The third meeting two days between foreign ministers of the Egyptian and Sudanese.

Ref: Gabriel Egyptian Premier Dr. Atef Ebeid announced today Tuesday that the Sudanese Minister, Mustapha Osman Ismail, has held talks today, Tuesday, with his Sudanese counterpart, Mustafa Osman Ismail, for the third time in two days.

He delivered a message from Mubarak during their meeting on Thursday.

He added to give a message from Mubarak at the meeting on Thursday in Cairo had decided to reconcile the Egyptian-Libyan initiative with the Government Authority for Development (GAD) in order to end the civil war.

The minister added that the Sudanese letter on Egypt's role in efforts to reach national reconciliation in Sudan.

Once had Ismail announced Sunday that his country supports efforts to achieve integration between African and Arab initiatives to end the civil war in Sudan more than 16 years old.

It was announced by the National Democratic grouping which comprises the opposition northern Sudanese southerners insurgents after their meeting last Thursday in Cairo had decided to reconcile the Egyptian-Libyan initiative for Development (GAD) to end the civil war.

Ref: that session should therefore be devoted exclusively to bilateral relations, the Egyptian commercial - SAF.

The third meeting two days between foreign ministers of the Egyptian and Sudanese

Ref: Ismail had declared on Sunday that his government supports the efforts made to achieve agreement between the Arab initiative and the African one, in order to put an end to the civil war taking place in Sudan for more than 16 years.

Ref: Once had Ismail announced Sunday that his country supports efforts to achieve integration between African and Arab initiatives to end the civil war in Sudan more than 16 years old.

Ref: After its meeting last Thursday in Cairo, the Democratic National Gathering, which includes the Sudanese southern opposition and the southern rebels, declared that it has decided to work in order to reconcile the Egyptian-Libyan initiative and that of the Government Authority for Development (GAD), to put an end to the civil war in Sudan.

Ref: Third meeting in two days between Egyptian and Sudanese Ministers of Foreign Affairs

Ref: Third meeting in two days between Egyptian and Sudanese Ministers of Foreign Affairs

Ref: That session should therefore be devoted exclusively to bilateral relations, the Egyptian commercial - SAF.

Ref: that # 5.14855 } @Phrase { por lo tanto , # therefore , # 6.44248 } @Phrase { ese # # 3.0102

@Phrase { exclusivamente a # deal exclusively with # 3.5631 } @Phrase { debe dedicarse # should # 11.8434 } @Phrase ( # 6.61431 } @Phrase { las negociaciones . # the negotiations . # 3.0102

Ref: Por lo tanto , ese per-o-odo de sesiones debe dedicarse exclusivamente a las negociaciones .

Ref: that session should therefore be devoted exclusively to negotiations .

Ref: that session should deal exclusively with the negotiations .

How it is done:

@Phrase ( por lo tanto , # therefore , # 6.44248 ) @Phrase ( ese # # 3.0102

@Phrase { exclusivamente a # deal exclusively with # 3.5631 ) @Phrase { las negociaciones . # the negotiations . # 3.0102

Ref: por lo tanto , ese per-o-odo de sesiones debe dedicarse exclusivamente a las negociaciones .

Ref: that session should therefore be devoted exclusively to negotiations .

Ref: that session should deal exclusively with the negotiations .
SMT Approach

- Advantages, Disadvantages
- Tasks in SMT

Statistical Approach

- Using statistical models
  - Create many alternatives, called hypotheses
  - Give a score to each hypothesis
  - Select the best -> search
- Advantages
  - Avoid hard decisions
  - Sometimes, optimality can be guaranteed
  - Speed can be traded with quality, no all-or-nothing
  - It works better!
- Disadvantages
  - Difficulties in handling structurally rich models, mathematically and computationally (but that's also true for non-statistical systems)
  - Need data to train the model parameters

Machine Translation Approaches

- Grammar-based
  - Interlingua-based
  - Transfer-based
- Direct
  - Example-based
  - Statistical

- Example-based
- Statistical

Statistical versus Grammar-Based

- Often statistical and grammar-based MT are seen as alternatives, even opposing approaches – wrong!!!
- Dichotomies are:
  - Use probabilities – everything is equally likely (in between: heuristics)
  - Rich (deep) structure – no or only flat structure
- Both dimensions are continuous
- Examples
  - EBMT: flat structure and heuristics
  - SMT: flat structure and probabilities
  - XFER: deep structure and heuristics
- Goal: structurally rich probabilistic models

