Statistische Maschinelle Übersetzung

Teil VI - Tuning

Thomas Schoenemann
Heinrich-Heine-Universität Düsseldorf

Sommersemester 2012

Überblick

In diesem Teil: Tuning der Gewichtungsparameter λ_i von log-linearen Modellen

Erinnerung: log-lineares Modell:

$$p(\mathbf{e}, \mathbf{a} \,|\, \mathbf{f}) \propto \exp\left(\sum_i \lambda_i h_i(\mathbf{e}, \mathbf{a}|\mathbf{f})\right)$$

e: übersetzter Satz, **a**: alles andere (z.B. Alignment, Segmentierung, Reorderingpermutationen etc.)

Erinnerung: Featurefunktionen

Gängige Featurefunktionen für phrasenbasierte Modelle:

- n-gram-LM: $h_{\mathtt{lm}}(\mathbf{e}, \cdot|\cdot) = \prod_{i=1}^{|\mathbf{e}|+1} p(e_i|e_{i-(n-1)}^{i-1})$
- Übersetzungswk.s: $h_{\mathtt{e|f}}(\mathbf{e}, \mathbf{a}|\mathbf{f})$ (hängt von $\mathbf{a}$ ab)
- invertierte Übersetzungswk.s: $h_{\mathtt{f|e}}(\mathbf{e}, \mathbf{a}|\mathbf{f})$
- Wortstrafterm: $h_{\mathtt{wp}}(\mathbf{e}, \cdot|\cdot) = |\mathbf{e}|$
- Phrasenstrafterm: $h_{\mathtt{pp}}(\mathbf{e}, \mathbf{a}|\mathbf{f})$
- ...

Tuningkriterium

Ziel: Anpassung der λ_i sodass wir möglichst gute Übersetzungen bekommen.

Dazu:
– benötige Referenzübersetzungen sowie eine Evaluationsmetrik (→ Teil V)
– hier: Accuracy Measure `BLEU-4` (für andere Metriken ähnlich)

Wichtig:
– Tuning sollte auf Daten laufen, auf denen *nicht* getestet wird (genannt *development set*).
– möglichst viele Daten einbeziehen, aber hohe Laufzeiten (500–5000 Sätze in der Praxis)

Erinnerung: BLEU-4

BLEU-4:

- Eingabe: Referenzübersetzungen $\mathbf{r}_s$ und Hypothesen $\mathbf{e}_s$, $s = 1, \ldots, S$ ($\mathbf{a}_s$ wird nicht benötigt).
- Ausgabescore zusammengesetzt aus:
  – Längenstrafterm (falls $\sum_s |\mathbf{e}_s| < \sum_s |\mathbf{r}_s|$)
  – n-gram Precisions für $n = 1, 2, 3, 4$

Beachte: BLEU-4 arbeitet auf dem Korpuslevel, nicht auf dem Satzlevel

$$\texttt{BLEU-4}(\{\mathbf{e}_s|\mathbf{r}_s\}) = \texttt{lenp}\Big(\sum_s |\mathbf{e}_s| \Big| \sum_s |\mathbf{r}_s|\Big) \prod_{n=1}^{4} \frac{\texttt{match}_n(\mathbf{e}_s|\mathbf{r}_s)}{\sum_s |\mathbf{r}_s|}$$

hier: $\texttt{match}_n(\mathbf{e}|\mathbf{r})$ = Anzahl n-gramme in $\mathbf{e}$, die auch in $\mathbf{r}$ sind.

Tuningkriterium - formal

$$\max_{\{\lambda_i\}} \texttt{BLEU-4}\left(\left\{ \operatorname*{argmax}_{\mathbf{e}} \max_{\mathbf{a}} \exp\left(\sum_i \lambda_i h_i(\mathbf{e},\mathbf{a}|\mathbf{f}_s)\right) \middle| s = 1,\ldots,S \right\}\right)$$

Nach Weglassen von exp :

$$\max_{\{\lambda_i\}} \texttt{BLEU-4}\left(\left\{ \operatorname*{argmax}_{\mathbf{e}} \max_{\mathbf{a}} \sum_i \lambda_i h_i(\mathbf{e},\mathbf{a}|\mathbf{f}_s) \middle| s = 1,\ldots,S \right\}\right)$$

Für unsere Zwecke: Betrachtung aller möglichen $\mathbf{e}$ zu komplex:

– wir können nicht (nennenswert) rekombinieren

(die jew. $\texttt{match}_n$ hängen von der gesamten (partiellen) Hypothese ab.)

