Statistische Maschinelle Übersetzung

Teil III - Sprachmodellierung

(engl. Language Modelling)

Thomas Schoenemann
Heinrich-Heine-Universität Düsseldorf

Sommersemester 2012

Anwendungen und Aufgaben

Erinnerung: Übersetzungsproblem:

$$\text{finde} \qquad \arg\max_{I, e_1^I} \; p(f_1^J | e_1^I) \cdot p(e_1^I, I)$$

$p(e_1^I, I)$ heißt **Sprachmodell**.

Aufgaben:

- Entscheidung über Wortordnung:

  $$p(\text{the house is small}) > p(\text{small the is house})$$

- Hilfe bei Disambiguierung:

  $$p(\text{I am going home}) > p(\text{I am going house})$$

n-gram Modelle

Mit Hilfe der Kettenregel (vgl. Einführung):

$$p(e_1^I) = \prod_{i=1}^{I} p(e_i \,|\, e_1, \ldots, e_{i-1})$$

Für n-gram-Modelle : $p(e_i \,|\, e_1, \ldots, e_{i-1}) = p(e_i | e_{i-(n-1)}, \ldots e_{i-1})$

Nur eine Approximation (nicht gerechtfertigte Unabhängigkeitsannahme), bedenke z.B.

– `macht ... auf`

– $p(\cdot|$`swallowed the large green`$)$ vs. $p(\cdot|$`the large green`$)$

Im Folgenden häufig abgekürzt:

$$p(e_i | e_{i-(n-1)}, \ldots e_{i-1}) = p(e_i | e_{i-(n-1)}^{i-1})$$

Die Wortfolge, auf die konditioniert wird, heißt Historie.

Satzanfang und -ende

Satzanfang und Satzende werden extra markiert. Vorteil: keine speziellen Wahrscheinlichkeiten für Anfang und Ende, alle Basiswahrscheinlichkeiten haben die gleiche Form.

Z.B. für $n = 3$:

`I am going home`

$\Downarrow$

`<s> <s> I am going home </s>`

Jeweils $n-1$ extra Anfangsmarker für ein n-gram Modell. Nur ein Endmarker (an Position $I+1$)

Positionen für Anfangsmarker: $0, -1, \ldots$

Beachte: durch $p(\texttt{</s>} \,|\, \texttt{going home})$ wird implizit die Satzlänge modelliert $\Rightarrow p(e_1^I, I) = \prod_{i=1}^{I+1} p(e_i \,|\, e_1, \ldots, e_{i-1})$.

Gängige n-gram Modelle

$$p(e_1^I, I) = \prod_{i=1}^{I+1} p(e_i \mid e_{i-(n-1)}^{i-1})$$

Beispiele:

- Unigram: $p(e_i)$
- Bigram: $p(e_i|e_{i-1})$
- Trigram: $p(e_i|e_{i-2}, e_{i-1})$

Terminologie: bis $n = 3$ lateinische Zahlen, danach Viergram, Fünfgram, etc. ("gram" vom griechischen "gramma" für Wort)

n-gram Modelle in der Übersetzung

für maschinelle Übersetzung:
– mindestens Trigram ratsam
– häufig: klassenbasiertes Fünfgram

generelles Problem: extrem viele Ereignisse
z.B. für `engl. Europarl`, $|\mathcal{V}| = 86.700 \Rightarrow |\mathcal{V}|^2 =$7.5 Milliarden Bigramme
nur 1.3 Millionen im Text, aber viele andere möglich!

Insgesamt:
- Sprachmodellierung ist eigenes Forschungsgebiet
- Forschung zu Übersetzung: Sprachmodelle meist gegeben (Black Box via Standard Toolkits)
- trotzdem: manche Techniken (z.B. Glättung) auch für Übersetzung nützlich.

Training von Sprachmodellen

Sprachmodelle werden auf großen Textsammlungen (Corpora) trainiert. Beachte: *mono*linguales Problem, oft deutlich mehr Trainingsdaten als für die Übersetzung.

