A Scalable Distributed Syntactic, Semantic and Lexical Language Model

Shaojun Wang

shaojun.wang@wright.edu

http://www.cs.wright.edu/~swang

Kno.e.sis Center
Department of Computer Science and Engineering
Wright State University

Joint work with Ming Tan, Wenli Zhou, Lei Zheng
What is natural language?

- Natural language is perhaps one of the most intriguing stochastic processes.
  - Natural language encodes messages via complex, hierarchically organized sequences.
  - The local lexical structure of the sequence conveys surface information, while the syntactic structure, encoding long range dependencies, carries deeper semantic information.
Language modeling: A density estimation problem

- Accurately calculate the probability of naturally occurring word sequences in human natural language.
- One of the core technologies of machine translation and speech recognition systems
- Three types of language models
  1. *n*-gram (Markov 1902, Shannon 1948, Jelinek et al. 1970s)
     - Encodes local word interactions
     - The *workhorse* of state-of-the-art machine translation and speech recognition systems
     - Ignores the rich syntactic and semantic structures that constrain natural languages
     - Exploits sentence level syntactic structure
     - Exploits document level semantic content
But

- Each model targets some specific, distinct linguistic phenomena
- Lack of a unified probabilistic framework to encode arbitrary aspects of natural language with tractable training algorithm
How to combine statistical models?

• Linear interpolation
  • Easy to use (good)
  • Makes suboptimal use of components; limited improvement (bad)

• Maximum entropy approach
  • Very popular in natural language processing community due to the work by Berger et al., 1996 and Della Pietra et al., 1997
  • Constrained convex optimization problem

\[
\max_{p(x)} H(p(x))
\]

s.t. \( \sum_x p(x)f_i(x) = \sum_x \tilde{p}(x)f_i(x), \ i = 1, \cdots, N \)
Maximum entropy = MLE for undirected MRFs

• **Dual problem:** Maximum likelihood estimation

\[
p_{\lambda}^{\text{MLE}}(x) = \arg \max_{p_{\lambda}(x)} \sum \tilde{p}(x) \log p_{\lambda}(x)
\]

where \( p_{\lambda}(x) = \frac{1}{\Phi_{\lambda}} \exp(\sum_{i=1}^{N} \lambda_{i} f_{i}(x)) \): undirected MRF

\[
p_{\lambda}^{\text{ME}}(x) = p_{\lambda}^{\text{MLE}}(x)
\]

• **Supervised learning:**
  Only model distributions over explicitly observed features, but for natural language there are hidden structures we do not observe directly  
  (bad)

• **Training is expensive:**
  Mainly in feature expectation and normalization;
  If the statistical model is too complex, training becomes intractable, have to use Markov chain Monte Carlo (MCMC) sampling methods  
  (bad)
How to combine statistical models?

• We propose a principled approach, directed Markov random fields (MRFs)

• Encode word lexical information, sentence syntactic structure, and document semantic content with a tractable parameter estimation algorithm by exploiting certain factorization property

• When applied to combine trigram, PCFG and PLSA (Wang et al. 2005)
  • ∃ a generalized inside-outside algorithm with cubic time complexity
  • Achieved moderate perplexity reduction on 40 million tokens corpus

• We now combine $n$-gram, structured language model (SLM) and PLSA
  • SLM is more complex and powerful than PCFG
Undirected MRFs vs directed MRFs?

- Undirected Markov random fields
  \[ p_\lambda(x) = \exp(\langle \lambda, f(x) \rangle - \log(\Phi_\lambda)) \]

ONE global normalization factor: \( \Phi_\lambda \)

- Directed Markov random fields
  \[ p_\lambda(x) = \prod_j \exp(\langle \lambda_j, f(x_j, \pi_j) \rangle - \log(\Phi_{\lambda_j}(\pi_j))) \]

MANY local normalization factor: \( \Phi_{\lambda_j}(\pi_j) \)
Undirected MRFs vs directed MRFs?

- $n$-gram can be represented either by directed MRF or undirected MRF
  - directed MRF: relative frequency estimates with proper smoothing (good)
  - undirected MRF: feature expectation and normalization, plus optimization such as iterative scaling, coordinate descent, quasi-Newton etc. (bad)
- PCFG and SLM can be represented either by directed MRF or undirected MRF too

Shaojun Wang, Kno.e.sis Center, Department of Computer Science and Engineering, Wright State University  
http://www.cs.wright.edu/~swang
\( n \)-gram

\[
\ldots w_{k-n}w_{k-n+1} \ldots w_{k-1}w_k \ldots
\]

- WORD-PREDICTOR: predicts each word on the basis of previous \( n-1 \) words
Structured language model (SLM)

- The SLM (Chelba and Jelinek 2000) uses syntactic information beyond the regular $n$-gram models to capture sentence level long range dependencies.

- $m$-th order SLM has three operators
  - The **WORD-PREDICTOR** generates the next word $w_k$ with probability $p(w_k|h_{-m}^{-1})$ based on $h_{-m}^{-1} = h_{-1}, \cdots, h_{-m}$, the $m$ most recent exposed headwords in the word-parse $k$-prefix
  
  \[
  h_m \cdots \quad h_2 \quad h_1 = (h_{-1}\text{.word}, h_{-1}\text{.tag})
  \]

  
  
  \[
  (s,SB) \quad \cdots \quad (w_i, t_i) \quad (w_{i+1}, t_{i+1}) \quad \cdots \quad (w_{k-1}, t_{k-1}) w_k \cdots \langle s \rangle
  \]

  
  - The **TAGGER** attaches tag $t_k$ to the most recently generated word $w_k$ with probability $p(t_k|w_k, h_{-1}\text{.tag}, \cdots, h_{-m}\text{.tag})$

  - The **CONSTRUCTOR** builds the partial parse $T_k$ from $T_{k-1}$, $w_k$, and $t_k$ in a series of moves ending with NULL, where a parse move $a$ is made with probability $p(a|h_{-m}^{-1})$; $a \in \{(\text{unary}, \text{NTlabel}), (\text{adjoin-left}, \text{NTlabel}), (\text{adjoin-right}, \text{NTlabel}), \text{null}\)$

Shaojun Wang, Kno.e.sis Center, Department of Computer Science and Engineering, Wright State University  http://www.cs.wright.edu/~swang–p.11/53
Probabilistic latent semantic analysis (PLSA)

Choose a document $d$ with probability $p(d)$;

SEMANTIZER: select a semantic class $g$ with probability $p(g|d)$

WORD-PREDICTOR: predict a word $w$ with probability $p(w|g)$.
Combining $n$-gram, $m$-order SLM and PLSA under directed MRF paradigm

