Statistical Language Models

Nicola Bertoldi
FBK-irst Trento, Italy

Fifth MT Marathon, Le Mans, France

13-18 September 2010

Outline

- Introduction
- Role of LM in ASR and MT
- Evaluation of Language Models
- n-gram Language Models
- Smoothing and Discounting
- Enhancement to n-gram LM
- Notes about training

Credits:

- M. Federico (FBK-irst, Trento)

Statistical Language Model

What is it?

- A Language Model provides a score for any word sequences to determine how likely they are:
  - ASR output: “recognize speech” or “wreck a nice beach”?
- probability distribution over the sequences of a given language V^∞:

$$\Pr(w_1^T), \quad w_i \in V, \quad i = 1, \ldots, T, \quad \exists T \tag{1}$$

What is it for?

- any application aiming at producing a fluent output
  - Speech Recognition
  - Machine Translation
  - Optical Character Recognition
  - Spelling Correction
  - ... and many other Statistical tasks coping with strings

Fundamental Equation of ASR

Goal: find the words $\mathbf{w}^*$ in a speech signal $\mathbf{x}$ such that:

$$\mathbf{w}^* = \operatorname*{argmax}_{\mathbf{w}} \Pr(\mathbf{x} \mid \mathbf{w}) \Pr(\mathbf{w}) \qquad (2)$$

Problems:

- language modeling (LM): estimating $\Pr(\mathbf{w})$
- acoustic modeling (AM): estimating $\Pr(\mathbf{x} \mid \mathbf{w})$
- search problem: computing Eq. (2)

AM sums over hidden state sequences $\mathbf{s}$ a Markov process of $(\mathbf{x}, \mathbf{s})$ from $\mathbf{w}$

$$\Pr(\mathbf{x} \mid \mathbf{w}) = \sum_{\mathbf{s}} \Pr(\mathbf{x}, \mathbf{s} \mid \mathbf{w})$$

Hidden Markov Model: hidden states “link” speech frames to words.

Fundamental Equation of SMT

Goal: find the English string $\mathbf{f}$ translating the foreign text $\mathbf{f}$ such that:

$$\mathbf{e}^* = \underset{\mathbf{e}}{\operatorname{argmax}} \Pr(\mathbf{f} \mid \mathbf{e}) \Pr(\mathbf{e}) \tag{3}$$

Problems:

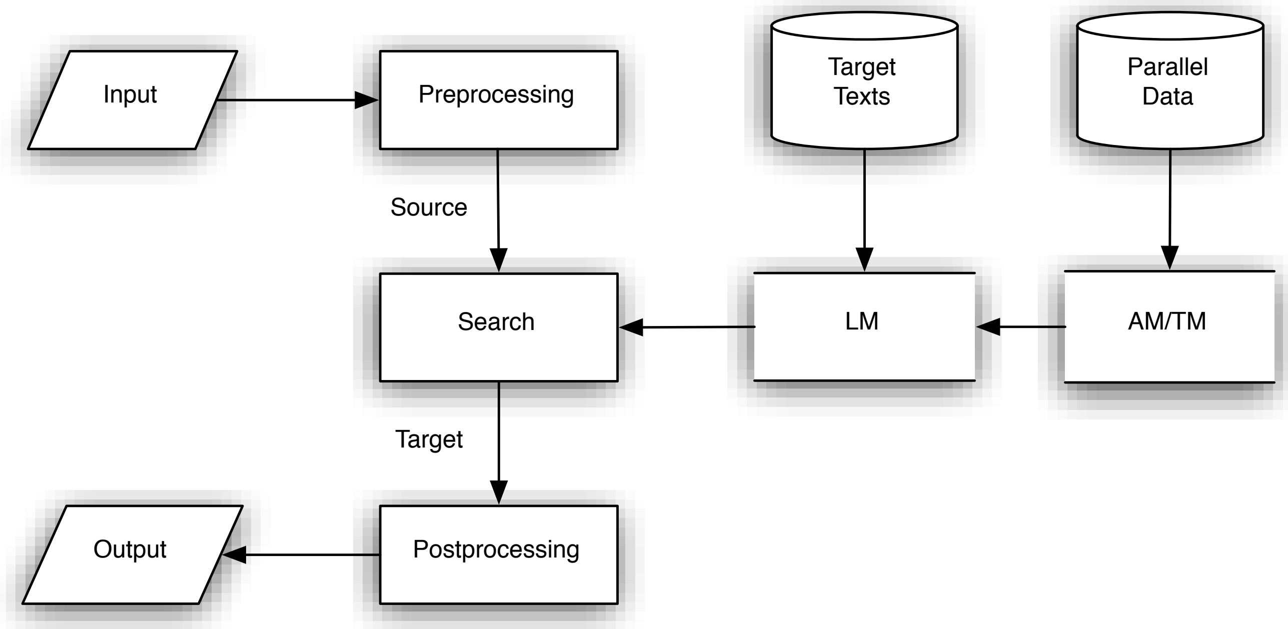
- language modeling (LM): estimating $\Pr(\mathbf{e})$
- translation modeling (TM): estimating $\Pr(\mathbf{f} \mid \mathbf{e})$
- search problem: computing Eq. (3)