Tasks in SMT

- Modelling
  - build statistical models which capture characteristic features of translation equivalences and of the target language
- Training
  - train translation model on bilingual corpus, train language model on monolingual corpus
- Decoding
  - find best translation for new sentences according to models
- Evaluation
  - Subjective evaluation: fluency, adequacy
  - Automatic evaluation: WER, BLEU, etc.
- And all the nitty-gritty stuff:
  - Data cleaning
  - Parameter tuning

Tasks in SMT

- Modelling
  - build statistical models which capture characteristic features of translation equivalences and of the target language
- Training
  - train translation model on bilingual corpus, train language model on monolingual corpus
- Decoding
  - find best translation for new sentences according to models
- Evaluation
  - Subjective evaluation: fluency, adequacy
  - Automatic evaluation: WER, BLEU, etc.
- And all the nitty-gritty stuff:
  - Data cleaning
  - Parameter tuning
Preprocessing

- Normalizing text
  - Separating punctuation
  - True casing, often all lower case (May (month) -> may?)
- Cleaning
  - Typos
  - Sentence alignment errors
  - Check sentence length
  - Check alignment perplexity (requires already initial training done)
  - Sentences with many non-words
- Tagging
  - Numbers
  - Names

Quality Control: MT Evaluation

- Simple metrics: WER and PER
- Most fashionable these days: BLEU and NIST
- Newcomers

Simple Evaluation Approaches

- WER (Edit distance)
  - Taken from speech recognition
  - Very harsh, as it does not allow for reordering
  - Multi-references (Alshawi)
  - Edit distance with word moves (Alshawi)
- PER (Position independence error rate)
  - Unigram precision
  - Word order not considered at all

N-gram based Evaluation Metrics

ViewPoint:

- The closer a machine translation is to a professional human translation, the better it is.
  - A numerical "translation closeness" metric
  - A corpus of good quality human reference translations
- Not perfect! There are subtleties and stylistic variations that are better appreciated by humans than machines.

Numerical “Translation Closeness”

Two components:
- Modified N-gram Precision
  - Percentage of n-grams in hypothesis that can be matched in references
  - Short n-gram matches capture "adequacy"
  - Longer n-gram matches capture "fluency"
  - A reference n-gram should be considered "exhausted" after being matched by an n-gram in hypothesis
- Sentence Brevity Penalty
  - Penalizes short hypothesis

Bleu Metric – Precisions

- Precisions of n-grams:
  - Where:
    - $N$ using 1,2,3...n-gram
    - $W_i$ weights for n-gram precision


$$\text{Precision} = \exp \left( \sum_{n} w_i \log p_i \right)$$

- A word-weighted average rather than a sentence-weighted average

$$p_i = \frac{\sum \text{Count}_{\text{source}}(n \cdot \text{gram})}{\sum \text{Count}(n \cdot \text{gram})}$$
Bleu Metric – Length Penalty

- Brevity Penalty
  \[ BP = \begin{cases} 
  1 - \frac{1}{e^{1 - \frac{c}{r}}} & \text{if } c > r \\
  0 & \text{if } c \leq r 
  \end{cases} \]

- \( c \): total length of hypothesis
- \( r \): effective reference length (Summing the best match lengths for each hyp. sent.)
- BP calculated over the entire test set, rather than for each sentence

- Note: NIST’s implementation of the Bleu metric uses the shortest reference translations to calculate the length penalty.

---

Bleu Metric

- Formula:
  \[ BLEU = BP \cdot \exp\left( \sum_{n=1}^{N} w_n \log p_n \right) \]

- Generally: \( N = 4 \)
- Uniform weights for \( n \) gram: \( w_n = \frac{1}{N} \)
- Score range: 0–1
- More references: higher score? Yet same system ranking order.

---

NIST Score – Precisions

- Modification to Bleu's precision metric
- NIST’s arguments:
  - Bleu uses geometric mean of co-occurrence over \( N \); alternatively use arithmetic average of \( n \)-gram counts
  - Wants to weigh more heavily those \( n \)-grams that are more informative

- Define:
  \[ \text{Info}(w_{i_1}, ..., w_{i_N}) = \log \left( \frac{\text{the \# \ of \ occurrences \ of \ } w_{i_1}, ..., w_{i_N}}{\text{the \# \ of \ occurrences \ of \ } w_{i_1}, w_{i_N}} \right) \]

- Modified \( n \)-gram precision by NIST:
  \[ \text{Precision} = \sqrt{\frac{\sum_{w_{i_1}, ..., w_{i_N}} \text{Info}(w_{i_1}, ..., w_{i_N})}{\sum_{w_{i_1}, ..., w_{i_N}} \text{Info}(w_{i_1}, w_{i_N})}} \]

---

NIST Score – Length Penalty

- Formula:
  \[ \exp(\beta \log[\min\left(\frac{L_{hyp}}{L_{ref}}\right)]) \]