- TAGGER and CONSTRUCTOR in SLM and SEMANTIZER in PLSA remain unchanged
- WORD-PREDICTOR generates the next word, $w_{k+1}$, based on most recent $m$ exposed headwords $h_{-m}^{-1}$ in the word-parse $k$-prefix, $n$-gram history

$$w_{k-n+1}^k = w_{k-n+1}, \ldots, w_k$$ and its semantic content $g_{k+1}$
Likelihood of fully labeled sentence

\[
P_p(W, T, G | d) = \prod_{g \in G} \left( p(g | d)^{\#(g, W, G, d)} \prod_{h_1, \ldots, h_m \in H}^{} \right)
\prod_{w, w-1, \ldots, w-n+1 \in V} p(w | w_{n+1}^{1} h_{m}^{1} g)^{\#(w_{n+1}^{1} w h_{m}^{1} g, W^l, T^l, G^l, d)}
\prod_{t \in \mathcal{O}} p(t | w h_{m}^{1} \cdot \text{tag})^{\#(t, w h_{m}^{1} \cdot \text{tag}, W^l, T^l, d)} \prod_{a \in \mathcal{A}} p(a | h_{m}^{1})^{\#(a, h_{m}^{1}, W^l, T^l, d)}
\]

with local normalizations

\[
\sum_{w \in V} p(w | w_{n+1}^{1} h_{m}^{1} g) = 1
\sum_{t \in \mathcal{O}} p(t | w h_{m}^{1} \cdot \text{tag}) = 1
\sum_{a \in \mathcal{A}} p(a | h_{m}^{1}) = 1
\sum_{g \in G} p(g | d) = 1
\]
Efficient training algorithm?

- Inference and parameter estimation problems seem to be plausibly intractable
  - The added complexity due to $n$-gram and PLSA models in the composite $n$-gram/$m$-SLM/PLSA language model
  - Violation of tree structure in the topology of the underlying random field model
Generalized inside-outside algorithm

- The likelihood of a training corpus $\mathcal{D}$, a collection of documents, is

$$
\mathcal{L}(\mathcal{D}, p) = \prod_{d \in \mathcal{D}} \left( \prod_{l} \left( \sum_{G^l} \left( \sum_{T^l} P_p(W^l, T^l, G^l | d) \right) \right) \right)
$$

- Good news:
  - Following Jelinek’s ingenious definition of the inside and outside probabilities for SLM (Jelinek 2004)
  - $\exists$ an exact EM training algorithm, a generalized inside-outside algorithm (Wang et al. 2006)

- Bad news:
  - Computational complexity $O(L^6)$, $L$: sentence length,
  - Exact EM is not practical for a large scale corpus even with the use of pruning on charts (Jelinek 2004)
**$N$-best approximate EM algorithm**

The linear time $N$-best list approximate EM involves two steps:

1. **N-best list search**: For each sentence $W$ in document $d$, find $N$-best parse trees,

   \[ T_N^l = \arg \max_{T'_N^l} \left\{ \sum_{G^l} \sum_{T^l \in T'_N^l} P_p(W^l, T^l, G^l | d), ||T'_N^l|| = N \right\} \]

   where $T_N$ is a collection of $T_N^l$ for sentences over entire corpus $D$.

2. **EM update**: Perform one iteration (or several iterations) of EM algorithm to estimate model parameters that maximizes $N$-best-list likelihood of the training corpus $D$.

   \[ \tilde{L}(D, p, T_N) = \prod_{d \in D} \left( \prod_{l} \left( \sum_{G^l} \left( \sum_{T^l \in T_N^l} P_p(W^l, T^l, G^l | d) \right) \right) \right) \]

Iterate steps (1) and (2) until the convergence of the $N$-best-list likelihood.

Its convergence properties can be proven by Zangwill’s global convergence theorem (Zangwill 1969)
$N$-best approximate EM algorithm (continue)

1. $N$-best list search: A synchronous, multi-stack search strategy
   - A set of stacks storing partial parses of the most likely ones for a given prefix $W_k$, the less probable parses are purged
     - The hypotheses are ranked according to $\log(\sum_{G_k} P(W_k, T_k, G_k|d))$
     - Each stack contains partial parses constructed by the same number of constructor operations
   - The width of the pruning is controlled by:
     - maximum number of stack entries
     - log-probability threshold

$\Rightarrow$ Greedy best first search!
$N$-best approximate EM algorithm (continue)

2. EM update:

- **E-step:** Compute expected counts
  - For the WORD-PREDICTOR and the SEMANTIZER, use forward-backward recursive formulas that are similar to those in hidden Markov models to compute the expected counts.
  - For the TAGGER and the CONSTRUCTOR, the expected count of $tw h_{-m}^{-1}.t a g$ and $a h_{-m}^{-1}$ over parse $T^l$ of sentence $W^l$ in document $d$ is the real count appeared in parse tree $T^l$ times the posterior distribution $P_p(T^l|W^l, d)$.

- **M-step:** Assuming that the count ranges and the corresponding interpolation values for each order are kept fixed to their initial values, the only parameters to be re-estimated are the maximal order counts for each model component.
  - The interpolation scheme obtain a smooth probability estimate for each model component.
Linear interpolation for a Markov chain

\[
\begin{align*}
    p_n(x|z_{-n+1} \cdots z_{-1}) &= p_{n-1}(x|z_{-n+2} \cdots z_{-1}) + f_n(x|z_{-n+1} \cdots z_{-1})
    \\
    p_{n-1}(x|z_{-n+2} \cdots z_{-1}) &= p_{n-2}(x|z_{-n+3} \cdots z_{-1}) + f_{n-1}(x|z_{-n+2} \cdots z_{-1})
    \\
    &\vdots
    \\
    p_0(x) &= p_{-1}(x)
\end{align*}
\]

- Recursive mixing scheme of relative frequency estimates at different orders
  \[
  p_n(x|z_{-n+1} \cdots z_{-1}) = \lambda(z_{-n+1} \cdots z_{-1}) \cdot p_{n-1}(x|z_{-n+2} \cdots z_{-1}) + (1 - \lambda(z_{-n+1} \cdots z_{-1})) \cdot f_n(x|z_{-n+1} \cdots z_{-1})
  \]
  \[
  p_{-1}(x) = \text{uniform}(x)
  \]

- For TAGGER or CONSTRUCTOR, conditional context is a Markov chain
- For WORD-PREDICTOR, conditional context \(w_{-n+1} h_{-m} g\) is not a Markov chain.
  \[
  \implies \text{generalize Jelinek and Mercer's (1981) original recursive mixing scheme}
  \]

Shaojun Wang, Kno.e.sis Center, Department of Computer Science and Engineering, Wright State University  
http://www.cs.wright.edu/~swang~p.20/53
The lattice is formed by 3 Markov chains, \(w_{-2}w_{-1}, h_{-2}h_{-1}\) and \(g\). Each vertex is visited in a bottom up, back to front, right to left order.