TM sums over hidden alignments $\mathbf{a}$ a stochastic process generating $(\mathbf{f}, \mathbf{a})$ from $\mathbf{e}$.

$$\Pr(\mathbf{f} \mid \mathbf{e}) = \sum_{\mathbf{a}} \Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$$

Alignment Models: hidden alignments "link" foreign words with English words.

ASR and MT Architectures



- Parallel data are samples of observations $(\mathbf{x}, \mathbf{w})$ and $(\mathbf{f}, \mathbf{e})$
- AM and TM can be machine-learned without observing $\mathbf{s}$ and $\mathbf{a}$
- AM is "simpler" than TM, because of monotonicity of $(\mathbf{x}, \mathbf{s})$ and $\mathbf{w}$

- LM is trained on monolingual texts

Language Model Evaluation

- Indirect: impact on task
  - Word Error Rate in ASR
  - BLEU score for MT
  - Precision and Recall for Spelling Correction

- Direct: capability of predicting words of your language
  - how difficult is the guess of:
    * the next digit of a phone number (after $+39339728$)? 10
    * the PIN number (of 5 digits)? 10^5
    * the next word after "the UEFA Champions"? 1 (if you are a football fan)
  - perplexity measure

Language Model Perplexity

The perplexity (PP) measure is the geometric average inverse probability

$$PP = \sqrt[T]{\frac{1}{\Pr(w_1^T)}} \quad (4)$$

but usually expressed as follows (for the sake of computation):

$$PP = 2^{LP} \quad \text{where} \quad LP = -\frac{1}{T}\log_2 p(w_1^T) \quad (5)$$

- w_1^T is a sufficiently long test sample
- $p(w_1^T)$ is the LM probability

Language Model Perplexity

The perplexity (PP) measure is the geometric average inverse probability

$$PP = 2^{LP} \quad \text{where} \quad LP = -\frac{1}{T}\log_2 p(w_1^T) \qquad (6)$$

Properties:

- $0 \leq PP \leq |V|$ (size of the vocabulary V)
- predictions are as good as guessing among PP equally likely options
- the cross-entropy of the model on test sample is 2^{PP}

- the true model has the lowest possible PP
- lower the PP, closer your model to the true model

Good: there is typical strong correlation between PP and BLUE scores!

Statistical Language Model

Goal: given a text $w_1^T = w_1 \ldots, w_t, \ldots, w_T$, where $w_i \in V$, we can compute its probability by:

$$\Pr(w_1^T) = \Pr(w_1) \prod_{t=2}^{T} \Pr(w_t \mid h_t) \quad (7)$$

where $h_t = w_1, \ldots, w_{t-1}$ indicates the history of word w_t.

Issues:

- $\Pr(w_t \mid h_t)$ becomes difficult to estimate as the history h_t grows
  - parameter space: exponential amount of parameters
  - data sparseness: most of $(w \mid h)$ are rare events even in large corpora.