- Beta is chosen to make the penalty=0.5 when \( L_{hyp} = \frac{2}{3} L_{ref} \)
- \( \beta = -4.22 \)
- \( L_{ref} \) is the averaged length of all references
- Length Penalty calculated over the whole test set
NIST Score – Length Penalty

- Less penalty than Bleu for sentences slightly shorter than references
- Dropping un-translated words results in higher score than replacing them by LM

NIST Score

\[ \text{Score} = \sum_{k=1}^{N} \left( \frac{\sum_{i=1}^{L(k)} \log(\exp]\left[ \frac{1}{N(k)} \right])}{L(k)} \right) \cdot \exp(\beta \log^2[\min(\frac{1}{N(k)}, 1)]) \]

- \( N=5 \)
- Score range: no theoretical range, typically \([0, 20]\)

Newcomers

- MT evaluation is the hot topic in MT research
- Heated debates at workshops
- “Everybody hates Bleu”
- Alternative metrics by the dozen

- Metrics which have caught on:
  - TER: translation error rate, essentially WER with reordering
  - Used for HTER: measuring amount of post-editing with TER
  - Meteor
    - Designed by Alon Lavie
    - Match translation against each reference with reordering
    - Use n-gram matches and fragmentation score
    - Can use stemming and synonyms from word net to make matching more robust
    - Higher correlation to human evaluation, esp. on a per sentence level

Joy’s Master Eval Script

```
perl ~joy/Eval/EvaluateXlat.pl
```

MT Evaluation Master Script: n Usage:
```
perl5 EvaluateXlat.pl [-l] [-b] testSuiteId sysId hypPlainFile
```
- \(-l\) for evaluate lowercase only
- \(-b\) for using Bleu v1.04 test normalization method

- testSuiteId:
  - C_dryrun Chinese Dryrun Dec 2001, 993 sentences
  - C_june02 Chinese June 2002, 378 sentences
  - C_map03 Chinese May 2003, 919 sentences
  - C_map03_tag Chinese May 2003, 919 sentences with numbers tagged
  - C_map04 Chinese MT04 data, 1789 sentences
  - C_map04_10Docs Chinese May 2004, 10 sentences
  - C_map04_20Docs Chinese May 2004, 191 sentences
  - C_MT05 Chinese MT05, 1082 sentences

Summary

- Automatic evaluation: cheap and fast
- Correlates reasonably well with human evaluation
  - And therefore reasonably well with each other
  - Still: systems can be tuned towards different metrics
- Different Metrics:
  - WER, PER, Bleu, NIST
  - Currently most popular: Bleu and NIST
  - New or modification of existing metrics are proposed on a (nearly) daily basis

The Basis: Word Alignment

- Word Alignment
- Mixture Model
  - IBM2
  - IBM1
- Training IBM1
SMT - Principle

- Translate a 'French' string \( f_j = f_1, f_2, ..., f_J \) into an 'English' string \( e_i = e_1, e_2, ..., e_I \).
- Bayes' decision rule for translation:
  \[
  e_i^* = \arg \max_i \left( \Pr(e_i | f_j) \cdot \Pr(f_j) \right)
  \]

- Why this inversion of the translation direction?
  - Decomposition of dependencies: makes modeling easier
  - Cooperation of two knowledge sources for final decision
- Note 1: Noisy channel model
- Note 2: Alternative is direct translation with log-linear model combination

Alignment Example

Observations:
- Often 1-1
- Often monotone
- Some 1-to-many
- Some 1-to-nothing
- Not always clear-cut

Mixture Model

- Interpretation as mixture model by direct decomposition
  \[
  \Pr(f_j^* | e_i) = \frac{\prod_j \Pr(f_j | J, e_i)}{\prod_j \Pr(f_j | J, e_i)} = \frac{\prod_j \Pr(f_j | J, e_i)}{\prod_j \Pr(f_j | J, e_i)}
  \]

- Again, simplifying model assumptions applied

IBM1 Model

- Assume uniform probability for position alignment
  \[
  p(i, j, J, J) = \frac{1}{J}
  \]

- Alignment probability
  \[
  \Pr(f_j^* | e_i) = \frac{\prod_j p(i, j, J, J \parallel p(f_j | e_i))}{\prod_j p(f_j | e_i)} = \frac{\prod_j p(i, j, J, J \parallel p(f_j | e_i))}{\prod_j p(f_j | e_i)}
  \]

- In training: only collect counts for word pairs

Training for IBM1 Model – Pseudo Code

# Accumulation (over corpus)
For each sentence pair
  For each source position \( j \)
    Sum += \( p(f_j | e_i) \)
  For each target position \( i \)
  Count(f_j, e_i) += \( p(f_j | e_i) / \text{Sum} \)

