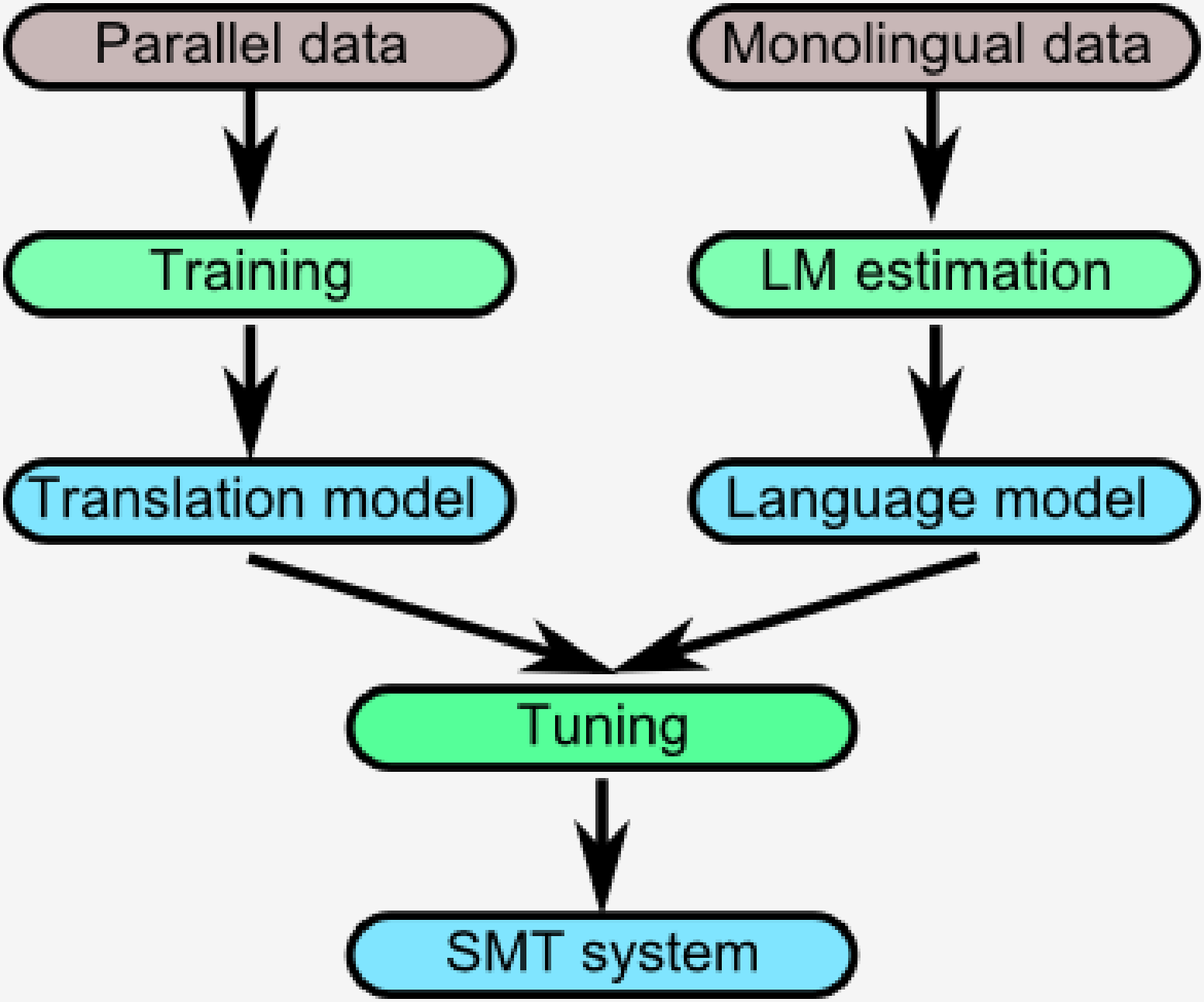
Statistical Machine Translation

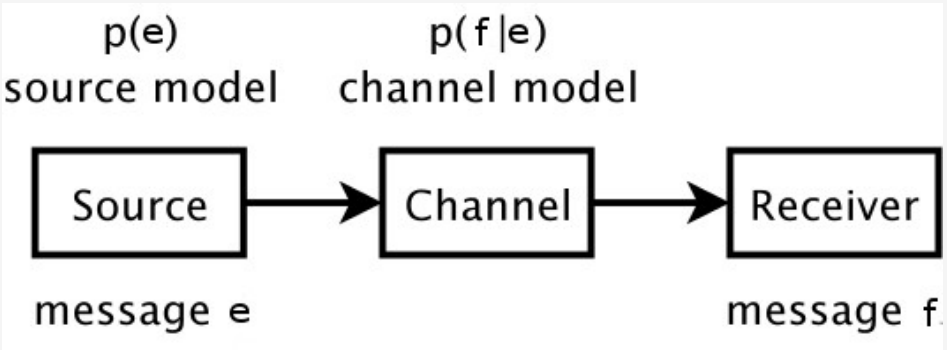
-language models-

Artem Sokolov
Computerlinguistik
Universität Heidelberg
Sommersemester 2015

material from P. Koehn, R. Shapire, D. McAllester


Parallel data
Monolingual data
Training
LM estimation
Translation model
Language model
Tuning
SMT system

$$\arg\max_e p(e|f) = \arg\max_e \underbrace{p(f|e)}_{\text{transl. model}} \underbrace{p(e)}_{\text{lang. model}}$$



- $p(e)$ = **language model**
- $p(f|e)$ = acoustic model
- SMT must deal with word reordering

reordering

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

right word choice:

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

$$p(x|y) = \frac{p(x,y)}{p(y)}$$
$$p(y)p(x|y) = p(x,y)$$

$$\begin{aligned} p(w_1^k) &= p(w_1, w_2, \ldots, w_k) \\ &= p(w_1)p(w_2|w_1)p(w_3|w_1, w_2) \ldots p(w_k|w_1, \ldots w_{k-1}) \end{aligned}$$

Example of a 3-gram model

the green (total: 1748)		
Word	Count	Prob.
paper	801	0.458
group	640	0.367
light	110	0.063
party	27	0.015
ecu	21	0.012

the red (total: 225)		
Word	Count	Prob.
cross	123	0.547
tape	31	0.138
army	9	0.040
card	7	0.031
,	5	0.022

the blue (total: 54)		
Word	Count	Prob.
box	16	0.296