Solutions:

- take an approximation for the history: $h_t \approx w_{t-n+1} \ldots w_{t-1}$
  - n-gram approximation: $h_t \approx w_{t-n+1} \ldots w_{t-1}$
  - class-based approximation: $h_t \approx c(w_1) \ldots c(w_{t-1})$

N-gram Language Model

Goal: given a text $w_1^T = w_1 \ldots, w_t, \ldots, w_T$ we can compute its probability by:

$$\Pr(w_1^T) = \Pr(w_1) \prod_{t=2}^{T} \Pr(w_t \mid w_{t-n+1} \ldots w_{t-1}) \quad (8)$$

where the n-gram approximation is applied: $h_t \approx w_{t-n+1} \ldots w_{t-1}$

e.g. Full history: $\Pr(\texttt{Parliament} \mid \texttt{I declare resumed the session of the European})$

$3 - gram$: $\Pr(\texttt{Parliament} \mid \texttt{the European})$

The choice of n determines the complexity of the LM (# of parameters):

- bad: no magic recipe about the optimal order n for a given task
- good: language models can be evaluated quite cheaply, because based on n-grams statistics gathered from a training corpus

FONDAZIONE BRUNO KESSLER

N-gram Probabilities

Estimating n-gram probabilities $\Pr(w_t \mid w_{t-n+1} \ldots w_{t-1})$ is not trivial due to:

- parameter space: with 10,000-word V we can form one trillion 3-grams!
- data sparseness: most of 3-grams are rare events even in large corpora.

Relative frequency estimate: MLE of any discrete conditional distribution is:

$$f(w \mid x\ y) = \frac{c(x\ y\ w)}{\sum_{w} c(x\ y\ w)}$$

where counts $c(\cdot)$ are taken over a large training corpus.

Problem: relative frequencies in general overfit the training data

- if the test sample contains a "new" n-gram, then PP $\rightarrow +\infty$
- with 4-grams or 5-grams LM this is largely the most frequent case!

We need smoothing!

Frequency Smoothing

Issue: $f(w \mid x\ y) > 0$ only if w was observed after $x\ y$ in the training data.

Idea: for each w take off some fraction of probability from $f(w \mid x\ y)$ and redistribute the total to words never observed after $x\ y$.

- the discounted frequency $f^*(w \mid x\ y)$ satisfies:

$$0 \leq f^*(w \mid x\ y) \leq f(w \mid x\ y) \qquad \forall x, y, w \in V$$

  Notice: in general $f^*(w \mid x\ y)$ does not sum up to 1!

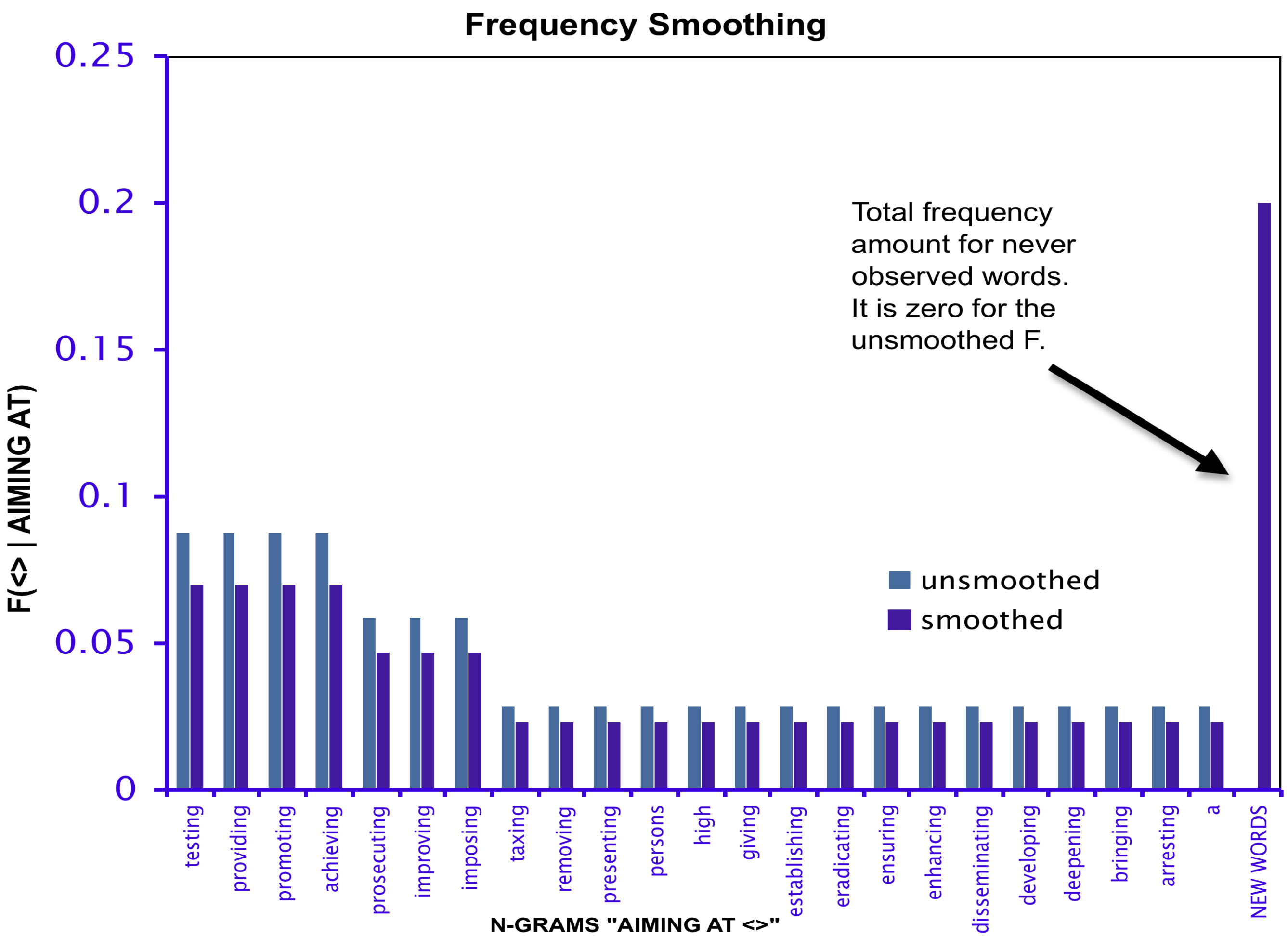
- the "total discount" is called zero-frequency probability $\lambda(x\ y)$[1]:

