. Die dabei verwendeten corpora, Switchboard und Broadcast News sind wie folgt aufgebaut:

- **Switchboard** ist ein corpus der 2400 Telefongespräche zwischen 543 Amerikanern über 70 verschiedene Themen, von denen 50 regelmäßig angewandt wurden, umfasst.
- **BroadcastNews** ist ein corpus der Aufzeichnungen von Radio- bzw. Fernsehsendungen von ABC, CNN, NPR, PBS im Zeitraum von Januar 1992 bis Juni 1996 enthält (Vokabulargröße 65491)

Tabelle 4: Performance des word-lattice-consensus-Ansatzes zur Wortfehlerminimierung

| | Switchboard WER [%] | BroadcastNews WER [%] | WS97 dev-test WER [%] |
|---|---|---|---|
| Standardansatz | 38.5 | 33.1 | 38.5 |
| n-best-Ansatz | 37.9 | 33.0 | 37.9 |
| lattice-consensus | 37.3 | 32.5 | 37.1 |
| n-best-consensus | N/A | N/A | 37.4 |

Man erkennt, daß der word-lattice-consensus-Ansatz den Wortfehler stärker minimiert als der n-best-Ansatz. Nimmt man als Ergebnis des lattice-Ansatzes den lattice-consensus, so verbessert das Verfahren die Wortfehlerrate im Verleich zum Standardansatz (Satzfehlerminimierung) absolut gesehen zwischen

0,6% und 1,4% (n-best-Ansatz zwischen 0,1% und 0,6%). In [2] wird ausserdem einen Vergleich zwischen der Performance von lattice-consensus und n-best-consensus dar. Beide Verfahren verbessern die WER im Gegensatz zum n-best-Ansatz (-0,5% bzw. -0,8%) was man den zum word-lattice-consensus-Ansatz führenden Vorüberlegungen zuschreiben kann [2]. Dabei ergibt sich allerdings auch, daß der lattice-consensus im Gegensatz zum n-best-consensus auf WS97 dev-test um 0,3% absolut gesehen besser abschneidet. Diesen Unterschied kann man dem beim lattice-consensus durch bei der Latticetransformation neu entstandene Relationen vergrößerten Hypothesenraum zuschreiben [2].

## 4.4 Wortfehlerminimierung über den "time frame error"-Ansatz

Wie auch beim word-lattice-consensus-Ansatz wird beim "time frame error"-Ansatz eine alternative Berechnung des Wortfehlers verfolgt, indem die Anpassungsart zwischen zwei Sätzen weiter variiert wird. Der word-lattice-consensus-Ansatz hat als Voraussetzung die identische Länge der Hypothesen, welche in der Regel erst hergestellt werden muss. Um diesem Problem aus dem Wege zu gehen wurde die Idee des "time frame error" entwickelt, eine 1:1-Anpassung nach Zeitpunkten [4]. Dies hat den Vorteil, daß im Gegensatz zum Begriff der Position im Satz, welche einen kontextabhängigen Begriff darstellt, ein Zeitpunkt eine absolute Größe ist und somit flexibler handhabbar ist.
Mit Hilfe der neu einzuführenden 1:1-Anpassung nach Zeitpunkten wird die Kostenfunktion des "time frame error" eingeführt und darauf basierend das Entscheidungskriterium zur Wortfehlerminimierung über den "time frame error" hergeleitet. Abschliessend wird dieses Kriterium anhand von Tests an unterschiedlichen corpora auf die Performance hin überprüft.

### 4.4.1 1:1-Anpassung nach Zeitpunkten

Bei der 1:1-Anpassung zweier Sätze nach Zeitpunkten wird zu jedem Zeitpunkt (time frame) überprüft ob die beiden vorliegenden Worte identisch oder unterschiedlich sind. Offensichtlich ist es bei dieser Anpassungsart wichtig für die einzelnen Sätze konkret die Wortübergänge zu kennen. Deshalb wird ein Satz nicht mehr einfach als Wortabfolge $w_1^N$ betrachtet, sondern daran gekoppelt sind die Wortgrenzen $t_1^N$, die jeweils den Endzeitpunkt des Wortes $w_i$ angeben. Ein Satz der Länge $N$ wird in der Folge somit als $[w;t]_1^N$ dargestellt. Die Anzahl der Zeitpunkte, zu denen die beiden Sätze voneinander abweichen wird als "time frame error" festgehalten.