.	6	0.111
flag	6	0.111
,	3	0.056
angel	3	0.056

“I would like to commend the rapporteur on this work”

Prediction	p_{LM}	$-\log_2 p_{\text{LM}}$
p_{LM}(*i*|</s><s>)	0.109	3.197
p_{LM}(would|<s>*i*)	0.144	2.791
p_{LM}(like|i would)	0.489	1.031
p_{LM}(to|would like)	0.905	0.144
p_{LM}(commend|like to)	0.002	8.794
p_{LM}(the|to commend)	0.472	1.084
p_{LM}(rapporteur|commend the)	0.147	2.763
p_{LM}(on|the rapporteur)	0.056	4.150
p_{LM}(his|rapporteur on)	0.194	2.367
p_{LM}(work|on his)	0.089	3.498
p_{LM}(.|his work)	0.290	1.785
p_{LM}(</s>|work .)	0.99999	0.000014
	Average	2.634

“I would like to commend the rapporteur on this work”

Word	Unigram	Bigram	Trigram	4-gram
i	6.684	3.197	3.197	3.197
would	8.342	2.884	2.791	2.791
like	9.129	2.026	1.031	1.290
to	5.081	0.402	0.144	0.113
commend	15.487	12.335	8.794	8.633
the	3.885	1.402	1.084	0.880
rapporteur	10.840	7.319	2.763	2.350
on	6.765	4.140	4.150	1.862
his	10.678	7.316	2.367	1.978
work	9.993	4.816	3.498	2.394
.	4.896	3.020	1.785	1.510
</s>	4.828	0.005	0.000	0.000
Average	8.051	4.072	2.634	2.251
Perplexity	265.136	16.817	6.206	4.758

unsmoothed:

$$p = \frac{c}{n}, \qquad \begin{matrix} c\text{=number of n-gram in corpus} \\ n\text{=count of history} \end{matrix} \tag{1}$$