Gegeben: Trainingssätze $\mathbf{e}_s$ der Längen I_s, $s = 1, \ldots S$

Maximum Likelihood Kriterium, hier für ein Bigram:

$$\max_{\{p(e|e')\}} \prod_{s=1}^{S} \prod_{i=1}^{I_s+1} p(e_i|e_{i-1})$$

Im Folgenden:
Beispiel `Europarl Englisch`
Vokabelgröße $|\mathcal{V}| = 86.700$
insgesamt $\sum_s I_s = 29,6$ Millionen Wörter.

Maximum Likelihood Schätzung

Maximum Likelihood führt zu relativen Häufigkeiten (hier für ein Trigram):

$$p(\tilde{e}|\tilde{e}', \tilde{e}'') = \frac{N(\tilde{e}', \tilde{e}'', \tilde{e})}{\sum_{\tilde{e}'''} N(\tilde{e}', \tilde{e}'', \tilde{e}''')}$$

wobei $N(\tilde{e}', \tilde{e}'', \tilde{e}) = \sum_s \sum_{i=1}^{I_s+1} \delta(e_i^s, \tilde{e})\delta(e_{i-1}^s, \tilde{e}'')\delta(e_{i-2}^s, \tilde{e}')$ angibt, wie häufig das Trigram im Trainingscorpus vorkommt.

Probleme:

- wir möchten keine Wortfolge völlig ausschließen.
- Sequenz $\tilde{e}', \tilde{e}''$ evtl. nicht in den Trainingsdaten
  $\Rightarrow$ Backing off: $p(\tilde{e}|\tilde{e}', \tilde{e}'') = p(\tilde{e}|\tilde{e}'')$
- Sequenz $\tilde{e}', \tilde{e}'', \tilde{e}$ evtl. nicht in den Trainingsdaten
  $\Rightarrow$ Glättung (engl. smoothing)

Glättung

Das Problem der Glättung ist von **enormer** Bedeutung für die Sprachmodellierung. Hier detailliert für ein Bigram Modell.

Gegeben: Counts für Bigramme
Aufgabe: Wahrscheinlichkeiten für ungesehene Bigramme auf > 0 setzen.

$\Rightarrow$ da Wahrscheinlichkeiten zu 1 summieren: passe auch die Wk.s von gesehenen Bigrammen an (aber bleibe möglichst nahe am Maximum Likelihood Schätzer).

Zur Lösung (zunächst): teile Text in zwei Teile
(`Training` und `Validierung`).

Glättung: Fundamentales

Grundfrage: wenn wir ein (nicht näher bestimmtes) Bigram im Training r-mal gesehen haben, wie häufig werden wir es in einem Text gleicher Länge sehen?

- Dieser Erwartungswert ist eine deutlich bessere Basis für die Schätzung von Wahrscheinlichkeiten:

$$p(\tilde{e}|\tilde{e}') = \frac{\text{erw. Anzahl Erscheinungen von } \tilde{e}'\tilde{e} \text{ in frischen Daten}}{\sum_{\tilde{e}''} \text{ erw. Anzahl Erscheinungen von } \tilde{e}'\tilde{e}'' \text{ in frischen Daten}}$$

- Generell: wenn $r = 0$ wird es durchschnittlich häufiger auftreten, ansonsten seltener.
- Im wesentlichen: gucke nur auf den Count eines Bigrams, nicht auf das Bigram selbst.

Notation

- Vokabular $\mathcal{V}$: Menge aller in der Trainingssequenz vorkommenden Wörter/Tokens.
- Anzahl der n-gramme (für gegebenes n), die im Training r-mal vorkamen:

$$N_r = \left|\{(\tilde{e}_1, \ldots, \tilde{e}_n) \,|\, N(\tilde{e}_1, \ldots, \tilde{e}_n) = r\}\right|$$

- Bei Historie $\tilde{e}$: Anzahl ungesehener Folgewörter:

$$N_0(\tilde{e}) = \left|\{(\tilde{e}, \tilde{e}') \,|\, \tilde{e}' \in \mathcal{V}, N(\tilde{e}, \tilde{e}') = 0\}\right|$$

  analog für längere Historien

- Bei Historie $\tilde{e}$: Anzahl (unterschiedl.) gesehener Folgewörter:

$$N_{1+}(\tilde{e}) = \left|\{(\tilde{e}, \tilde{e}') \,|\, \tilde{e}' \in \mathcal{V}, N(\tilde{e}, \tilde{e}') > 0\}\right|$$

  analog für längere Historien

Statistik

(gesehen im Training vs. gesehen im Validierungsteil, `engl. Europarl`): Annahme: Training- und Validierungstext sind gleich lang.