- Each vertex is linearly interpolated with lower vertice and its frequency, such as,

\[
p(w|w_{-2}w_{-1}h_{-2}h_{-1}g) = \lambda_w(w_{-2}w_{-1}h_{-2}h_{-1}g) \cdot p(w|w_{-1}h_{-2}h_{-1}g) \\
+ \lambda_h(w_{-2}w_{-1}h_{-2}h_{-1}g) \cdot p(w|w_{-2}w_{-1}h_{-1}g) \\
+ \lambda_g(w_{-2}w_{-1}h_{-2}h_{-1}g) \cdot p(w|w_{-2}w_{-1}h_{-2}h_{-1}g) \\
+(1 - \lambda_w(w_{-2}w_{-1}h_{-2}h_{-1}g) - \lambda_h(w_{-2}w_{-1}h_{-2}h_{-1}g) - \lambda_g(w_{-2}w_{-1}h_{-2}h_{-1}g)) \cdot f(w|w_{-2}w_{-1}h_{-2}h_{-1}g)
\]

\(p(u)\): uniform distribution of \(u\)
Distributed architecture

Challenges when handling large scale corpus

- Data can’t be stored in a single machine
- Parameters can’t be stored in a single machine

Strategy: adopt server-client paradigm

- Clients store partitioned data and perform “Map” function: compute expected counts
- Servers store parameters (counts) for “Reduce” function
- Hash $w_k^{k-n+1}h^{-1}_m g_k$ by the last word $w_k$ and topic $g_k$
Follow-up EM algorithm to improve word prediction power

Use a large amount of the partial parse trees that are generated during the synchronous, multi-stack search strategy to re-estimate WORD-PREDICTOR to improve its predictive power.

- The language model probability assignment for the word at position $k+1$ in the input sentence can be written as:

$$P_p(w_{k+1}|W_k, d) = \sum_{h^{-1}_m \in T_k; T_k \in Z_k; g_k+1} p(w_{k+1}|w^k_{k-n+2}h^{-1}_m g_k+1)p(g_k+1|d)p(T_k|W_k, d)$$

where

$$p(T_k|W_k, d) = \frac{\sum_{G_k} p(W_k, T_k, G_k|d)}{\sum_{T_k \in Z_k} \sum_{G_k} p(W_k, T_k, G_k|d)}$$

to ensure a proper probability normalization over word strings $W_k$.

- $Z_k$ is the set of all parses present in our stacks at the current stage $k$.
- $G_k$ is the semantic node string up to $k$. 

Shaojun Wang, Kno.e.sis Center, Department of Computer Science and Engineering, Wright State University  http://www.cs.wright.edu/~swang – p.23/53
Follow-up EM algorithm to improve word prediction power (continue)

• The likelihood of a training corpus $\mathcal{D}$ that uses partial parse trees generated during the process of the synchronous, multi-stack search strategy can be written as

$$\tilde{L}(\mathcal{D}, p) = \prod_{d \in \mathcal{D}} \prod_{l} \left( \prod_{k} P_p(w_{k+1}^{l} | W_k^{l}, d) \right)$$

(1)

• Estimate $p(w_{k+1} | w_{k-n+2}^k h_{-m}^{-1} g_{k+1}^k)$ by maximizing Eqn. (1) using EM

  • **E-step:** gather expected joint counts $C(w_{k+1} w_{k-n+2}^k h_{-m}^{-1} g_{k+1}^k, d)$ of the WORD-PREDICTOR model with each count weighted by $p(w_{k+1} | w_{k-n+2}^k h_{-m}^{-1} g_{k+1}^k)p(g_{k+1} | d)p(T_k | W_k, d)$ normalized over $h_{-1}, \cdots, h_{-m} \in T_k \in Z_k$ and $g_{k+1}$

  • **M-step:** uses the same count smoothing technique as that described in the N-best list approximate EM

• Adopt a similar distributed architecture

• In fact, most improvements are from this algorithm
Use the model for testing

- A document of the test data is not contained in the original training corpus, the parameters \( p(g|d) \) have to be re-estimated by maximizing the probability of word subsequence seen so far, i.e., a pseudo-document \( \tilde{d}_k = (W_k, S) \), while holding the other parameters fixed.

- We use three methods:
  - One step online EM, \( \gamma = \frac{1}{|\tilde{d}_k| + 1} \)
  - Online EM with fixed learning rate, \( \gamma = 0.2 \)
  - Batch EM

\[
p(g|\tilde{d}_k) = \gamma \frac{\sum_{h_{-m} \in T_{k-1}} p(w_k | w_{k-n+1}^{k-1} h_{-m}, g)p(g|\tilde{d}_{k-1})P_p(T_{k-1}|W_{k-1}, \tilde{d}_{k-1})}{\sum_{h_{-m} \in T_{k-1}} p(w_k | w_{k-n+1}^{k-1} h_{-m}, g)p(g|\tilde{d}_{k-1})P_p(T_{k-1}|W_{k-1}, \tilde{d}_{k-1}) + (1 - \gamma)p(g|\tilde{d}_{k-1})}
\]

- Perplexity results are sensitive to these three methods and the initial values
- Online EM with fixed learning rate not only has the cheapest computational cost but also leads to highest perplexity reductions
Combining $n$-gram, $m$-SLM and PLSA through latent maximum entropy (LME)

$$
\max_{P(W,T,G,D)} \sum_{W,T,G,D} P(W,T,G,D) \log P(W,T,G,D) - \sum_{W,T,G,D} P(W,T,G,D) \log P(W,T,G,D)
$$

subject to the following nonlinear constraints

$$
\sum_{W,T,G,D} P(W,T,G,D) f(w_{n+1}^{-1} w_h^{-1} m g) = \sum_{W,D} \tilde{P}(W,D) \sum_{T,G} P(T,G|W,D) f(w_{n+1}^{-1} w_h^{-1} m g)
$$

$$
\sum_{W,T,G,D} P(W,T,G,D) f(t w h_{m}^{-1}.tag) = \sum_{W,D} \tilde{P}(W,D) \sum_{T,G} P(T,G|W,D) f(t w h_{m}^{-1}.tag)
$$

$$
\sum_{W,T,G,D} P(W,T,G,D) f(a h_{m}^{-1}) = \sum_{W,D} \tilde{P}(W,D) \sum_{T,G} P(T,G|W,D) f(a h_{m}^{-1})
$$

$$
\sum_{W,T,G,D} P(W,T,G,D) f(g d) = \sum_{W,D} \tilde{P}(W,D) \sum_{T,G} P(T,G|W,D) f(g d)
$$

where $\tilde{P}(W,D)$ denotes the empirical distribution of a sentence in a document over training corpus. This is a non-convex optimization problem due to the nonlinear constraints and there is no closed form solution.
Combining $n$-gram, $m$-SLM and PLSA under undirected MRF paradigm

- The likelihood of fully labeled sentence

$$P\lambda(W, T, G, D) = \frac{1}{Z\lambda} e^{<\Delta, \#f(W, T, G, D)>}$$

One global normalization factor $Z\lambda = \sum_{W, T, G, D} e^{<\Delta, \#f(W, T, G, D)>}$ to ensure a proper distribution.