(2)

unsmoothed: $$p = \frac{c}{n}, \qquad \begin{matrix} c\text{=number of n-gram in corpus} \\ n\text{=count of history} \end{matrix} \tag{1}$$

"Add-one"-smoothing: $$p = \frac{c+1}{n+v}, \qquad v = \text{size of vocabulary} \tag{2}$$

$$p = \frac{c + \alpha}{n + \alpha v}, \qquad \alpha < 1, \alpha \text{ optimized on held-out set} \tag{3}$$

$$p = \frac{c + \alpha}{n + \alpha v}, \qquad \alpha < 1, \alpha \text{ optimized on held-out set} \tag{3}$$

Example: 2-grams in Europarl

Count	**Adjusted count**		**Test count**
c	"add 1"	"add α	t_c
0	0.00378	0.00016	0.00016

$$p = \frac{c + \alpha}{n + \alpha v}, \qquad \alpha < 1, \alpha \text{ optimized on held-out set} \tag{3}$$

Example: 2-grams in Europarl

Count	**Adjusted count**		**Test count**
c	"add 1"	"add α	t_c
0	0.00378	0.00016	0.00016
6	0.02644	5.74266	5.33762
8	0.03399	7.65683	7.15074
10	0.04155	9.57100	9.11927
20	0.07931	19.14183	18.95948

$$p = \frac{c + \alpha}{n + \alpha v}, \qquad \alpha < 1, \alpha \text{ optimized on held-out set} \tag{3}$$

Example: 2-grams in Europarl

Count	Adjusted count		Test count
c	"add 1"	"add α	t_c
0	0.00378	0.00016	0.00016
1	0.00755	0.95725	0.46235
2	0.01133	1.91433	1.39946
3	0.01511	2.87141	2.34307
4	0.01888	3.82850	3.35202
5	0.02266	4.78558	4.35234
6	0.02644	5.74266	5.33762
8	0.03399	7.65683	7.15074
10	0.04155	9.57100	9.11927
20	0.07931	19.14183	18.95948

$$r^* = \frac{T_r^1 + T_r^2}{N_r^1 + N_r^2}$$

Count r	Count of counts N_r	Count in held-out T_r	Exp. count $E[r] = T_r/N_r$	Test count t_c
0	7,515,623,434	938,504	0.00012	0.00016
1	753,777	353,383	0.46900	0.46235
2	170,913	239,736	1.40322	1.39946
3	78,614	189,686	2.41381	2.34307
4	46,769	157,485	3.36860	3.35202
5	31,413	134,653	4.28820	4.35234
6	22,520	122,079	5.42301	5.33762
8	13,586	99,668	7.33892	7.15074
10	9,106	85,666	9.41129	9.11927
20	2,797	53,262	19.04992	18.95948

$$r^* = (r+1)\frac{N_{r+1}}{N_r}$$

N_r – # of n-grams that occur r times in our corpus
N_0 – total # of n-grams.

Count r	Count of counts N_r	Adjusted count r^*	Test count t
0	7,514,941,065	0.00015	0.00016
1	1,132,844	0.46539	0.46235
2	263,611	1.40679	1.39946
3	123,615	2.38767	2.34307
4	73,788	3.33753	3.35202
5	49,254	4.36967	4.35234
6	35,869	5.32928	5.33762
8	21,693	7.43798	7.15074
10	14,880	9.31304	9.11927
20	4,546	19.54487	18.95948

$$
\begin{aligned}
p_I(w_3|w_1, w_2) = \quad & \lambda_1 \; p_1(w_3) \\
& + \lambda_2 \; p_2(w_3|w_2) \\
& + \lambda_3 \; p_3(w_3|w_1, w_2)
\end{aligned}
$$

$$
\forall \lambda_n : 0 \leq \lambda_n \leq 1
$$

$$
\sum_n \lambda_n = 1
$$

$$p_n^I(w_i|w_{i-n+1},...,w_{i-1}) = \lambda_{w_{i-n+1},...,w_{i-1}} \; p_n(w_i|w_{i-n+1},...,w_{i-1}) + (1 - \lambda_{w_{i-n+1},...,w_{i-1}}) \; p_{n-1}^I(w_i|w_{i-n+2},...,w_{i-1})$$