# Re-estimate probabilities (over count table)
For each target word \( e \)
  Sum = 0.0
  For each source word \( f \)
    Sum += Count(f, e)
    \( p(f | e) = \frac{\text{Count}(f, e)}{\text{Sum}} \)

# Repeat for several iterations

Notation

- Source language
  - \( f \): source (French) word
  - \( J \): length of source sentence
  - \( j \): position in source sentence (target position)
  - \( f_j = f_1, f_2, ..., f_J \): source sentence

- Target language
  - \( e \): target (English) word
  - \( I \): length of target sentence
  - \( i \): position in target sentence (source position)
  - \( e_i = e_1, e_2, ..., e_I \): target sentence

- Alignment: relation mapping source to target positions
  - \( i = a_j \): position \( i \) of \( e \) which is aligned to \( j \)
  - \( a_j = a_1, a_2, ..., a_J \): whole alignment
Word Alignment Models

- IBM1 – lexical probabilities only
- IBM2 – lexicon plus absolute position
- IBM3 – plus fertilities
- IBM4 – inverted relative position alignment
- IBM5 – non-deficient version of model 4
- HMM – lexicon plus relative position
- BBr – Bilingual Bracketing, lexical probabilities plus reordering via parallel segmentation
- Syntactical alignment models


GIZA++ Alignment Toolkit

- All standard alignment models (IBM1 ... IBM5, HMM) are implemented in GIZA++
- This toolkit was started (as GIZA) at John Hopkins University workshop 1998
- Extended and improved by Franz Josef Och
- Now used by many groups: de facto standard

Summary

- SMT approach: noisy channel
- Simplest Model: IBM1
  - word-word probabilities
  - Simple training with Expectation-Maximization
- Cascade of word alignment models
- GIZA++ toolkit implements IBM1...IBM5 plus HMM

Next Step: Phrase Alignment

- Why Phrase Alignment?
- Phrase Pair Extraction as Sentence Splitting

Alignment Example

- One Chinese word aligned to multi-word English phrase
- In lexicon individual entries with 'the', 'development', 'of'
- Difficult to generate from words
  - Main translation 'development'
  - Test if insertions of 'the' and 'of' improves LM probability
  - Easier to generate if we have phrase pairs available

Why Phrase to Phrase Translation

- Captures n x m alignments
  - one to one: 使役 the post
  - one to many: 单语动
  - many to one: 使役 活动
- Encapsulates context
  - 'the' -> 'a'
  - 'of' -> 'a'
- Local reordering
  - 兴起: rise
  - 经历: experience
- Compensate: 兴起 (rise) to rise
**How to get Phrase Translation**

- Typically: Train word alignment model and extract phrase-to-phrase translations from Viterbi path
  - IBM model 4 alignment
  - HMM alignment
  - Bilingual Bracketing
- Phrase translation models not using Viterbi alignment:
  - Integrated segmentation and alignment (developed by Joy)
  - Phrase Pair Extraction via full Sentence Alignment
- Notes:
  - Often better results when training target to source for extraction of phrase translations due to asymmetry of alignment models
  - Phrases are not fully integrated into the alignment model, they are extracted only after training is completed – how to assign probabilities?

**Word Alignment Matrix**

- Alignment probabilities according to lexicon

**Viterbi Path**

- Calculate Viterbi path (i.e. path with highest probability or best score)

**Phrases from Viterbi Path**

- Read off source phrase – target phrase pairs

**Dealing with Asymmetry**

- Word alignment models are asymmetric; Viterbi path has:
  - multiple source words – one target word alignments
  - but no one source word – multiple target words alignments
- Train alignment model also in reverse direction, i.e. target -> source
- Using both Viterbi paths:
  - Simple: extract phrases from both directions and merge tables
  - ‘Merge’ Viterbi paths and extract phrase pairs according to resulting pattern

**Combine Viterbi Paths**

- F->E
- E->F
- Intersect
- Fill Gaps
**Combine Viterbi Paths**
- Intersections: high precision, but low recall
- Union: lower precision, but higher recall
- Refined: start from intersection and fill gaps using points in union
- Different heuristics have been used
  - See e.g. papers by Och and Koehn
- Quality of phrase translation pairs depends on:
  - Quality of word alignment
  - Quality of combination of Viterbi paths
  - Actually, Koehn reports results showing that the combination heuristics is more important than the quality of the initial word alignment