`Count-training`	`Count-validation`
0	0.00016
1	0.46235
2	1.39946
3	2.34307
4	3.35202
5	4.35234
6	5.33762
8	7.15074
10	9.11927
20	18.95948

D.h. Bigramme, die im Training nicht gesehen wurden, sind im Validierungsteil durchschnittlich 0.00016 mal vorhanden.

Einfache Glättung: Add-α

Add-α Verfahren:

$$p(e|e') = \frac{N(e', e) + \alpha}{\sum_{\tilde{e}}[N(e', \tilde{e}) + \alpha]}$$

Zur Wahl von α: z.B. so, dass der Validation-count für ungesehene Ereignisse stimmt.

Resultat (α bestimmt als 0.00017):

`Count-training`	`Add-`α`-Count`	`Count-validation (wirklich)`
0	0.00016	0.00016
1	0.95725	0.46235
2	1.91433	1.39946
3	2.87141	2.34307
4	3.82850	3.35202
5	4.78558	4.35234

Die Schätzer für gesehene Bigramme sind recht weit von den realen Daten entfernt.

Deleted Estimation

Anstatt des ursprünglichen Counts $N(e_1, e_2)$ für das Bigram e_1, e_2, nehme einfach den entsprechenden Testcount.

Statistik für unterschiedliche Validierungsets:

`Count-training`	`Count-validation-1`	`Count-validation-2`
0	0.00012	0.00016
1	0.46900	0.46235
2	1.40322	1.39946
3	2.41381	2.34307
4	3.36860	3.35202
5	4.28820	4.35234

Die Statistiken für unterschiedliche Corpora sind recht ähnlich
$\Rightarrow$ ein recht verlässlicher Schätzer.
Aber: die Trainingsdaten müssen in zwei Teile geteilt werden.

Fortgeschrittene Glättungsverfahren

- Good-Turing
  - r-mal gesehen im Training $\Rightarrow$ erwartet $r^* = (r+1)\frac{N_{r+1}}{N_r}$-mal in neuen Daten (gleicher Länge)
  - für (z.B.) $r > 6$: besser unmodifiziert lassen ($r^* = r$)
  - somit $p(\tilde{e}|\tilde{e}') = \frac{[N(\tilde{e}',\tilde{e})]^*}{\sum_{\tilde{e}''}[N(\tilde{e}',\tilde{e}'')]^*}$
  - wird auch für Übersetzung verwendet
- Absolutes Discounting (Ney et al.)

$$p(\tilde{e}|\tilde{e}') = \begin{cases} \frac{N(\tilde{e}',\tilde{e})-\delta}{\sum_{\tilde{e}''} N(\tilde{e}',\tilde{e}'')} & \text{wenn } N(\tilde{e}',\tilde{e}) > 0 \\ \frac{N_{1+}(\tilde{e})\,\delta}{N_0(\tilde{e})\sum_{\tilde{e}''} N(\tilde{e}',\tilde{e}'')} & \text{sonst} \end{cases}$$

  mit $\delta = \frac{N_1}{N_1+2N_2}$
  - gängiste Methode (heutzutage)

Hinter diesen Formeln: solide Theorie.

Beachte: wir können nun *alle* verfügbaren Daten im Training verwenden.

Backing Off

Auch für Trigramme, Viergramme etc. : reserviere Masse für ungesehene Ereignisse.

Aber: Konditionierung ungesehener Ereignisse auf die gesamte Historie nicht sinnvoll.