$$\lambda(x\ y) = 1.0 \ - \ \sum_{w \in V} f^*(w \mid x\ y)$$

How to redistribute the total discount?

[1]Notice: $\lambda(x\ y) = 1$ if $f(w \mid x\ y) = 0$ for all w, i.e. $c(x\ y) = 0$.

Discounting Example



Frequency Smoothing

Insight: redistribute $\lambda(x\ y)$ according to the lower-order probability $p(w \mid y)$:

Two major hierarchical schemes to compute the smoothed probability $p(w \mid x\ y)$:

- Back-off, i.e. select the best available n-gram approximation:

$$p(w \mid x\ y) = \begin{cases} f^*(w \mid x\ y) & \text{if } f^*(w \mid x\ y) > 0 \\ \alpha_{xy}\lambda(x\ y)p(w \mid y) & \text{otherwise} \end{cases} \tag{9}$$

  where α_{xy} is an appropriate normalization term.

- Interpolation, i.e. sum up the two approximations:

$$p(w \mid x\ y) = f^*(w \mid x\ y) + \lambda(x\ y)p(w \mid y). \tag{10}$$

Smoothed probability are learned bottom-up, starting from 1-grams ...

Frequency Smoothing of 1-grams

Unigram smoothing permits to treat out-of-vocabulary (OOV) words in the LM.

Assumptions:

- $|U|$ is an upper-bound estimate of the size of language vocabulary
- $f^*(w)$ is strictly positive on the observed vocabulary V
- λ is the total discount reserved to OOV words

Then: 1-gram back-off and interpolation collapse to:

$$p(w) = \begin{cases} f^*(w) & \text{if } \; w \in V \\ \lambda \, \frac{1}{(|U|-|V|)} & \text{otherwise} \end{cases} \tag{11}$$

Notice: LMs make also other approximations when an OOV word x appears:

$$p(w \mid h_1 \; x \; h_2) = p(w \mid h_2) \;\; \text{and} \;\; p(x \mid h) = p(x)$$

Important: use a common value $|U|$ when comparing/combining different LMs!