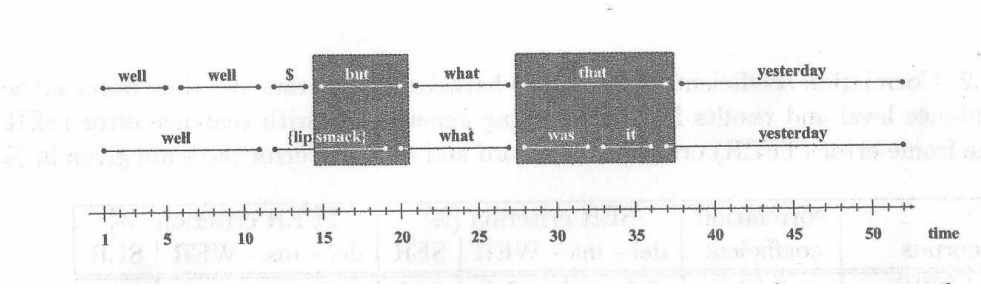


Abbildung 5: Idee des 'time frame errors' [4]

Mathematisch ausformuliert bedeutet dies, wenn wir den "time frame error" zwischen zwei Sätzen $w_1^N$ und $v_1^M$ als $\mathcal{C}([w;t]_1^N, [v;t]_1^M)$ bezeichnen [4]:

$$\mathcal{C}([w;t]_1^N, [v;\tau]_1^M) = \sum_{n=1}^{N} \frac{\sum_{\hat{t}=t_{n-1}+1}^{t_n} \left[1 - \sum_{\substack{m=1: \\ \tau_{m-1} \leq \hat{t} \leq \tau_m}}^{M} \delta(w_n, v_m)\right]}{1 + \lambda \cdot (t_n - t_{n-1} - 1)} \tag{16}$$

Hierbei wird nicht nur die Anzahl der "time frame error" gezählt, sondern dieser wird durch die Anzahl der betrachtetetn Zeitpunkte normalisiert. Dadurch erreicht man eine objektivere Bewertung der Abweichung da bei längeren Aussagen sinnvollerweise mehr Abweichungen erlaubt sind. Anhand der Grafik kann man auch eine Schwäche des "time frame error" erkennen. Ein langgezogenes "well" und ein doppeltes "well, well" werden nicht unterschieden, da lediglich die reine Gleichheit zwischen den Sätzen zu einem Zeitpunkt betrachtet wird, nicht allerdings die Gesamtentwicklung des Satzes in Hinblick auf Wortübergänge.

### 4.4.2 Herleitung des Entscheidungskriteriums

Basierend auf der neuen Kostenfunktion des "time frame error" kann nun mit Hilfe der auf Anwendung für Sätze mit Wortgrenzen modifizierten Formel (5) und $L([w;t]_1^N, [v;\tau]_1^M) = \mathcal{C}([w;t]_1^N, [v;\tau]_1^M)$ das Bayessche Entscheidungskriterium zur Wortfehlerminimierung über den "time frame error" hergeleitet werden

[4].

$$
\begin{aligned}
\{w_1^N\}_{opt} &= \underset{[w;t]_1^N}{\operatorname{argmin}}\{ \sum_{[v;\tau]_1^M} \mathcal{C}([w;t]_1^N,[v;\tau]_1^M)\cdot p([v;\tau]_1^M|x_1^T)\} \\
&\qquad \sum_{\hat{t}=t_{n-1}+1}^{t_n}[1- \sum_{\substack{m=1:\\ \tau_{m-1}\le\hat{t}\le\tau_m}}^{M} \delta(w_n,v_m)] \\
&= \underset{[w;t]_1^N}{\operatorname{argmin}}\{ \sum_{[v;\tau]_1^M}\sum_{n=1}^{N} \frac{}{1+\lambda\cdot(t_n-t_{n-1}-1)}\cdot p([v;\tau]_1^M|x_1^T)\} \\
&\qquad t_n-t_{n-1}-\sum_{\hat{t}=t_{n-1}+1}^{t_n} \sum_{\substack{m=1:\\ \tau_{m-1}\le\hat{t}\le\tau_m}}^{M} \delta(w_n,v_m)] \\
&= \underset{[w;t]_1^N}{\operatorname{argmin}}\{ \sum_{[v;\tau]_1^M}\sum_{n=1}^{N} \frac{}{1+\lambda\cdot(t_n-t_{n-1}-1)}\cdot p([v;\tau]_1^M|x_1^T)\} \\
&\qquad t_n-t_{n-1}-\sum_{\hat{t}=t_{n-1}+1}^{t_n}\sum_{[v;\tau]_1^M} \sum_{\substack{m=1:\\ \tau_{m-1}\le\hat{t}\le\tau_m}}^{M} \delta(w_n,v_m)\cdot p([v;\tau]_1^M|x_1^T) \\
&= \underset{[w;t]_1^N}{\operatorname{argmin}}\{\sum_{n=1}^{N} \frac{}{1+\lambda\cdot(t_n-t_{n-1}-1)}\} \\
&\qquad \sum_{\hat{t}=t_{n-1}+1}^{t_n}[1-\sum_{[v;\tau]_1^M} \sum_{\substack{m=1:\\ \tau_{m-1}\le\hat{t}\le\tau_m}}^{M} \delta(w_n,v_m)\cdot p([v;\tau]_1^M|x_1^T)] \\
&= \underset{[w;t]_1^N}{\operatorname{argmin}}\{\sum_{n=1}^{N} \frac{}{1+\lambda\cdot(t_n-t_{n-1}-1)}\} \\
&= \underset{[w;t]_1^N}{\operatorname{argmin}}\{\sum_{n=1}^{N} \frac{\sum_{\hat{t}=t_{n-1}+1}^{t_n}[1-p_{\hat{t}}(w_n|x_1^T)]}{1+\lambda\cdot(t_n-t_{n-1}-1)}\} \qquad (17) \\
&=: \underset{[w;t]_1^N}{\operatorname{argmin}}\{\sum_{n=1}^{N} \mathcal{S}[w_n;t_{n-1}+1,t_n]\} \qquad (18)
\end{aligned}
$$