Beispiel:

- Historie `Scottish beer`: nur selten gesehen
- Trigramme `Scottish beer drinkers` und `Scottish beer eaters` beide ungesehen.
- Bisherige Glättungsverfahren: $\Rightarrow$ beide gleichwahrscheinlich
- besser: Konditionierung nur auf die Historie `drinkers` (weniger Datenknappheit).

Backing Off Modelle (vereinfacht)

Für $n \geq 3$ (denkbar auch für $n = 2$):

$$p_n\left(e_i|e_{i-(n-1)}^{i-1}\right) =$$

$$\begin{cases} d(e_{i-(n-1)}^{i-1})\, \tilde{p}_n\left(e_i|e_{i-(n-1)}^{i-1}\right) & \text{wenn } N(e_{i-(n-1)}^{i}) > 0 \\ (1 - d(e_{i-(n-1)}^{i-1}))\, p_{n-1}\left(e_i|e_{i-(n-2)}^{i-1}\right) & \text{sonst} \end{cases}$$

Hierbei ist $\tilde{p}_n(e_i|e_{i-(n-1)}^{i-1})$ eine beliebige Verteilung, die Masse nur auf gesehene Ereignisse verteilt, z.B. :

– ML-Schätzer

– durch Glättung modifizierte Counts, renormalisiert auf gesehene Ereignisse (gibt gleichzeitig den Wert für $d(e_{i-(n-1)}^{i-1})$)

Beachte: auch die Backoff-Verteilung $p_{n-1}(e_i|d(e_{i-(n-2)}^{i-1}))$ kann ihrerseits wieder Backing off einsetzen ($\rightarrow$ Rekursion).

Backing Off Modelle (präzise)

Problem beim sonst-Fall (letzte Folie): die Verteilung $p_{n-1}(\cdot|e_{i-(n-2)}^{i-1})$ normiert sich zu 1 für *alle* Wörter in $\mathcal{V}$, wird aber nur für Wörter ausgewertet, die nach $e_{i-(n-1)}^{i-1}$ nicht vorkamen

Korrekter sonst-Fall: Renormalisierung

$$(1 - d(e_{i-(n-1)}^{i-1})) \frac{p_{n-1}\left(e_i|e_{i-(n-2)}^{i-1}\right)}{\sum_{\tilde{e}:N(\tilde{e},e_{i-(n-1)}^{i-1})=0} p_{n-1}\left(\tilde{e}|e_{i-(n-2)}^{i-1}\right)}$$

Backing Off mit Glättung (vereinfacht)

Z.B. für Turing-Good:

$$p_n\left(e_i|e_{i-(n-1)}^{i-1}\right) =$$

$$\begin{cases} \frac{[N(e_{i-(n-1)}^{i})]^*}{\sum_{\tilde{e}}[N(e_{i-(n-1)}^{i-1},\tilde{e})]^*} & \text{wenn } N(e_{i-(n-1)}^{i}) > 0 \\ \frac{N_0(e_{i-(n-1)}^{i-1})\,[0]^*}{\sum_{\tilde{e}}[N(e_{i-(n-1)}^{i-1},\tilde{e})]^*}\, p_{n-1}\left(e_i|e_{i-(n-2)}^{i-1}\right) & \text{sonst} \end{cases}$$

Anmerkung: Auch bei positiven, aber kleinen $N(e_{i-(n-1)}^{i})$ kann Backing Off sinnvoll sein. Die Formel muss dann angepasst werden.

Backing Off: Witten-Bell Glättung

Betrachte:

– `spite` und `constant` beide 993 mal im Text.

– `spite` 979 mal gefolgt von `of`.

– `constant` gefolgt von 415 verschiedenen Wörtern.

$\Rightarrow p(\cdot|\texttt{constant})$ sollte mehr Masse für ungesehene Ereignisse reservieren als $p(\cdot|\texttt{spite})$.