- Maximum likelihood estimation (MLE)

$$\max_{\lambda} \sum_{W, D} \tilde{P}(W, D) \sum_{T, G} \log P\lambda(W, T, G, D)$$

- Stationary points of MLE $\equiv$ feasible solutions of LME (good)
- Computing $Z\lambda$ is intractable (bad)
- Computing right hand side feature expectations is tractable (good)
- Computing left hand side feature expectations is intractable (bad)
Related work

The most relevant work is by Khudanpur and Wu (2000), where they

- used SLM and a word clustering model to extract relevant grammatical and semantic features
- then integrated these features with $n$-grams by maximum conditional entropy approach
- thus they used undirected conditional MRFs

Comments:

- Our composite language model is a generative model, all features play important roles in EM iterations to allow maximal order events for WORD-PREDICTOR to appear;

  vs.

in Khudanpur and Wu (2000), the counts for all events are fixed after feature extraction from SLM and word clustering, which hinders the predictive power of WORD-PREDICTOR

- The training algorithm in Khudanpur and Wu (2000) is computationally expensive, mainly in feature expectation and normalization;

  vs.

ours is quite simple which is just expected relative frequency estimates with proper smoothing
Google’s famous results

- Use simple $n$-gram language model, the more the data, the better the result
  - Double data size $\implies 0.5\% \uparrow$ BLEU score
  - Overall 4.5\% BLEU score improvement from billion tokens to trillion tokens

- Why?

Shaojun Wang, Kno.e.sis Center, Department of Computer Science and Engineering, Wright State University
http://www.cs.wright.edu/~swang – p.29/53
Google’s famous results

- Use simple $n$-gram language model, the more the data, the better the result
- Why? Excellent $n$-gram hit ratio on unseen test data
Language modeling: a data rich and feature rich density estimation problem

**Glivenko–Cantelli theorem:**

\[ \tilde{p} \rightarrow \hat{p} \]

\[ p^n_M \rightarrow \hat{p}_n \]

**Pythagorean theorem**

\[ D(\hat{p}_n, p^n_M) = D(\hat{p}, \tilde{p}_n) + D(\tilde{p}_n, p^n_M) \]

\[ \text{approximation error} \quad \text{estimation error} \]

- \( \hat{p} \): true (but unknown) distribution; \( \hat{p}_n, n = 3, 4, 5, 6 \): information projection of \( \hat{p} \) to \( n \)-gram
- \( \tilde{p} \): empirical distribution, in particular, \( \tilde{p}_M \): empirical distribution for million words corpus, \( \tilde{p}_B \): empirical distribution for billion words corpus, \( \tilde{p}_T \): empirical distribution for trillion words corpus;
- \( p^n_M, n = 3, 4, 5 \): information projection of \( \tilde{p}_M \) to \( n \)-gram
Language modeling: a data rich and feature rich density estimation problem

- Pythagorean theorem breaks down, nevertheless there is always a trade-off between approximation error and estimation error
- The supply of more data needs to be matched by demand on the model side
Experiments set-up

- Three training corpora taken from the LDC English gigaword corpus.
  1. 44 million tokens
  2. 230 million tokens
  3. 1.3 billion tokens
- One independent test corpus
  1. 320k tokens
- Two check corpora to determine linear interpolation coefficients:
  1. 1.7 million tokens for training corpus 1
  2. 13.7 million tokens for training corpora 2 and 3
- The vocabulary sizes are:
  - word (also WORD-PREDICTOR operation) vocabulary: 60 k, open - all words outside the vocabulary are mapped to the $<$unk$>$ token, chosen from the most frequently occurred words in 44 millions tokens corpus
  - POS tag (also TAGGER operation) vocabulary: 40, closed
  - non-terminal tag vocabulary: 52, closed
  - CONSTRUCTOR operation vocabulary: 107, closed
## Data sets and model initialization

<table>
<thead>
<tr>
<th>Data set</th>
<th>Tokens</th>
<th>Start Date</th>
<th>End Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.3 billion tokens training corpus</td>
<td></td>
<td>19940512.0003</td>
<td>19961015.0568</td>
</tr>
<tr>
<td>afp</td>
<td>19940512.0003</td>
<td>19961015.0568</td>
<td></td>
</tr>
<tr>
<td>afw</td>
<td>19941111.0001</td>
<td>19960414.0652</td>
<td></td>
</tr>
<tr>
<td>nyt</td>
<td>19940701.0001</td>
<td>19950131.0483</td>
<td></td>
</tr>
<tr>
<td>nyt</td>
<td>19950401.0001</td>
<td>20040909.0063</td>
<td></td>
</tr>
<tr>
<td>xin</td>
<td>19970901.0001</td>
<td>20041125.0119</td>
<td></td>
</tr>
<tr>
<td>230 million tokens training corpus</td>
<td></td>
<td>19940622.0336</td>
<td>19961031.0797</td>
</tr>
<tr>
<td>afp</td>
<td>19940622.0336</td>
<td>19961031.0797</td>
<td></td>
</tr>
<tr>
<td>apw</td>
<td>19941111.0001</td>
<td>19960419.0765</td>
<td></td>
</tr>
<tr>
<td>nyt</td>
<td>19940701.0001</td>
<td>19941130.0405</td>
<td></td>
</tr>
<tr>
<td>44 million tokens training corpus</td>
<td></td>
<td>19940601.0001</td>
<td>19950721.0137</td>
</tr>
<tr>
<td>afp</td>
<td>19940601.0001</td>
<td>19950721.0137</td>
<td></td>
</tr>
<tr>
<td>nyt</td>
<td>19950201.0001</td>
<td>19950331.0494</td>
<td></td>
</tr>
<tr>
<td>13.7 million tokens check corpus</td>
<td></td>
<td>19950201.0001</td>
<td>19950331.0494</td>
</tr>
<tr>
<td>nyt</td>
<td>19950201.0001</td>
<td>19950331.0494</td>
<td></td>
</tr>
<tr>
<td>1.7 million tokens check corpus</td>
<td></td>
<td>19940512.0003</td>
<td>19940531.0197</td>
</tr>
<tr>
<td>afp</td>
<td>19940512.0003</td>
<td>19940531.0197</td>
<td></td>
</tr>
<tr>
<td>354 k tokens test corpus</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cna</td>
<td>20041101.0006</td>
<td>20041217.0009</td>
<td></td>
</tr>
</tbody>
</table>