Discounting Methods

Witten-Bell estimate (WB) [Witten and Bell, 1991]

- Insight: learn $\lambda(x\ y)$ by counting "new word" events in 3-grams `x y *`
  - corpus: `x y u x x y t t x y u w x y w x y t u x y u x y t`
  - then $\lambda(x\ y) \propto$ number of "new word" events (i.e. 3)
  - and $f^*(w \mid x\ y) \propto$ relative frequency (linear discounting)
- Solution:

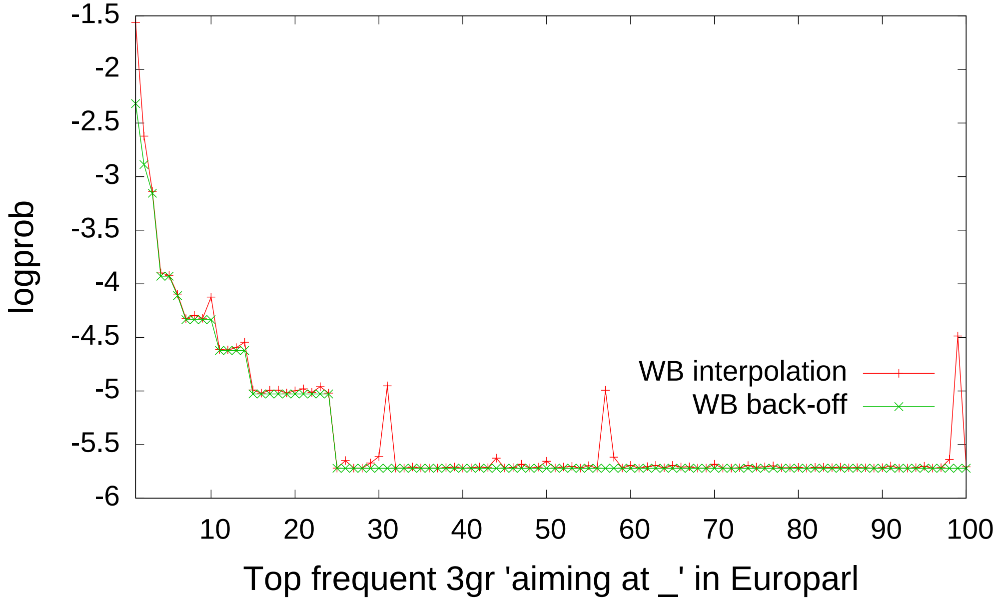
$$\lambda(x\ y) = \frac{n(x\ y\ *)}{c(x\ y) + n(x\ y\ *)} \quad \text{and} \quad f^*(w \mid xy) = \frac{c(x\ y\ w)}{c(x\ y) + n(x\ y\ *)}$$

where $c(x\ y) = \sum_w c(x\ y\ w)$ and $n(x\ y\ *) = |\{w : c(x\ y\ w) > 0\}|$.

- Pros: easy to compute, robust for small corpora, works with artificial data.
- Cons: underestimates probability of frequent n-grams

Discounting Methods

- interpolation and back-off with WB discounting
- trigram LMs estimated on the English Europarl corpus
- logprobs of 3-grams of type `aiming at _` observed in training



- peaks correspond to very probable 2-grams interpolated with f^* respectively: `at that`, `at national`, `at European`
- Practically, interpolation and back-off perform similarly

Discounting Methods

Absolute Discounting (AD) [Ney and Essen, 1991]

- Insight:
  – discount by subtracting a small constant β $(0 < \beta \leq 1)$ from each counts
- Solution:

$$f^*(w \mid x\ y) = max\left\{\frac{c(xyw) - \beta}{c(xy)}, 0\right\} \text{ which gives } \quad \lambda(xy) = \beta\frac{\sum_{w:c(xyw)>1} 1}{c(xy)}$$

where $\beta \approx \frac{n_1}{n_1+2n_2} < 1$ and $n_r = |\{x\ y\ w : c(x\ y\ w) = r\}|$

- Notice:
  – one distinct β for each n-gram order
  – leave-one-out estimate of β on the training data [Ney, Essen and Kneser, 1994]
- Pros: easy to compute, accurate estimate of frequent n-grams.
- Cons: problematic with small and artificial samples.