Umgangssprachlich formuliert gibt die Größe $\mathcal{S}[w_n;t_{n-1}+1,t_n]$ an, wie groß die durch die Länge des betrachteten Wortes normalisierte Wahrscheinlichkeit ist, daß im Zeitintervall $[t_{n-1}+1,t_n]$ das Wort $w_n$ nicht vorliegt. Dieser Wert kann für alle Worthypothesen $[w_n;t_{n-1}+1,t_n]$ unabhängig vom eigentlichen Auswertungsprozess offline berechnet werden, die eigentliche Findung der treffendsten Hypothese kann somit recht schnell durchgeführt werden [4]. Berechnet wird der $\mathcal{S}$-Wert jeweils auf Basis der Wahrscheinlichkeit $p_{\hat{t}}(w_n)$, die sich durch Aufsummieren der Satz-posteriori-Wahrscheinlichkeiten der Hypothesen ergibt, bei denen zum Zeitpunkt $\hat{t}$ das Wort $w_n$ vorliegt. Der optimale Satz ist aus den Worthypothesen $[w_n;t_{n-1}+1,t_n]$ zusammengesetzt, die zeitlich zueinander passen (d.h. in Bezug auf die Wortgrenzen aneinander anschliessen) und in Bezug

auf die Summe der $\mathcal{S}$-Werte ein Minimum ergeben.

### 4.4.3 Performance

Die Ergebnisse der folgenden Tabelle entstammen zum größten Teil aus Versuchen, die Frank Wessel in seiner Dissertation durchgeführt hat [4]. Die verwendeten corpora sind wie folgt aufgebaut:

- **ARISE** ist ein corpus der Mensch-Maschine-Gespräche, die über Telefon aufgenommen wurden enthält (Vokabulargröße 985)
- **Verbmobil** ist ein deutscher corpus, der verschiedene Anzahlen zwischenmenschlicher Gespräche enthält (Vokabulargröße 7128)
- **NAB 20k / NAB 64k** sind corpora, die Aufnahmen hoher Qualität von vorgelesenen Zeitungsartikel enthalten (Vokabulargröße 20.000 bzw. 64.000 Worte)
- **BroadcastNews** ist ein corpus der Aufzeichnungen von Radio- bzw. Fernsehsendungen von ABC, CNN, NPR, PBS im Zeitraum von Januar 1992 bis Juni 1996 enthält (Vokabulargröße 65491)

Tabelle 5: Performance des "time frame error"-Ansatzes zur Wortfehlerminimierung

| | | Satzfehler | | time frame error | |
|---|---|---|---|---|---|
| corpus | Vokabulargröße | WER [%] | SER [%] | WER [%] | SER [%] |
| ARISE | 985 | 15.8 | 24.3 | 15.0 | 23.6 |
| Verbmobil | 7128 | 33.6 | 70.2 | 32.3 | 71.2 |
| NAB 20k | 19987 | 13.2 | 79.4 | 12.9 | 79.7 |
| NAB 64k | 64736 | 11.1 | 74.8 | 10.7 | 75.8 |
| BN | 65491 | 33.3 | 91.4 | 32.3 | 91.6 |

Festzuhalten hierbei ist, daß der "time frame error"-Ansatz den Wortfehler verglichen mit dem Wortfehler bei durchgeführter Satzfehlerminimierung bei allen corpora verringert (relativ gesehen um 2,3% bis 5,1%), am stärksten ist die Verbesserung bei corpora mit Spontansprache.