- Each model component of WORD-PREDICTOR, TAGGER, and CONSTRUCTOR is initialized from a set of parsed sentences.
- We use the "openNLP" software, a maxent parser trained by Upenn treebank, to parse all sentences in 44 and 230 million tokens corpora, and a portion of 1.3 billion tokens corpus, then use them to initialize model parameters.
# Baseline $n$-grams

- Perplexity results using linear interpolation and Kneser-Ney (KN) smoothing

<table>
<thead>
<tr>
<th></th>
<th>44 M</th>
<th>LINEAR</th>
<th>KN</th>
<th>230 M</th>
<th>LINEAR</th>
<th>KN</th>
<th>1.3 B</th>
<th>LINEAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n=3$</td>
<td>262</td>
<td>244</td>
<td>$n=3$</td>
<td>217</td>
<td>195</td>
<td>$n=3$</td>
<td>161</td>
<td></td>
</tr>
<tr>
<td>$n=4$</td>
<td>258</td>
<td>235</td>
<td>$n=4$</td>
<td>200</td>
<td>183</td>
<td>$n=4$</td>
<td>141</td>
<td></td>
</tr>
<tr>
<td>$n=5$</td>
<td>260</td>
<td>235</td>
<td>$n=5$</td>
<td>201</td>
<td>183</td>
<td>$n=5$</td>
<td>138</td>
<td></td>
</tr>
</tbody>
</table>

- Statistics about the number of types of $n$-grams

<table>
<thead>
<tr>
<th></th>
<th>$n=3$</th>
<th>$n=4$</th>
<th>$n=5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>44 M</td>
<td>14,302,355</td>
<td>23,833,023</td>
<td>29,068,173</td>
</tr>
<tr>
<td>230 M</td>
<td>51,115,539</td>
<td>94,617,433</td>
<td>120,978,281</td>
</tr>
<tr>
<td>1.3 B</td>
<td>224,767,319</td>
<td>481,645,099</td>
<td>660,599,586</td>
</tr>
</tbody>
</table>
**$m$-SLMs**

- SLMs’ perplexity results using linear interpolation

<table>
<thead>
<tr>
<th></th>
<th>44 M</th>
<th>230 M</th>
<th>1.3 B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LINEAR</td>
<td>LINEAR</td>
<td>LINEAR</td>
</tr>
<tr>
<td>$m=2$</td>
<td>279</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$m=3$</td>
<td></td>
<td>190</td>
<td></td>
</tr>
<tr>
<td>$m=4$</td>
<td></td>
<td></td>
<td>137</td>
</tr>
</tbody>
</table>

- Counts of the types in the predictor of the $m$-SLMs.

<table>
<thead>
<tr>
<th></th>
<th>$m=2$</th>
<th>$m=3$</th>
<th>$m=4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>44 M</td>
<td>189,002,525</td>
<td>269,685,833</td>
<td>318,174,025</td>
</tr>
<tr>
<td>230 M</td>
<td>267,507,672</td>
<td>1,154,020,346</td>
<td>1,417,977,184</td>
</tr>
<tr>
<td>1.3 B</td>
<td>946,683,807</td>
<td>1,342,323,444</td>
<td>1,849,882,215</td>
</tr>
</tbody>
</table>

For 230 million and 1.3 billion tokens corpora, fractional expected counts that are less than a threshold are pruned to significantly reduce the number of $m$-SLM predictor’s types by 70%.
**n-gram/PLSA**

- Fix the size of topics in PLSA to be 200
- Perplexity (ppl) results and time consumption when different numbers of most likely topics are kept for each document in PLSA

<table>
<thead>
<tr>
<th>CORPUS</th>
<th>(n)</th>
<th># OF TOPICS</th>
<th>PPL</th>
<th>TIME (HOURS)</th>
<th># OF SERVERS</th>
<th># OF CLIENTS</th>
<th># OF TYPES OF (w_{-n+1}^{-1}w_g)</th>
</tr>
</thead>
<tbody>
<tr>
<td>44M</td>
<td>3</td>
<td>5</td>
<td>196</td>
<td>0.5</td>
<td>40</td>
<td>100</td>
<td>120.1M</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>10</td>
<td>194</td>
<td>1.0</td>
<td>40</td>
<td>100</td>
<td>218.6M</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>20</td>
<td>190</td>
<td>2.7</td>
<td>80</td>
<td>100</td>
<td>537.8M</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>50</td>
<td>189</td>
<td>6.3</td>
<td>80</td>
<td>100</td>
<td>1.123B</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>100</td>
<td>189</td>
<td>11.2</td>
<td>80</td>
<td>100</td>
<td>1.616B</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>200</td>
<td>188</td>
<td>19.3</td>
<td>80</td>
<td>100</td>
<td>2.280B</td>
</tr>
<tr>
<td>230M</td>
<td>4</td>
<td>5</td>
<td>146</td>
<td>25.6</td>
<td>280</td>
<td>100</td>
<td>0.681B</td>
</tr>
<tr>
<td>1.3B</td>
<td>5</td>
<td>2</td>
<td>111</td>
<td>26.5</td>
<td>400</td>
<td>100</td>
<td>1.790B</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>5</td>
<td>102</td>
<td>75.0</td>
<td>400</td>
<td>100</td>
<td>4.391B</td>
</tr>
</tbody>
</table>

- Unpruned 5 topics in general account for 70% probability in \(p(g|d)\)
- Prune 200 topics to 5 in the rest of experiments
### Perplexity results for various LMs