Discounting Methods

Kneser-Ney method (KN) [Kneser and Ney, 1995]

- Insight:
  - marginals of the higher-order smoothed probs should match the training data
  - count all "back-off" events in 3-grams of type `* y w` (cf. WB method)
  - corpus: `x y w x t y w t x y w u y w t y w u x y w u u y w`

- Solution:

$$f^*(w \mid y) = max\left\{\frac{n(*\ y\ w) - \beta}{n(*\ y\ *)}, 0\right\} \text{ which gives } \quad \lambda(y) = \beta \frac{\sum_{w:n(*\ y\ w)>1} 1}{n(*\ y\ *)}$$

where $n(*\ y\ w) = |\{x : c(x\ y\ w) > 0\}|$ and $n(*\ y\ *) = |\{x\ w : c(x\ y\ w) > 0\}|$

- Pros: better back-off probabilities, can be applied to other methods
- Cons: higher-order probs can not be estimated from lower order probs
- Notice: corrected counts (usually) used only for 1- and 2-grams

Discounting Methods

Modified Kneser-Ney (MKN) [Chen and Goodman, 1999]

- Insight: specific discounting coefficients for unfrequent n-grams
- Solution:

$$f^*(w \mid x\ y) = \frac{c(x\ y\ w) - \beta(c(x\ y\ w))}{c(x\ y)}$$

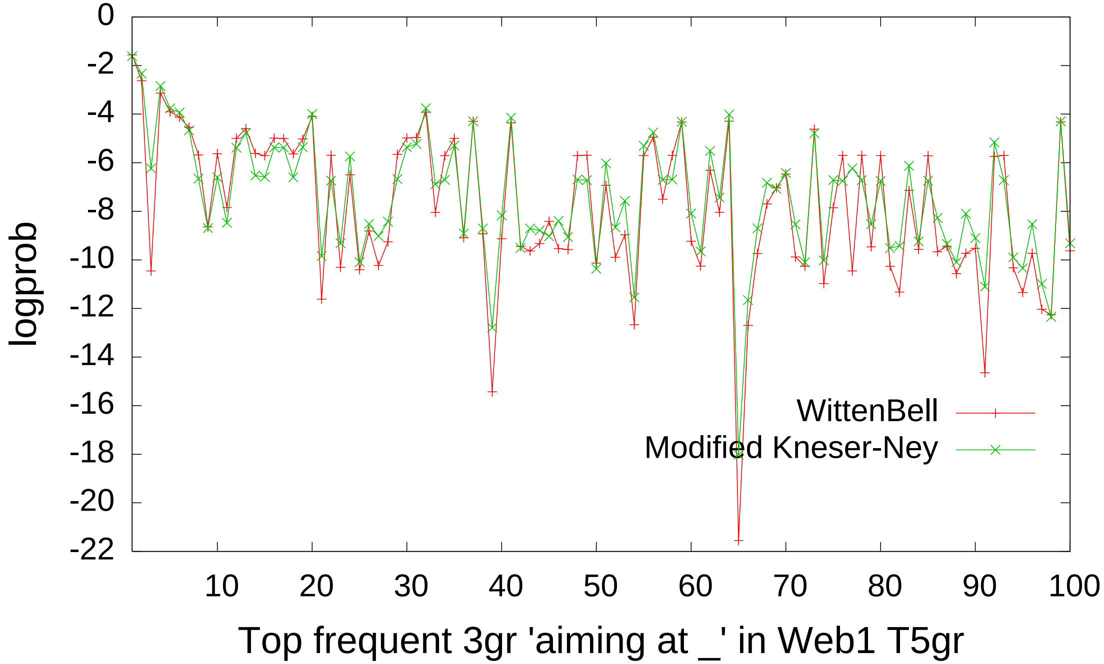
where $\beta(0) = 0$, $\beta(1) = D_1$, $\beta(2) = D_2$, $\beta(c) = D_{3+}$ if $c \geq 3$,

- Notice: coefficients are computed from n_r statistics, corrected counts used for lower order n-grams
- Pros: see previous + more fine grained smoothing
- Cons: see previous + more sensitiveness to noise

Important: LM interpolation with MKN is the most popular training method. Under proper training conditions it gives the best PP and BLEU scores!