Erinnerung: für Anzahl der unterschiedlichen Folgewörter:

$$N_{1+}(\tilde{e}_1, \ldots, \tilde{e}_{n-1}) = \left|\{\tilde{e}_n : N(\tilde{e}_1, \ldots, \tilde{e}_n) > 0\}\right|$$

Witten-Bell Masse für ungesehene Ereignisse:

$$1 - d(e_{i-(n-1)}^{i-1}) = \frac{N_{1+}(e_{i-(n-1)}^{i-1})}{N_{1+}(e_{i-(n-1)}^{i-1}) + N(e_{i-(n-1)}^{i-1})}$$

Interpolation

Anstatt von $p_n(e_i|e_{i-(n-1)}^{i-1})$ meist lieber Interpolation:

$$p_n^I(e_i|e_{i-(n-1)}^{i-1}) = \lambda(e_{i-(n-1)}^{i-1})\, p_n(e_i|e_{i-(n-1)}^{i-1}) + (1-\lambda(e_{i-(n-1)}^{i-1}))\, p_{n-1}(e_i|e_{i-(n-2)}^{i-1})$$

Hier: auch $p_{n-1}(e_i|e_{i-(n-2)}^{i-1})$ kann interpoliert werden
($\rightarrow$ Rekursion)

Schätzung von $\lambda(e_{i-(n-1)}^{i-1})$: auch hier Witten-Bell Glättung
(tatsächlich ursprünglich für Interpolation vorgeschlagen, nicht für Backing Off)

Backing Off & Interpolation

Gesamtverfahren (Interpolation + Backing Off):

$$p_n\left(e_i|e_{i-(n-1)}^{i-1}\right) =$$

$$\begin{cases} d(e_{i-(n-1)}^{i-1}) \left[\lambda(e_{i-(n-1)}^{i-1})\, \tilde{p}_n\left(e_i|e_{i-(n-1)}^{i-1}\right)\right. & \text{wenn } N(e_{i-(n-1)}^{i}) > 0 \\ \quad \left. + (1-\lambda(e_{i-(n-1)}^{i-1}))\, p_{n-1}\left(e_i|e_{i-(n-2)}^{i-1}\right)\right] & \\ (1-d(e_{i-(n-1)}^{i-1}))\, p_{n-1}\left(e_i|e_{i-(n-2)}^{i-1}\right) & \text{sonst} \end{cases}$$

D.h. bei Backing Off keine Interpolation

(bzw. diese hebt sich wieder auf)

Reflektion

- Glättung:
  – wichtig, damit alle Ereignisse eine positive Wk. haben.
  – beachte: extrem viele ungesehene n-gramme
  – alle ungesehenen Wörter erhalten die gleiche Wk. (bei gegeb. Historie)
- Backing Off:
  – Verkürzung der Historie wenn n-gram ungesehen
  – Ziel: bei ungesehenen Ereignissen suche nächstliegendes gesehene
  – Jetzt: ungesehene Wörter erhalten unterschiedliche Wk.s
- Interpolation:
  – Kombination verschiedener n-gramme
  – Motivation: bei hohen n gibt es meist nur wenige n-gramme mit einer gegeb. Historie $\Rightarrow$ Statistik nicht sonderlich aussagekräftig.
  – bei ungesehenen Wörtern: meist Backing Off.

Anmerkungen

- Die Counts N_r beziehen sich jeweils auf die Gesamtheit der n-gramme im Trainingskorpus, d.h. die ersten $n-1$ Wörter müssen *nicht* der gegebenen Historie entsprechen.
- Literatur: Glättung meist für Verbundverteilung der n-gramme.
- Add-α und Absolutes Discouting wirken sehr ähnlich, sind aber sehr unterschiedlich (besonders was die praktische Eignung angeht).
  - Bei Add-α wird zu *jedem* Count etwas hinzuaddiert, auch wenn der Count ursprünglich 0 war.
  - Beim Absoluten Discounting wird den *gesehenen* n-grammen etwas vom Count abgezogen.

Software

vollwertige Toolkits (Training + Anfragen): u.a.

- SRILM (Stanford Research Institute)
- IRSTLM (Trento, Italy)
- CMU Cam Toolkit

ausschließlich Anfragen:

- KenLM (integriert in MOSES)