**Non-Viterbi Phrase Alignment**
- Search translation for one source phrase

**Non-Viterbi Phrase Alignment (2)**
- What we would like to find

**Phrase Alignment**
- Search for optimal boundaries

**Phrase-Pair Extraction as Sentence Splitting**
- Calculate modified IBM1 word alignment: don’t sum over words in ‘forbidden’ (grey) areas
- Select target phrase boundaries which maximize sentence alignment probability
  - Modify boundaries $i_1$ and $i_2$
  - Calculate sentence alignment
  - Take best
Phrase Alignment

Search for optimal boundaries

Phrase Alignment

Search for optimal boundaries

Phrase Alignment

Search for optimal boundaries

Phrase Alignment

Search for optimal boundaries

Phrase Alignment

Search for optimal boundaries
Phrase Alignment

- Search for optimal boundaries

\[
\begin{align*}
\text{Phrase Alignment – Best Result} & \\
\text{Optimal target phrase} & \\
\end{align*}
\]

\[
\begin{align*}
\text{Phrase Alignment – Use n-best} & \\
\text{Use all translation candidates with scores close to the best one} & \\
\end{align*}
\]

Phrase Extraction via Sentence Splitting

- Calculate modified IBM1 word alignment: don’t sum over words in ‘forbidden’ areas

\[
\begin{align*}
\text{Looking from Both Sides} & \\
\text{Calculate both } & \\
\text{Interpolate the probabilities from both direction} & \\
\text{Find the target phrase boundary } & \\
\text{Interpolation factor } c \text{ can be tuned on development test set} & \\
\text{We have experimented with additional features:} & \\
\text{Only 'inside' alignment probability} & \\
\text{Using word fertility information for phrase length model} & \\
\text{Retraining lexicon from phrase pairs (Jaedy, MT-lab project)} & \\
\end{align*}
\]

\[
\begin{align*}
\text{Formula} & \\
\text{Select target phrase boundaries which maximize sentence} & \\
\text{alignment probability } & \\
\text{Looking from Both Sides} & \\
\text{Calculate both } & \\
\text{Interpolate the probabilities from both direction} & \\
\text{Find the target phrase boundary } & \\
\text{Interpolation factor } c \text{ can be tuned on development test set} & \\
\text{We have experimented with additional features:} & \\
\text{Only 'inside' alignment probability} & \\
\text{Using word fertility information for phrase length model} & \\
\text{Retraining lexicon from phrase pairs (Jaedy, MT-lab project)} & \\
\end{align*}
\]
Number of (Source) Phrases

- Small corpus: 40k sentences with 400k words

<table>
<thead>
<tr>
<th>Seen&gt;5</th>
<th>Seen&gt;2</th>
<th>Seen&gt;1</th>
<th>Count</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9,026</td>
<td>5796</td>
<td>4516</td>
<td>2998</td>
</tr>
<tr>
<td>2</td>
<td>82,289</td>
<td>92,351</td>
<td>30,382</td>
<td>18,096</td>
</tr>
<tr>
<td>3</td>
<td>173,817</td>
<td>346,132</td>
<td>35,743</td>
<td>17,982</td>
</tr>
<tr>
<td>4</td>
<td>210,496</td>
<td>476,628</td>
<td>23,837</td>
<td>9,643</td>
</tr>
<tr>
<td>5</td>
<td>208,583</td>
<td>685,211</td>
<td>14,046</td>
<td>4,870</td>
</tr>
<tr>
<td>6-10</td>
<td>735,989</td>
<td>1,421,200</td>
<td>16,617</td>
<td>4,291</td>
</tr>
</tbody>
</table>

- Number of phrases quickly exceeds number of words in corpus
- Numbers are for source phrases only; each phrase typically has multiple translations (factor 5 – 20)

Length of Phrases

- Do we find any long phrases in our training data?
  - Yes!
    - E.g. TIDES 2004 Chinese testing data
      - matched against 4.6 million word FBIS training data
      - matched against 60 million word UN training data

| Phrase Length vs. Translation Quality |

- Increasing the length of phrases improves the translation scores

<table>
<thead>
<tr>
<th>Max. Phrase Lenth</th>
<th>Modified n-gram Precision (%)</th>
<th>NIST5</th>
<th>BLEU4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>59,67</td>
<td>29,66</td>
<td>5,91</td>
</tr>
<tr>
<td>2</td>
<td>64,57</td>
<td>25,06</td>
<td>9,33</td>
</tr>
<tr>
<td>5</td>
<td>65,82</td>
<td>26,91</td>
<td>10,95</td>
</tr>
<tr>
<td>10</td>
<td>66,04</td>
<td>27,48</td>
<td>11,47</td>
</tr>
<tr>
<td>19</td>
<td>66,04</td>
<td>27,47</td>
<td>11,45</td>
</tr>
</tbody>
</table>

- Longer n-gram matches are not used to calculate standard BLEU/NIST scores:
  - Bleu up to 4-grams
  - Nist up to 5-grams
  - Translations are more fluent for human to read.