<table>
<thead>
<tr>
<th>language model</th>
<th>44 M \ n=3, m=2</th>
<th>230 M \ n=4, m=3</th>
<th>1.3 B \ n=5, m=4</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline n-gram (linear)</td>
<td>262</td>
<td>200</td>
<td>138</td>
</tr>
<tr>
<td>n-gram (KN)</td>
<td>244 \ 6.9%</td>
<td>183 \ 8.5%</td>
<td>--</td>
</tr>
<tr>
<td>m-SLM</td>
<td>279 \ -6.5%</td>
<td>190 \ 5.0%</td>
<td>137 \ 0.0%</td>
</tr>
<tr>
<td>PLSA</td>
<td>825 \ -214.9%</td>
<td>812 \ -306.0%</td>
<td>773 \ -460.0%</td>
</tr>
<tr>
<td>n-gram+m-SLM</td>
<td>247 \ 5.7%</td>
<td>184 \ 8.0%</td>
<td>129 \ 6.5%</td>
</tr>
<tr>
<td>n-gram+PLSA</td>
<td>235 \ 10.3%</td>
<td>179 \ 10.5%</td>
<td>128 \ 7.2%</td>
</tr>
<tr>
<td>n-gram+m-SLM+PLSA</td>
<td>222 \ 15.3%</td>
<td>175 \ 12.5%</td>
<td>123 \ 10.9%</td>
</tr>
<tr>
<td>n-gram/m-SLM</td>
<td>243 \ 7.3%</td>
<td>171 \ 14.5%</td>
<td>(125) \ 9.4%</td>
</tr>
<tr>
<td>n-gram/PLSA</td>
<td>196 \ 25.2%</td>
<td>146 \ 27.0%</td>
<td>102 \ 26.1%</td>
</tr>
<tr>
<td>m-SLM/PLSA</td>
<td>198 \ 24.4%</td>
<td>140 \ 30.0%</td>
<td>(103) \ 25.4%</td>
</tr>
<tr>
<td>n-gram/PLSA+m-SLM/PLSA</td>
<td>183 \ 30.2%</td>
<td>140 \ 30.0%</td>
<td>(93) \ 32.6%</td>
</tr>
<tr>
<td>n-gram/m-SLM+m-SLM/PLSA</td>
<td>183 \ 30.2%</td>
<td>139 \ 30.5%</td>
<td>(94) \ 31.9%</td>
</tr>
<tr>
<td>n-gram/m-SLM+n-gram/PLSA</td>
<td>184 \ 29.8%</td>
<td>137 \ 31.5%</td>
<td>(91) \ 34.1%</td>
</tr>
<tr>
<td>n-gram/m-SLM+n-gram/PLSA +m-SLM/PLSA</td>
<td>180 \ 31.3%</td>
<td>130 \ 35.0%</td>
<td>--</td>
</tr>
<tr>
<td>n-gram/m-SLM/PLSA</td>
<td>176 \ 32.8%</td>
<td>--</td>
<td>--</td>
</tr>
</tbody>
</table>

+ denotes linear combination, / denotes composite model, topics are pruned from 200 to 5
Model size is a big issue!

- Counts of the types in predictor of 5-gram/PLSA, 5-gram/2-SLM (or 2-gram/4-SLM), and 4-SLM/PLSA when trained on 1.3b corpus.

<table>
<thead>
<tr>
<th>COMPOSITE MODEL</th>
<th>TYPES OF</th>
<th># OF TYPES</th>
<th># OF SERVERS</th>
<th># OF CLIENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>5-GRAM/PLSA</td>
<td>$w_{-4}w_g$</td>
<td>4.39 B</td>
<td>400</td>
<td>100</td>
</tr>
<tr>
<td>5-GRAM/2-SLM</td>
<td>$w_{-4}w_{h-2}$</td>
<td>2.01 B</td>
<td>240</td>
<td>100</td>
</tr>
<tr>
<td>OR</td>
<td>$w_{-1}w_{h-4}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-GRAM/4-SLM</td>
<td>$w_{h-4}g$</td>
<td>4.88 B</td>
<td>400</td>
<td>100</td>
</tr>
</tbody>
</table>

Heavy pruning: purge fractional expected counts that are less than a threshold to reduce the number of predictor’s types by 85%.
Using different testing methods

• Perplexity results

<table>
<thead>
<tr>
<th>language model</th>
<th>44 M n=3, m=2</th>
<th>230 M n=4, m=3</th>
<th>1.3 B n=5, m=4</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline n-gram (linear)</td>
<td>262</td>
<td>200</td>
<td>138</td>
</tr>
<tr>
<td>n-gram/PLSA</td>
<td>202 22.9%</td>
<td>150 25.0%</td>
<td>107 22.5%</td>
</tr>
<tr>
<td>n-gram/m-SLM+n-gram/PLSA</td>
<td>192 26.7%</td>
<td>142 29.0%</td>
<td>(97) 29.1%</td>
</tr>
<tr>
<td>n-gram/PLSA</td>
<td>196 25.2%</td>
<td>146 27.0%</td>
<td>102 26.1%</td>
</tr>
<tr>
<td>n-gram/m-SLM+n-gram/PLSA</td>
<td>184 29.8%</td>
<td>137 31.5%</td>
<td>(91) 34.1%</td>
</tr>
<tr>
<td>n-gram/PLSA</td>
<td>201 23.3%</td>
<td>148 26.0%</td>
<td>104 24.6%</td>
</tr>
<tr>
<td>n-gram/m-SLM+n-gram/PLSA</td>
<td>189 27.9%</td>
<td>140 30.0%</td>
<td>(92) 33.3%</td>
</tr>
</tbody>
</table>

1. one step online EM
2. online EM with fixed learning rate $\gamma = 0.2$
3. batch EM
Why composite language model powerful?

- Composite model predicts word based on a hierarchically organized committee and the relative frequency estimate it extracts.
- Linear combination is restricted to a simplex.
Why composite language model powerful?

- Composite model encodes valuable relative frequency estimates explicitly or implicitly in training corpus
- The weights used in simple linear combination are context independent, thus more restricted
- The WORD-PREDICTOR of the composite model oversees all available information to make the most powerful prediction.

<table>
<thead>
<tr>
<th>Model Configuration</th>
<th>ppl</th>
</tr>
</thead>
<tbody>
<tr>
<td>trigram</td>
<td>176</td>
</tr>
<tr>
<td>trigram/2-SLM</td>
<td>143</td>
</tr>
<tr>
<td>trigram/PLSA</td>
<td>262</td>
</tr>
<tr>
<td>trigram/2-SLM/PLSA</td>
<td>247</td>
</tr>
<tr>
<td>trigram/PLSA+2-SLM</td>
<td>196</td>
</tr>
<tr>
<td>trigram/PLSA/2-SLM</td>
<td>235</td>
</tr>
<tr>
<td>trigram/2-SLM+2-SLM</td>
<td>222</td>
</tr>
<tr>
<td>trigram/2-SLM+2-SLM/PLSA</td>
<td>183</td>
</tr>
<tr>
<td>trigram/2-SLM+2-SLM+PLSA</td>
<td>180</td>
</tr>
</tbody>
</table>

Shaojun Wang, Kno.e.sis Center, Department of Computer Science and Engineering, Wright State University  
An example of sentence probability

- Training corpus: 1.3 billion tokens

- Test document: <XIN_ENG_20041126_0168.story>
  
  a. 5-gram: perplexity = 97
  
  b. 5-gram+PLSA: perplexity = 93
  
  c. 5-gram+4-SLM+PLSA: perplexity = 83
  
  d. 5-gram/PLSA: perplexity = 71
  
  e. 5-gram/PLSA+4-SLM/PLSA: perplexity = 64

- The first four sentences

  <S> cpc initiates education campaign to strengthen members’ wavering convictions </S> <S> by zhao lei </S> <S> beijing nov. ’nmbh xinhua the communist party of china cpc has decided to launch a mass internal educational campaign from january next year to prevent its members from wavering in their convictions </S> <S> the decision aiming to keep the nature of the party members intact was made at the meeting of the political bureau of the cpc central committee on this oct. ’nmbh the cpc ’s top power organ </S> · · · · ·
An example of sentence probability (continue)