⇒ sequentielle Betrachtung aller (berücksichtigten) $\mathbf{e}$

⇒ Einschränkung auf einige gute $\mathbf{e}$ → N-best Listen (vgl. Teil IV)

Üblich: $N = 100$

Reduktion auf das Wesentliche

Gegeben für (eine Iteration) des Tunings:
– pro Quellsatz $\mathbf{f}_s$ $(s = 1, \ldots, S)$:
Hypothesen $\mathbf{e}_{s,k}, k = 1, \ldots 100$, mit zugehörigen $\mathbf{a}_{s,k}$
– ermittelt durch den Decoder für gegebene λ_i (anfangs $\lambda_i = 1$)

Für Tuning lediglich wichtig:

- Zur Ermittlung der Wks (für verschiedene λ_i):
  Werte der $h(\mathbf{e}_{s,k}, \mathbf{a}_{s,k}|\mathbf{f}_s)$
- Zur Ermittlung des BLEU-4-Scores:
  – Hypothesenlänge $|\mathbf{e}|$
  – für $n = 1, 2, 3, 4$: Anzahl n-gram Matches zwischen $\mathbf{e}_{s,k}$ und $\mathbf{r}_s$
  Notation: $\mathtt{match}_n(\mathbf{e}_{s,k})$ oder $\mathtt{match}_n^{s,k}$

$\Rightarrow$ die $\mathbf{e}_{s,k}$ und $\mathbf{a}_{s,k}$ selbst müssen wir nicht im Speicher halten

Tuning auf N-best Listen

Kriterium:

$$\max_{\{\lambda_i\}} \texttt{BLEU-4}\left(\left\{ \arg\max_{k=1,\ldots N} \sum_i \lambda_i h_i(\mathbf{e}_{s,k}, \mathbf{a}_{s,k}|\mathbf{f}_s) \middle| s = 1, \ldots, S \right\}\right)$$

Im Folgenden: Abkürzung $h_{i,k,s} = h_i(\mathbf{e}_{s,k}, \mathbf{a}_{s,k}|\mathbf{f}_s)$

Bei F Features im log-linearen Modell: Optimierung über $\mathbb{R}^F$

lokale Optimierung (keine globalen Algorithmen bekannt)

– hier betrachtet: Powell-Suche

– Alternativen: Downhill-Simplex, Sampling, Maximum Entropy

Powell-Suche

Kriterium (wiederholt):

$$\max_{\boldsymbol{\lambda}} \texttt{BLEU-4}\left(\left\{ \arg\max_{k=1,\ldots N} \sum_i \lambda_i h_{i,k,s} \,\middle|\, s = 1,\ldots,S \right\}\right)$$

Für Powell-Suche:

1. wähle initialen Punkt $\boldsymbol{\lambda}$ (z.B. $\lambda_i = 1\ \forall i$)
2. wiederhole einige Male:
   - wähle eine Richtung $\boldsymbol{\lambda}_P$

     z.B. in Iteration k: $\lambda_{P,i} = 1$ für $i = k \bmod F$, sonst $\lambda_{P,i} = 0$

     oder: zufällige Richtung
   - Line Search (1D-Optimierung): finde

     $$\alpha^* = \arg\max_{\alpha \in \mathbb{R}} \texttt{BLEU-4}\left(\left\{ \arg\max_{k=1,\ldots,N} \sum_i (\lambda_i + \alpha\lambda_{P,i})\, h_{i,k,s} \,\middle|\, s \right\}\right)$$