Just-In-Time Phrase Pair Extraction

- Given a test sentence: find occurrences of all substrings (n-grams) in the bilingual corpus
- Use suffix array to index source part of corpus
  - Space efficient (for each word – one pointer)
  - Search requires binary search
  - Can find n-grams up to any n (restricted within sentence boundaries)
- Extract phrase-translation pairs
  - Find phrase alignment based on word alignment
  - Can use Viterbi alignment (could be pre-calculated)
  - Or use new phrase alignment approach
- Mixed approach: high frequency phrases aligned offline, low frequency phrases aligned online

Indexing a Corpus using a Suffix Array

- For alignment the sentence numbers are needed:
  - Insert <sos> markers into the corpus
  - Insert sentence numbers into the corpus
Searching a String using a Suffix Array

- Search "the economy"
  1. step: search for range of "the" => \[l_1, r_1\]
  2. step: search for range of "the economy" within \[l_1, r_1\] => \[l_2, r_2\]

Locating all Sub-Strings of a Sentence

- Testing sentence: "growth is the essence of the economy"

Summary

- Phrase-based translation for higher translation quality
- Phrase alignment based on underlying word alignment
- Different phrase alignment approaches
  - From Viterbi paths
  - Integrated segmentation and alignment
  - Phrase alignment as optimizing sentence splitting
- Looking from both side to cope with asymmetry of word alignment models
- Phrase translation table is huge: Restrict phrase to short and/or high frequency phrases
- Online phrase alignment
  - Use suffix array to index all phrases in corpus
  - Binary search
  - Efficient way to find all phrase in a sentence

Dealing with the Target Side: Language Model

- N-gram LMs
- Suffix array LM
- Dealing with very large corpora

N-gram Language Models

- N-gram LMs have long tradition in speech recognition
- Also the LM used in pretty much all SMT systems
- Sentence probability is product over n-gram probabilities
  \[ p(c_i') = \prod_{i=1}^{n} \Pr(c_i | c_{i-1} \ldots c_n) \approx \prod_{i=1}^{n} \Pr(c_i | c_{i-m} \ldots c_n) \]
  
  - Typically, 3-grams are used, with larger memory computers also 4-grams and 5-grams
  
  - Cut-offs for low frequency n-grams
  
  - Filter LM for given vocabulary (1 LM per sentence)

- LM toolkits
  - SRI LM Toolkit – most popular
  - CMU/Cambridge LM Toolkit

Suffix Array Language Model

- Suffix array allows to find n-grams up to any length efficiently
- Pointers to a give n-gram are contiguous -> range = n-gram count
- Pointers to a given n-gram are embedded inside the range of the pointers to the (n-1)-gram, i.e. we get counts for calculating the relative frequencies
- Simple smoothing by interpolating the 1-gram, ... n-gram probabilities
- LM can be restricted to using n-grams only up to specified length
- LM build from a 1 billion word corpus takes 9 GB
    - But no cut-offs necessary
    - History length unrestricted, i.e. varigram LM
    - We have seen improvements up to 6-gram LMs
Distributed LM

- Want to use large LMs, which do not fit into memory of one workstation?
- Use client-server architecture
  - Each server owns part of the LM data
  - Use suffix array to find n-gram and get n-gram counts
  - Efficiency has been greatly improved using batch-mode communication between client and server.
- Client collects n-gram occurrence counts and calculates probabilities
- Simple smoothing: linear interpolation with uniform weights

Client-Server Architecture for LM

- Suffix array for 50M words needs ~450M RAM
- We use 2.9 billion words from Gigaword corpus …

Putting it together: Decoding

- Decoding strategies
  - Two step decoding
    - Generation of translation lattice
      - Using phrase tables
      - Using online phrase extraction
    - Best path search
    - Word reordering
  - Specific Issues
    - Recombination of hypotheses
    - Pruning
    - N-best list generation

Decoder

- Decoding strategies
  - Two step decoding
    - Generation of translation lattice
      - Using transducer
    - Using online phrase extraction
    - Best path search
    - Word reordering
  - Specific Issues
    - Recombination of hypotheses
    - Pruning
    - N-best list generation

Decoding Strategies

- Sequential construction of target sentence
  - Extend partial translation by words which are translations of words in the source sentence
  - Language model can be applied immediately
  - Mechanism to ensure proper coverage of source sentence required
- Left – right over source sentence
  - Find translations for sequences of words
  - Construct translation lattice
  - Apply language model and select best path
  - Extensions to allow for word reordering
- Growing coverage
  - Start with sections with high confidence
  - Expand these regions
  - How to incorporate the language model? Connections at both ends!