Discounting Methods

- train: interpolation with WB and MKN discounting on Europarl
- test: 3-grams of type `aiming at _` are from the Google 1TWeb sample

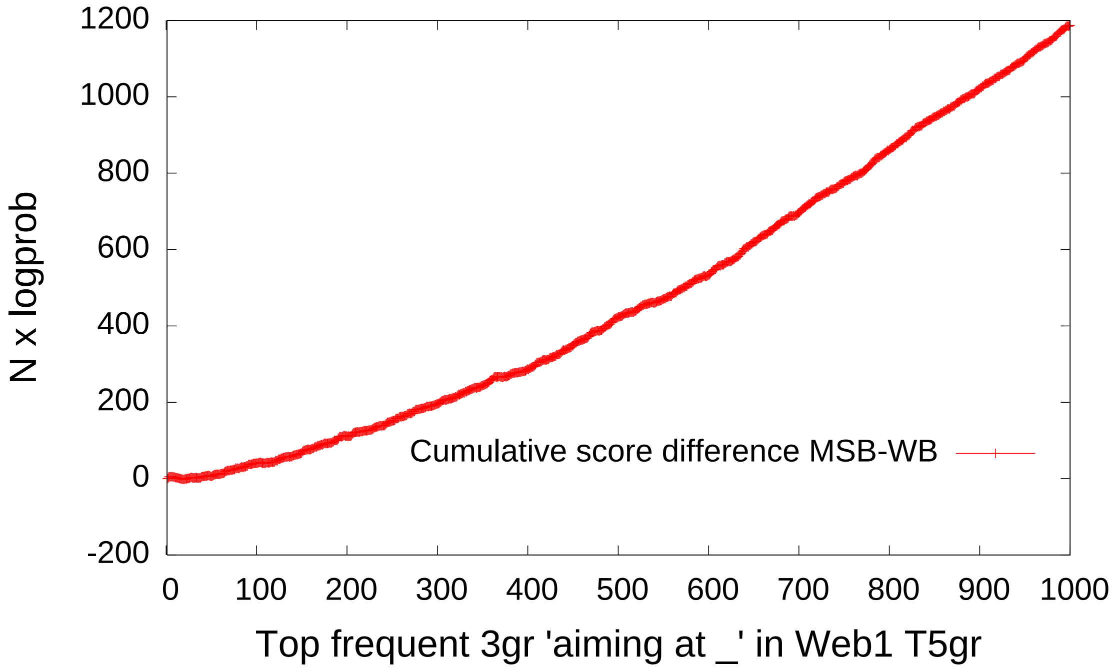


- the trend is the same but MKN outperforms WB smoothing

If you don't believe, check the next slide ....

Discounting Methods

- train: interpolation with WB and MKN discounting on Europarl
- test: 3-grams of type `aiming at _` are from the Google 1TWeb sample
- plot: cumulative score differences between MKN and WB on top 1000 3-grams



Is LM Smoothing Necessary?

or it is enough increasing training data?

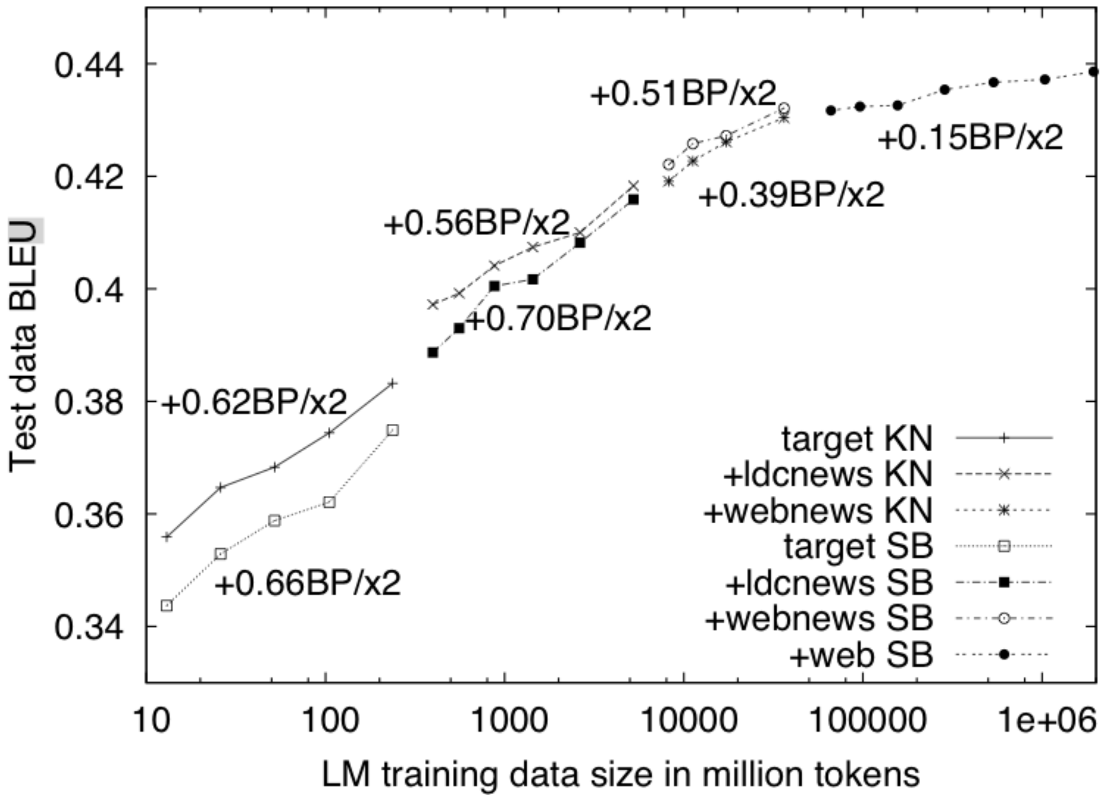
- Stupid Back-off [Brants et al., 2007]
  – simple smoothing, no correct normalization