   - setze nun $\lambda_i := \lambda_i + \alpha^* \lambda_{P,i}$

Powell-Suche : Vereinfachung

Vereinfachung des Line-Search Problems

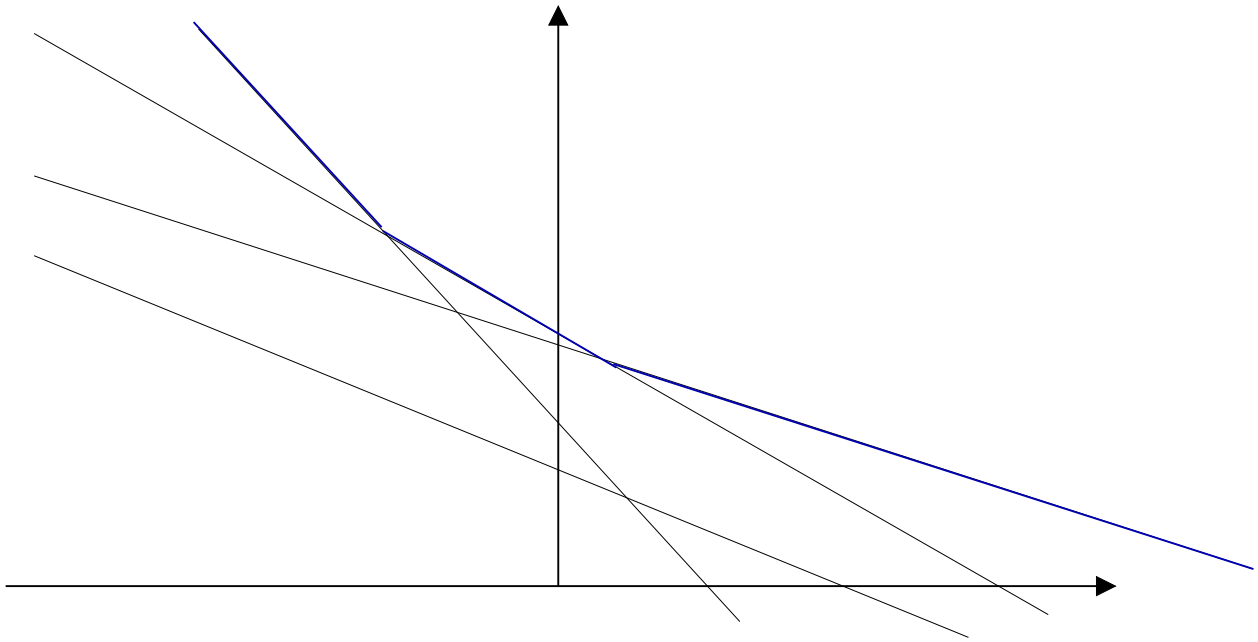
$$\arg\max_{\alpha \in \mathbb{R}} \texttt{BLEU-4}\left(\left\{ \arg\max_{k=1,\ldots,N} \sum_i (\lambda_i + \alpha \lambda_{P,i})\, h_{i,k,s} \,\middle|\, s \right\}\right)$$

$$= \arg\max_{\alpha \in \mathbb{R}} \texttt{BLEU-4}\left(\left\{ \arg\max_{k=1,\ldots,N} o_{k,s} + \alpha c_{k,s} \,\middle|\, s \right\}\right)$$

mit Offset $o_{k,s} = \sum_i \lambda_i\, h_{i,k,s}$
und Koeffizient $c_{k,s} = \sum_i \lambda_{P,i}\, h_{i,k,s}$

Illustration

Veranschaulichung von $\arg \max_{k=1,\ldots,N} o_{k,s} + \alpha c_{k,s}$



Gute Nachricht: Die Funktion ist konvex und kann durch endlich viele Liniensegmente dargestellt werden.

Geschlossene Repräsentation

Gewünscht: geschlossene Repräsentation
von $\arg\max_{k=1,\ldots,N} o_{k,s} + \alpha c_{k,s}$ in Abhängigkeit von α.