Decoder: The Knowledge Sources

- Lexical information
  - Statistical lexicon
  - Manual lexicon
  - Phrase translations
  - Named entities
    - All lexical information stored as transducers or extracted on the fly
- Language model: standard 3-gram
- Position alignment model for word reordering
- Sentence length model
The Decoder: Two Level Approach

- Build translation lattice
  - Run left-right over test sentence
  - Search for matching phrases between source sentence and phrases extracted from corpus
  - For each translation, insert edges into the lattice
- First best search
  - Run left-right over lattice
  - Apply trigram language model
  - Combine translation model score and language model score
  - Recombine and prune hypotheses
  - At sentence end: add sentence length model score
  - Trace back best hypothesis (or n-best hypotheses)

Building Translation Lattice

Sentence: "ich komme morgen zu dir"
Reference: "I will come to you tomorrow"

- Search in corpus for phrases and their translations
- Insert edges into the lattice

Translation Lattice from Bil. Grammar

Phrase Table as Tree-Transducer

Hierarchical Transducers

- Transducers with labels allowed:
  - @DAY # Monday
  - @DAY # Tuesday
  - @DATE # on @DAY
  - @DATE # @DAY , der @NUM @MONTH # @DAY
  - @DATE # der @NUM @MONTH # @MONTH , the @NUM
  - Apply transducers sequentially (right order required)
  - Propagate translations:
    - @DATE ( am @DAY ( Monag # Monday ) # on Tuesday)
  - Mechanism to allow incorporation of simple bilingual grammars

Searching for Best Path

- Hypotheses describe a partial translation
- Coverage information, Back-trace information, Score
- Expand hypothesis over uncovered position
Reordering Strategies

- All permutations
  - Any re-ordering possible
  - Complexity of traveling salesman -> only possible for very short sentences
- Small jumps ahead – filling in the gaps pretty soon
  - Only local word reordering
  - Implemented in current decoder
- Leaving small number of gaps – fill in at any time
  - Allows for global but limited reordering
  - Similar decoding complexity – exponential in number of gaps
  - IBM-style reordering (described in IBM patent)
- Merging neighboring regions with swap – no gaps at all
  - Allows for global reordering
  - Complexity lower than 1, but higher than 2 and 3

Dealing with Word Order – Coverage Info

- Need to know which source words have already been translated
- Don’t want to miss some words
- Don’t want to translate words twice
- Can compare hypotheses which cover the same words
- Use Coverage vector to store this information
  - Essentially a bit vector
  - For ‘small jumps ahead’: position of first gap plus short bit vector
  - For ‘small number of gaps’: array of positions of uncovered words
  - For ‘merging neighboring regions’: left and right position

Word Reordering

- Word and phrase reordering within a given window
  - From first un-translated source word next k positions
  - Window length 1: monotone decoding
  - Restrict total number of reordering (typically 3 per 10 words)
- Simple ‘Jump’ model
  - One reordering typically includes two jumps
  - Jump distance \( \Delta \) depends on gap and also on phrase length
distance measured from center of phrase to center of phrase
  - Currently simply Gaussian distribution: \( p(\Delta) = \exp(-\frac{\Delta^2}{2}) \)
  - Scaling factor to weight jump model wrt TM and LM
  - Can (and should be) replaced by lexicalized jump model

Jumping ahead in the Lattice

- Hypothesis describes a partial translation
  - Coverage information, back-trace information, Score
  - Expand hypothesis over uncovered position

Coverage Information

- We need to keep track of words already translated
- Use bit vector 100110... = words 1, 4, and 5 translated
- For long sentences long bit vectors translated
  but only limited reordering allowed
  therefore:
    Coverage = ( first untranslated word, bit vector)
i.e. 111100110... -> (4, 00110...)
Pruning
- Pruning
  - Larger search space requires stronger pruning
  - Remove hypotheses which are worse than best hypothesis by a factor k
- Flexible pruning, using combinations of
  - Number of covered positions
  - Covered positions
  - Language model state (history)
  - Number of generated words
- Use one or two features less as used for recombination
- Sometimes still too many hypotheses within the beam - need to add histogram pruning

Search Space
- Example: sentence with 48 words
- Full search for using coverage and language model state
- Av. Expanded is for entire test set, i.e. 4991 words

<table>
<thead>
<tr>
<th>R</th>
<th>Expanded</th>
<th>Collisions</th>
<th>Av. Expanded</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>183,806</td>
<td>0</td>
<td>6,467</td>
</tr>
<tr>
<td>1</td>
<td>1,834,212</td>
<td>588,293</td>
<td>72,343</td>
</tr>
<tr>
<td>2</td>
<td>8,589,221</td>
<td>3,479,193</td>
<td>326,470</td>
</tr>
<tr>
<td>3</td>
<td>33,851,161</td>
<td>12,127,175</td>
<td>1,230,020</td>
</tr>
</tbody>
</table>