<table>
<thead>
<tr>
<th></th>
<th>the</th>
<th>decision</th>
<th>aiming</th>
<th>to</th>
<th>keep</th>
<th>the</th>
<th>nature</th>
<th>of</th>
<th>the</th>
</tr>
</thead>
<tbody>
<tr>
<td>a.</td>
<td>−2.00317</td>
<td>−5.99654</td>
<td>−14.9793</td>
<td>−0.852055</td>
<td>−4.68269</td>
<td>−1.49193</td>
<td>−9.84554</td>
<td>−0.526566</td>
<td>−0.671103</td>
</tr>
<tr>
<td>b.</td>
<td>−2.05502</td>
<td>−6.08843</td>
<td>−13.2655</td>
<td>−0.950885</td>
<td>−4.78594</td>
<td>−1.56474</td>
<td>−9.81423</td>
<td>−0.6258</td>
<td>−0.761926</td>
</tr>
<tr>
<td>c.</td>
<td>−2.05416</td>
<td>−6.07556</td>
<td>−13.3486</td>
<td>−0.871798</td>
<td>−4.69523</td>
<td>−1.57311</td>
<td>−9.99731</td>
<td>−0.897362</td>
<td>−0.829652</td>
</tr>
<tr>
<td>d.</td>
<td>−1.72696</td>
<td>−5.65359</td>
<td>−14.2013</td>
<td>−0.99068</td>
<td>−5.43248</td>
<td>−1.65002</td>
<td>−7.6</td>
<td>−0.612751</td>
<td>−0.755122</td>
</tr>
<tr>
<td>e.</td>
<td>−1.80167</td>
<td>−5.73861</td>
<td>−14.5548</td>
<td>−0.893825</td>
<td>−5.05692</td>
<td>−1.60568</td>
<td>−7.92909</td>
<td>−0.751419</td>
<td>−0.755122</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>party</th>
<th>members</th>
<th>intact</th>
<th>was</th>
<th>made</th>
<th>at</th>
<th>the</th>
<th>meeting</th>
<th>of</th>
</tr>
</thead>
<tbody>
<tr>
<td>a.</td>
<td>−6.52337</td>
<td>−5.93013</td>
<td>−14.992</td>
<td>−5.5802</td>
<td>−5.91863</td>
<td>−3.47798</td>
<td>−1.0155</td>
<td>−3.77026</td>
<td>−3.11882</td>
</tr>
<tr>
<td>c.</td>
<td>−6.48696</td>
<td>−5.81026</td>
<td>−8.11845</td>
<td>−3.04638</td>
<td>−2.21191</td>
<td>−2.80501</td>
<td>−1.12155</td>
<td>−3.85156</td>
<td>−2.3551</td>
</tr>
<tr>
<td>d.</td>
<td>−3.46383</td>
<td>−5.03999</td>
<td>−15.242</td>
<td>−5.27819</td>
<td>−4.73655</td>
<td>−3.03394</td>
<td>−0.69443</td>
<td>−3.23709</td>
<td>−3.40986</td>
</tr>
<tr>
<td>e.</td>
<td>−3.80075</td>
<td>−5.16911</td>
<td>−8.52597</td>
<td>−3.38567</td>
<td>−2.54778</td>
<td>−2.74127</td>
<td>−0.790644</td>
<td>−3.36195</td>
<td>−2.64652</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>the</th>
<th>political bureau</th>
<th>of</th>
<th>the</th>
<th>cpc</th>
<th>central committee</th>
</tr>
</thead>
<tbody>
<tr>
<td>a.</td>
<td>−0.619712</td>
<td>−5.91994</td>
<td>−1.36559</td>
<td>−0.17816</td>
<td>−0.127888</td>
<td>−1.55966</td>
</tr>
<tr>
<td>b.</td>
<td>−0.710967</td>
<td>−5.96757</td>
<td>−1.47083</td>
<td>−0.278998</td>
<td>−0.313708</td>
<td>−1.66454</td>
</tr>
<tr>
<td>c.</td>
<td>−0.636643</td>
<td>−6.0839</td>
<td>−1.43513</td>
<td>−0.6519</td>
<td>−0.634246</td>
<td>−2.10113</td>
</tr>
<tr>
<td>d.</td>
<td>−0.475928</td>
<td>−4.13345</td>
<td>−0.527685</td>
<td>−0.226433</td>
<td>−0.204276</td>
<td>−1.55903</td>
</tr>
<tr>
<td>e.</td>
<td>−0.475442</td>
<td>−4.43649</td>
<td>−0.702968</td>
<td>−0.427385</td>
<td>−0.388118</td>
<td>−1.79781</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>on</th>
<th>this</th>
<th>oct.</th>
<th>‘nmbr</th>
<th>the</th>
<th>cpc</th>
<th>’s</th>
<th>top</th>
<th>power</th>
</tr>
</thead>
<tbody>
<tr>
<td>a.</td>
<td>−4.33953</td>
<td>−7.02792</td>
<td>−10.7495</td>
<td>−0.0380615</td>
<td>−3.87067</td>
<td>−9.93617</td>
<td>−3.54366</td>
<td>−4.19702</td>
<td>−7.6261</td>
</tr>
<tr>
<td>b.</td>
<td>−4.37441</td>
<td>−6.88172</td>
<td>−10.6397</td>
<td>−0.141938</td>
<td>−3.65821</td>
<td>−8.81816</td>
<td>−3.60823</td>
<td>−4.29886</td>
<td>−7.64586</td>
</tr>
<tr>
<td>c.</td>
<td>−3.57338</td>
<td>−6.86285</td>
<td>−10.9656</td>
<td>−0.131813</td>
<td>−3.8662</td>
<td>−8.85551</td>
<td>−3.42688</td>
<td>−4.28615</td>
<td>−7.82392</td>
</tr>
<tr>
<td>d.</td>
<td>−4.61674</td>
<td>−6.49064</td>
<td>−13.0595</td>
<td>−0.255452</td>
<td>−3.73302</td>
<td>−5.55244</td>
<td>−3.60481</td>
<td>−3.97708</td>
<td>−7.85289</td>
</tr>
<tr>
<td>e.</td>
<td>−3.85647</td>
<td>−6.61406</td>
<td>−12.5666</td>
<td>−0.178075</td>
<td>−3.92356</td>
<td>−5.90511</td>
<td>−3.46416</td>
<td>−4.03158</td>
<td>−7.91198</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>organ</th>
<th>&lt;/s&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>a.</td>
<td>−5.97561</td>
<td>−2.62716</td>
</tr>
<tr>
<td>b.</td>
<td>−6.08022</td>
<td>−2.67444</td>
</tr>
<tr>
<td>c.</td>
<td>−6.01553</td>
<td>−2.65078</td>
</tr>
</tbody>
</table>
Why composite language model powerful?