$$p(w \mid x\ y) = \begin{cases} f(w \mid x\ y) & \text{if } f(w \mid x\ y) > 0 \\ k \cdot p(w \mid y) & \text{otherwise} \end{cases} \quad (12)$$

where $k = 0.4$ and $p(w) = c(w)/N$.

- Comparison between Stupid Back-off (SB) and Modified Kneser-Ney (KN) on the 2006 Arabic-English NIST MT task

Is LM Smoothing Necessary?



- Conclusion: proper smoothing useful up to 1 billion word training data?

Class-based Language Model

- Insight:
  - some words are similar in their meaning and syntactic function
  - the probability of such similar words in similar context are likely similar

- Solution: given the class c_i of any word $w_i \in w_1^T$

$$\Pr(w_1^T) = \Pr(c_1^T) \prod_{t=1}^{T} \Pr(w_t \mid c_t) \qquad (13)$$

- Notice:
  - reduction of data sparseness, more reliable estimation for rare events
  - used when few training data
  - usually combined with the n-gram approximation over classes
  - longer context (larger n)
  - any classification/clustering methods could be applied

Language Model interpolation

Given several LMs $\mathrm{Pr}_i(w \mid h)$ estimated on different training corpora, an interpolated LM can be built by means of:

- External interpolation:

$$\mathrm{Pr}(w \mid h) = \sum_{i=1}^{K} \eta_i \, \mathrm{Pr}_i(w \mid h) \quad (14)$$

- Internal interpolation: Notice: all LMs of the same type

$$f^*(w \mid h) = \sum_{i=1}^{K} \mu_i(h) \; f_i^*(w \mid h) \qquad \lambda(h) = \sum_{i=1}^{K} \mu_i(h) \; \lambda_i(h) \quad (15)$$

- Notice:
  - domain adaptation and adaptation over time
  - split training effort

References

[Brants et al., 2007] Brants, T., et al. (2007). Large language models in machine translation. In EMNLP, pages 858–867, Prague, Czech Republic.

[Chen and Goodman, 1999] Chen, S. F. and Goodman, J. (1999). An empirical study of smoothing techniques for language modeling. Computer Speech and Language, 4(13):359–393.

[Federico and Bertoldi, 2006] Federico, M. and Bertoldi, N. (2006). How many bits are needed to store probabilities for phrase-based translation? In Workshop on Statistical Machine Translation, pages 94–101, New York City.

[Kneser and Ney, 1995] Kneser, R. and Ney, H. (1995). Improved backing-off for m-gram language modeling. In ICASSP, volume 1, pages 181–184, Detroit, MI.

[Ney and Essen, 1991] Ney, H. and Essen, U. (1991). On smoothing techniques for bigram-based natural language modelling. In ICASSP, pages S12.11:825–828, Toronto, Canada.

[Ney, Essen and Kneser, 1994] Ney, H, et al. (1994). On structuring probabilistic dependences in stochastic language modelling Computer Speech and Language, 8:1-38.

[Witten and Bell, 1991] Witten, I. H. and Bell, T. C. (1991). The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. IEEE Transaction Information Theory, IT-37(4):1085–1094.

Goodman, J. http://research.microsoft.com/en-us/um/people/joshuago/icmldescription.htm