$$p_n^{BO}(w_i|w_{i-n+1},...,w_{i-1}) =$$

$$\begin{cases} d_n(w_{i-n+1},...,w_{i-1})\ p_n(w_i|w_{i-n+1},...,w_{i-1}) \\ \qquad\qquad \text{if } \mathsf{count}_n(w_{i-n+1},...,w_i) > k \\ \alpha_n(w_i|w_{i-n+1},...,w_{i-1})\ p_{n-1}^{BO}(w_i|w_{i-n+2},...,w_{i-1}) \\ \qquad\qquad \text{otherwise} \end{cases}$$

$$p_n^{BO}(w_i|w_{i-n+1},...,w_{i-1}) = \begin{cases} d_n(w_{i-n+1},...,w_{i-1})\ p_n(w_i|w_{i-n+1},...,w_{i-1}) \\ \qquad \text{if } \text{count}_n(w_{i-n+1},...,w_i) > k \\ \alpha_n(w_i|w_{i-n+1},...,w_{i-1})\ p_{n-1}^{BO}(w_i|w_{i-n+2},...,w_{i-1}) \\ \qquad \text{otherwise} \end{cases}$$

- k usually a small number – 0 or found emprically
- Good-Turing estimation reduces counts (and probabilites), we can just use the discount GT gives as a discounting function $d_n(w_1,...,w_{n-1})$.
- $\alpha_n(w_i|w_{i-n+1},...,w_{i-1})$ then collect all the missing probability mass:

$$p_n^{BO}(w_i|w_{i-n+1},...,w_{i-1}) = \begin{cases} d_n(w_{i-n+1},...,w_{i-1})\ p_n(w_i|w_{i-n+1},...,w_{i-1}) \\ \qquad \text{if } \mathsf{count}_n(w_{i-n+1},...,w_i) > k \\ \alpha_n(w_i|w_{i-n+1},...,w_{i-1})\ p_{n-1}^{BO}(w_i|w_{i-n+2},...,w_{i-1}) \\ \qquad \text{otherwise} \end{cases}$$

- k usually a small number – 0 or found emprically
- Good-Turing estimation reduces counts (and probabilites), we can just use the discount GT gives as a discounting function $d_n(w_1,...,w_{n-1})$.
- $\alpha_n(w_i|w_{i-n+1},...,w_{i-1})$ then collect all the missing probability mass:

$$\beta(w_{i-n+1},...,w_{i-1}) = 1 - \sum_{\mathsf{count}_n(w_{i-n+1},...,w_i)>k} d_n(w_{i-n+1},...,w_{i-1})\ p_n(w_i|w_{i-n+1},...,w_{i-1})$$

$$p_n^{BO}(w_i|w_{i-n+1},...,w_{i-1}) = \begin{cases} d_n(w_{i-n+1},...,w_{i-1})\ p_n(w_i|w_{i-n+1},...,w_{i-1}) \\ \qquad \text{if } \text{count}_n(w_{i-n+1},...,w_i) > k \\ \alpha_n(w_i|w_{i-n+1},...,w_{i-1})\ p_{n-1}^{BO}(w_i|w_{i-n+2},...,w_{i-1}) \\ \qquad \text{otherwise} \end{cases}$$

- k usually a small number – 0 or found emprically
- Good-Turing estimation reduces counts (and probabilites), we can just use the discount GT gives as a discounting function $d_n(w_1,...,w_{n-1})$.
- $\alpha_n(w_i|w_{i-n+1},...,w_{i-1})$ then collect all the missing probability mass:

$$\beta(w_{i-n+1},...,w_{i-1}) = 1 - \sum_{\text{count}_n(w_{i-n+1},...,w_i)>k} d_n(w_{i-n+1},...,w_{i-1})\ p_n(w_i|w_{i-n+1},...,w_{i-1})$$

$$\alpha_n(w_i|w_{i-n+1},...,w_{i-1}) = \frac{\beta(w_{i-n+1},...,w_{i-1})}{\sum_{\text{count}_n(w_{i-n+1},...,w_i)\leq k} p_{n-1}^{BO}(w_i|w_{i-n+2},...,w_{i-1})}$$