- More reordering -> more collisions
- Growth of search space is counteracted by recombination of hypotheses

Effect of Pruning
- Average number of expanded hypotheses and NIST scores for different recombination and pruning combinations and different beam widths
- Test Set: Arabic DevTest (203 sentences)

<table>
<thead>
<tr>
<th>Beam Width</th>
<th>1</th>
<th>2</th>
<th>5</th>
<th>10</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Av. Hyps exp.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C : c</td>
<td>825</td>
<td>899</td>
<td>1,132</td>
<td>1,492</td>
<td>1,801</td>
</tr>
<tr>
<td>C : c : c</td>
<td>1,174</td>
<td>1,857</td>
<td>6,213</td>
<td>30,293</td>
<td>214,402</td>
</tr>
<tr>
<td>C : C</td>
<td>2,792</td>
<td>4,248</td>
<td>12,921</td>
<td>53,228</td>
<td>287,278</td>
</tr>
<tr>
<td>NIST</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C : c</td>
<td>8.18</td>
<td>8.81</td>
<td>8.21</td>
<td>8.22</td>
<td>8.27</td>
</tr>
<tr>
<td>C : c : c</td>
<td>8.41</td>
<td>8.62</td>
<td>8.68</td>
<td>8.95</td>
<td>8.98</td>
</tr>
<tr>
<td>C : C</td>
<td>8.47</td>
<td>8.68</td>
<td>8.85</td>
<td>8.98</td>
<td>8.98</td>
</tr>
</tbody>
</table>

Sentence Length Model
- Different language have different level of ‘wordiness’
- Histogram over source sentence length – target sentence length shows that distribution is rather flat - \( P( J | I ) \) is not very helpful
- Very simple sentence length model: the more – the better
  - i.e. give bonus for each word (not a probabilistic model)
  - Balances shortening effect of LM
  - Can be applied immediately, as absolute length is not important
- However: this is insensitive to what’s in the sentence
  - Optimize length of translations for entire test set, not each sentence
  - Long sentences are made longer to cover for sentences which are too short
N-Best List Generation

- Benefit:
  - Required for optimizing model scaling factors
  - Rescoring
  - For translation with pivot language: L1 -> L2 -> L3
- We have n-best translations at sentence end
- But: Hypotheses are recombined -> many good translations don’t reach the sentence end
- Recover those translations

N-Best Rescoring

- Generate n-best list
- Use different TM and/or LM to rescore each translation -> reordering of translations, i.e. different best translation
- Different TMs:
  - Use IBM1 lexicon for entire translation
  - Use HMM-FB and IBM4 lexicons
  - Forced alignment with HMM alignment model
- Different LMs:
  - Link grammar (already done offline by Bing and Peter – too slow)
  - Other syntax-based LMs, e.g. Charniak’s parser?

Tuning the SMT System

- We use different models in SMT system
  - Models have simplifications
  - Trained on different amounts of data
- => Different levels of reliability
- => Give different weight to different Models
  \[ Q = c_1 Q_1 + c_2 Q_2 + \ldots + c_n Q_n \]
- Find optimal scaling factors \( c_1 \ldots c_n \)
- Optimal means: Highest score for chosen evaluation metric

Automatic Tuning

- Many algorithms to find (near) optimal solutions available
  - Simplex
  - Maximum entropy
  - Minimum error training
  - Minimum Bayes risk training
  - Genetic algorithm
- Note: models are not improved, only their combination
- Large number of fully translations required
  => still problematic when decoding is slow

Automatic Tuning on N-best List

- Generate n-best lists,
  e.g. for each of 500 source sentences 1000 translations
- Loop
  - Changing scaling factors results in re-ranking the n-best lists
  - Evaluate new 1-best translations
- Apply any of the standard optimization techniques
- Advantage: much faster
- Can pre-calculate the counts (e.g. n-gram matches) for each translation to speed up evaluation
- For Bleu or NIST metric with global length penalty do local hill climbing for each individual n-best list

Minimum Error Training

- For each scaling factor we have: \( Q = c_i Q_i + Q_{\text{best}} \)
- For different values different hyps have lowest score
- Different hyps lead to different MT eval scores
Summary

- Decoder
  - Generating translation lattice
  - Finding best path
  - Limited word reordering
- Generation of N-best list
  - Esp used for tuning system
  - May also be used for downstream NLP modules
- Tuning of System
  - Find optimal set of scaling factors
  - Done on n-best list for speed
  - Direct minimization of MT eval metric