# 5 Zusammenfassung / Vergleich der vorgestellten Verfahren

In der Ausarbeitung haben wir gesehen, daß eine Wortfehlerminimierung explizit durch ein entsprechendes Minimierungskriterium geschehen muss, da die gängige Form der Satzfehlerminimierung andere Ergebnisse liefert. Zu diesem Zwecke wurden drei verschiedene Verfahren vorgestellt. Allgemein kann man sagen, daß das primäre Ziel der Wortfehlerminimierung von allen drei vorgestellten Verfahren erzielt wird, was man an den jeweiligen Testläufen auf verschiedenen corpora feststellen kann. Der Unterschied zwischen den Verfahren

liegt in dem Kompromiss der eingegangen wird um die Wortfehlerminimierung explizit betreiben zu können. Der n-best-list-Ansatz schränkt den Hypothesenraum ein um die Komplexität des Kriteriums einzuschränken. Im Gegensatz dazu nähern die beiden anderen Verfahren die Levensthein-Anpassung durch 1:1-Anpassung nach Position (word-lattice-consensus-Ansatz) bzw. Zeitpunkt ("time frame error"-Ansatz) an wodurch eine einfachere Berechnung des Wortfehlers ermöglicht wird als sie die Levensthein-Anpassung bietet. Dieser Aspekt führt dazu, daß die jeweiligen Entscheidungskriterien, die auf einer Annäherung des Wortfehlers basieren, analytisch vereinfacht werden können und somit rechnerisch leicht durchführbar sind. Die gemachten Annäherungen weichen jeweils in der Praxis, wie verschiedene Versuche gezeigt haben, absolut um 0.15% von dem Wortfehler nach Levensthein ab. Diese Abweichung wird eingegangen, da sie im Vergleich zum erreichten Berechnungskomfort verschwindend gering ist. Die Unterschiede zwischen den einzelnen Verfahren werden noch einmal tabellarisch festgehalten:

Tabelle 6: Vergleich der Wortfehlerminimierungskriterien

| | Wortfehlerminimierungkriterium | | |
|---|---|---|---|
| | n-best-list | word-lattice-consensus | "time frame error" |
| Anpassungsart/ Fehlerart | Levensthein | 1:1 nach Wortposition | 1:1 nach Zeitpunkt |
| Nachteile | begrenzter Hypothesenraum | lediglich angenähert | lediglich angenähert |

# 6 Ausblick

Bei den verschiedenen Verfahren wird deutlich, daß durch die Überlegungen zur Wortfehlerminimierung mit Hilfe der Kriterien zur Wortfehlerminimierung der Wortfehler bedeutend gesenkt wird. Dennoch liegt die Wortfehlerrate in hohen Bereichen, was es in Zukunft weiter zu verbessern gilt. Die vorgestellten Verfahren basieren auf der Verwendung von Satz-posteriori-Wahrscheinlichkeiten die in herkömmlicher Weise durch Zerlegung in Sprach- und akustischem Modell berechnet werden. Diese Werte sind nicht optimal, an komplexeren und somit genaueren Berechnungen von Satz-posteriori-Wahrscheinlichkeiten wird gearbeitet, die die Performance der Ansätze zur Wortfehlerminimierung durchaus weiter verbessern könnten.

## Literatur

[1] M. Weintraub A. Stolcke, Y. Knig. Explicit word error rate minimization in n-best list rescoring. In *Proc. Europ. Conf. on Speech Communication and Technology*, volume 2, pages 163–166, September 1997.

[2] A. Stolcke L. Mangu, E. Brill. Finding consensus among words: Lattice-based word error minimization. In *Proc. Europ. Conf. on Speech Communication and Technology,*, volume 1, pages 495–498, September 1999.

[3] Lidia Mangu. Finding consensus in speech recognition. Technical report, Baltimore, Maryland, April 2000.

[4] Frank Wessel. Word posterior probabilities for large vocabulary continuous speech recognition. Technical report, RWTH Aachen, Januar 2002.