• Statistics when $n$-gram is the same as SLM’s WORD-PREDICTOR headword

<table>
<thead>
<tr>
<th>CORPUS</th>
<th>$w_{-2}^{-1} = h_{-2}^{-1}$</th>
<th>$w_{-3}^{-1} = h_{-3}^{-1}$</th>
<th>$w_{-4}^{-1} = h_{-4}^{-1}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>44 M</td>
<td>57%</td>
<td>46%</td>
<td>38%</td>
</tr>
<tr>
<td>230 M</td>
<td>59%</td>
<td>46%</td>
<td>38%</td>
</tr>
<tr>
<td>1.3 B</td>
<td>55%</td>
<td>48%</td>
<td>43%</td>
</tr>
</tbody>
</table>

• $n$-gram alone is not viable to achieve similar affect even using web scale data
• The directed MRF paradigm effectively synergizes $n$-gram, $m$-SLM, and PLSA in a complementary, supplementary, and coherent way to form a powerful language model for word prediction of natural language.
BLEU score results for re-ranking $N$-best list in machine translation

- Same 1000-best list used by Zhang et al. 2006.
- Generated on 919 sentences from the MT03 Chinese-English evaluation set by Hiero (Chiang 2005). Its decoder uses a trigram with modified Kneser-Ney smoothing that trained on a 200 million words corpus.
- Each translation has 11 features and language model is one of them.
- We substitute 5-gram/2-SLM+2-gram/4-SLM+5-gram/PLSA language model trained by 1.3 billion word corpus and use MERT (Och 2003) to optimize the BLEU score and re-rank the 1000-best list. 10-fold cross-validation BLEU score results are

<table>
<thead>
<tr>
<th>System model</th>
<th>mean (%)</th>
<th>variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>31.75</td>
<td>0.0431</td>
</tr>
<tr>
<td>5-gram</td>
<td>32.53</td>
<td>0.0484</td>
</tr>
<tr>
<td>(5-gram/SLM)</td>
<td>32.87</td>
<td>0.0502</td>
</tr>
<tr>
<td>5-gram/PLSA$^{1}$</td>
<td>33.01</td>
<td>0.0412</td>
</tr>
<tr>
<td>(5-gram/SLM/PLSA$^{1}$)</td>
<td>33.32</td>
<td>0.0431</td>
</tr>
</tbody>
</table>
Translations’ “readability” results

- Similar to the study conducted by Charniak et al. 2003
- The translations are sorted into 4 groups: good/bad syntax crossed with good/bad meaning by human judges

<table>
<thead>
<tr>
<th>System model</th>
<th>P</th>
<th>S</th>
<th>G</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>95</td>
<td>398</td>
<td>20</td>
<td>406</td>
</tr>
<tr>
<td>5−gram</td>
<td>122</td>
<td>406</td>
<td>24</td>
<td>367</td>
</tr>
<tr>
<td>(5−gram/SLM/PLSA(^1))</td>
<td>153</td>
<td>428</td>
<td>33</td>
<td>305</td>
</tr>
</tbody>
</table>

P: perfect  S: only semantically correct  G: only grammatically correct  W: wrong
Is textbook right?

On page 482, “We said earlier that statistical parsers can take advantage of longer-distance information than $n$-grams, which suggests that they might do a better job at language modeling/word prediction. It turns out that if we have a very large amount of training data, a 4-gram or 5-gram is nonetheless still the best way to do language modeling.”

About data

- “There is no data like more data” (Mercer at Arden House, 1985 from Jelinek 2004)
- “More data is more important than better algorithms” (Brill’s opinion from Jelinek 2004)
- But this doesn’t mean
  - Simple algorithms are better than sophisticated algorithms
- They should be compared at the same ground: using the same size of training data
- Training a composite model on trillion tokens corpus is feasible, affordable, and cheap in the era of cloud computing
Better MT evaluation metric?

- BLEU: $n$-gram based MT evaluation metric
- Closer agreement may be possible by incorporating syntactic structure and semantic information into the BLEU score evaluation
  - semantically similar words like “insure” and “ensure” in the example of BLEU paper (Papineni et al. 02) should be substituted in the formula
  - assign a weight to measure the goodness of syntactic structure

This modification will lead to a better metric and such information can be provided by our composite language models
Conclusion

• We have implemented a composite language model that
  • integrates n-gram, SLM and PLSA under the directed MRF paradigm
  • trained using corpora up to a billion tokens
  • stored on a supercomputer with up to 1000 processors
• The large scale distributed composite language model
  • gives significant perplexity reduction over n-grams
  • achieves significantly better translation quality measured by the BLEU score
  • “readability” of translations when applied to the task of reranking the N-best list in statistical machine translation
• As far as we know, this is the first work
  • to build a complex large scale distributed language model with a principled approach that
    • simultaneously exploits syntactic, semantic and lexical regularities
    • more powerful than n-grams trained on a very large corpus
Future work

- Integrate more advanced topic language models such as LDA, CTM, DTM (Blei et al. 2003, 2006, 2007)
- Resort to hierarchical non-parametric Bayesian model (Teh 2006) for smoothing fractional counts due to latent variables in Kneser-Ney’s sense in a principled manner
- Use Blue waters that has 300k cores to handle trillion tokens corpus
- Put it into
  - a phrased-based machine translation decoder (Koehn et al. 2003) that produces a lattice of alternative translations/transcriptions
  - a syntax-based decoder (Chiang 2007) that produces a forest of alternatives (such integration would, in the exact case, reside in an extremely difficult complexity class, probably PSPACE-complete)

We expect:
- 45% ~ 50% perplexity reduction
- 3% ~ 5% BLEU score improvement $\Rightarrow$ 64 ~ 1024 times training data
Acknowledgements

- We would like to dedicate this work to the memory of Fred Jelinek, who passed away while we were finalizing this manuscript
- Work done jointly with Wright State University students Ming Tan, Wenli Zhou and Lei Zheng
- This research is supported by NSF under grant IIS:RI-small 0812483, a Google research award and AFOSR under grant FA9550-10-1-0335
- This work is supported in part by an allocation of computing time from the Ohio Supercomputer Center
- We would like to thank
  - Ciprian Chelba for providing the SLM code, answering many questions regarding SLM and consulting on various aspects of the work
  - Ying Zhang and Philip Resnik for providing the 1000-best list from Hiero for re-ranking in machine translation
  - Peng Xu for suggesting to look at conditional probability of a word given its document history to make the perplexity result much convincing.