- für jedes α: Maximierung über lineare Funktionen.
- wenn zwei lineare Funktionen k_1 und k_2 unterschiedliche Koeffizienten haben (o.B.d.A. $c_{k_1,s} > c_{k_2,s}$), so treffen sie sich in einem Punkt (genannt *Breakpoint*) α_{k_1,k_2} mit:

$$o_{k_1,s} + \alpha_{k_1,k_2} c_{k_1,s} = o_{k_2,s} + \alpha_{k_1,k_2} c_{k_2,s}$$

  für $\alpha \geq \alpha_{k_1,k_2}$ liegt die Fkt. k_1 über der Fkt. k_2
  für $\alpha \leq \alpha_{k_1,k_2}$: umgekehrt.
- Ist Fkt. $\bar{k}$ die einzige optimale Fkt. für ein bestimmtes α, so ist sie optimal bis zu dem Punkt $\bar{\alpha} = \min_{k': c_{k',s} > c_{\bar{k},s}} \alpha_{\bar{k},k'}$

Konstruktion einer geschlossenen Repräsentation

- Für $\alpha = -\infty$: optimale Funktion hat minimales $c_{k,s}$.
  Falls mehrere Funktionen mit minimalem $c_{k,s}$: es gewinnt diejenige mit maximalem $o_{k,s}$
- Startpunkt: setze $\alpha := -\infty$, $\bar{k}$ wie oben
- Iteriere nun:
  falls $\nexists k' : c_{k',s} > c_{\bar{k},s}$:
  notiere $\bar{k}$ als optimal für Intervall (α, ∞). Stoppe.
  sonst ermittele nächsten *Breakpoint*: $\bar{\alpha} = \min_{k':c_{k',s}>c_{\bar{k},s}} \alpha_{\bar{k},k'}$
  notiere $\bar{k}$ als optimal für Intervall $(\alpha, \bar{\alpha})$.
  Setze $\alpha := \bar{\alpha}$, $\bar{k} = \arg\min_{k':c_{k',s}>c_{\bar{k},s}} \alpha_{\bar{k},k'}$

Line Search

Nach der Konstruktion der geschlossenen Repräsentation:
pro Quellsatz eine endliche Menge von Intervallen mit zugehörigen Übersetzungen. (Anz. Intervalle $\leq$ Anz. Hypothesen in der N-best Liste)

Prinzipiell also:
Minimierung über eine unendliche Menge ($\alpha \in \mathbb{R}$) reduziert auf Abarbeitung endliche vieler Intervalle

Aber: BLEU-4 arbeitet auf dem Korpuslevel
$\Rightarrow$ Erstelle gemeinsame Liste von Breakpoints, die zu mindestens einem Satz gehören (Intervalle folgen daraus).
$\Rightarrow$ dann Abarbeitung jedes Intervalls ($\rightarrow$ nächste Folie).

Abarbeitung eines Intervalls

Für jedes Intervall interessiert uns der BLEU-4 Score der zugehörigen Übersetzungen.

Dazu:

- für das gegebene Intervall, bezeichne mit $\mathbf{e}_{k,s}$ die zugehörige maximierende Hypothese für Satz s.
- für den Längenstrafterm addiere die Längen aller Hyps: $\sum_s |\mathbf{e}_{k,s}|$
- für $n = 1, 2, 3, 4$ berechne $\sum_s \texttt{match}_n(\mathbf{e}_{k,s}|\mathbf{r}_s)$ wobei k die jeweilige notierte Übersetzung bezeichnet.
- $\Rightarrow$ der BLEU-4 Score kann nun berechnet werden.

Am Ende: gebe das α mit dem besten BLEU-4 Score aus.

Iterierte Powell-Suche

- Erinnerung: $\operatorname{argmax}_{\mathbf{e}} \max_{\mathbf{a}} \sum_i \lambda_i h_i(\mathbf{e}, \mathbf{a}|\mathbf{f}_s)$ approximiert durch N-best Listen für jeden Satz s
- N-best Listen erstellt für vorheriges $\boldsymbol{\lambda}$
  Aber: Wenn sich $\boldsymbol{\lambda}$ stark ändert werden sich meist auch die N-besten Hypothesen stark ändern.
  (z.B. vorher Sätze sehr lang $\Rightarrow$ Wortpenalty erhöht $\Rightarrow$ jetzt Sätze sehr kurz)
- Deswegen: Iterierung des Prozesses. Für jedes neue $\boldsymbol{\lambda}$ berechne neue N-best Liste (für jeden Satz) und **vereinige** diese mit der vorherigen
  $\Rightarrow$ mit der Zeit ergibt sich eine deutlich repräsentativere Menge von Hypothesen
- Ende des Prozesses: wenn sich $\boldsymbol{\lambda}$ nur unmerklich ändert oder nach (meist) 25 Iterationen.