

Machine Translation with Inferred Stochastic Finite-State Transducers

Francisco Casacuberta and Enrique Vidal

MTRG - June 2, 2005

Presented by Sarah Schwarm

Intro: Formal Transducers

- Many applications of transducers in pattern recognition and language processing:
 - Tasks that can be formulated as translation from one sequence of symbols to another sequence of (different) symbols
 - E.g. ASR, handwritten-character recognition, language translation

Finite-State Transducers

- FSTs are a particular class of transducers, corresponding to **rational transductions** (Berstel 1979)
- Much more computationally tractable than other (more powerful) transducers
- FSTs can be learned automatically from data, although few techniques exist
- This paper: leverage techniques for inferring regular grammars from data to infer rational transducers

FST Definition

- $T = \langle \Sigma, \Delta, Q, q_0, F, \delta \rangle$
- Σ = set of source symbols
- Δ = set of target symbols
- Q = set of states
- $q_0 \in Q$ = start state
- $F \subseteq Q$ = set of final states
- $\delta \subseteq Q \times \Sigma \times \Delta \times Q$ = transitions (current state, input symbol, output symbol, next state)

More Definitions

- **Translation form** ϕ of length I is a sequence of valid transitions

$$\varphi = (q_0, s_1, t_1, q_1)(q_1, s_2, t_2, q_2)\dots(q_{I-1}, s_I, t_I, q_I)$$

- **Translation pair:** $(\mathbf{s}, \mathbf{t}) \in \Sigma^* \times \Delta^*$ is a translation pair if there is a ϕ of length $I = |\mathbf{s}|$ and $\mathbf{t} = t_1 t_2 \dots t_I$
- $d(\mathbf{s}, \mathbf{t})$ is the set of translation forms associated with the pair (\mathbf{s}, \mathbf{t})
- **Rational translation:** set of all translation pairs of an FST T

Comparison to Regular/ Finite-State Grammars

- Grammar doesn't have target symbols, so transitions are $Q \times \Sigma \times Q$
- “Translation form” / “derivation”
- “Rational translation” / “regular language”
- Theorem 1: (see p. 206)

$T \subseteq \Sigma^* \times \Delta^*$ is a rational translation iff there is a corresponding regular language L over alphabet Γ and two morphisms h_Σ and h_Δ that map strings in L to Σ and Δ

Statistical Translation Using FSTs (1/2)

- For a source string s , find the target string t that maximizes $\Pr(s,t)$ - see eqn 2 p. 207
- $\Pr(s,t)$ can be modeled by a stochastic FST

Stochastic FSTs

$$T_P = \langle \Sigma, \Delta, Q, q_0, p, f \rangle$$

- $p: Q \times \Sigma \times \Delta^* \times Q \rightarrow [0,1]$

Probability of each transition

- $f: Q \rightarrow [0,1]$

Probability that each state is a final state

- Must satisfy the following for each q

$$f(q) + \text{sum of prob. of all transitions out of } q = 1$$

Stochastic FSTs for Translation

- Set of transitions of FST T is the set of tuples (q, s, t, q') in T_P with probability > 0
- Set of final states is the set of states with nonzero $f(q)$
- Prob. of a translation pair (\mathbf{s}, \mathbf{t}) is the sum of the probs. of all translation forms of (\mathbf{s}, \mathbf{t})
- Prob. of a translation form ϕ is the product of the probabilities of all the transitions in ϕ
- See eqn 3, p. 207

Stochastic FSTs for Translation, more details (1/2)

- Ignore transducers with “useless states”, i.e. for every state in T there is a path to a final state
- Assume $P_{T_P}(\mathbf{s}, \mathbf{t}) = 0$ if there is no translation form for (\mathbf{s}, \mathbf{t}) in T
- Then P_{T_P} is the **stochastic translation** defined by T_P , i.e. a joint distribution on $\Sigma^* \times \Delta^*$
- See eqn 4, p 207

Stochastic FSTs for Translation, more details (2/2)

- Embedded source (P_i) and target (P_o) stochastic regular languages
- Obtained by dropping the target or source symbols, respectively
- Theorem 2 (extension of theorem 1 to stochastic FSTs - see p. 208)

Search with Stochastic FSTs

- Finding the optimal translation \mathbf{t} is computationally difficult
- Use an approximation of the probability of a translation pair (Viterbi score)
- Now we can efficiently calculate approximate translations using dynamic programming

Inferring FSTs

- Inferring a stochastic FST from a parallel corpus A of string pairs $(\mathbf{s}, \mathbf{t}) \in \Sigma^* \times \Delta^*$
 1. Transform each pair (\mathbf{s}, \mathbf{t}) into a string z from an extended alphabet Γ yielding $S \subset \Gamma^*$
 2. Infer a stochastic regular grammar G from S
 3. Transform the Γ -symbols of the grammar rules back into source/target pairs from $\Sigma^* \times \Delta^*$

Steps 1 and 3

- Step 1 done by “labeling function”

$$\mathcal{L}: \Sigma^* \times \Delta^* \rightarrow \Gamma^*$$

- Step 3 done by “inverse labeling function”

$$\Lambda(\mathcal{L}(A)) = A$$

- From theorems 1 and 2, $\Lambda(\cdot)$ consists of two morphisms h_Σ and h_Δ such that for $z \in \Gamma^*$,

$$\Lambda(z) = (h_\Sigma(z), h_\Delta(z))$$

- h_Σ and h_Δ are determined by \mathcal{L}

Requirements for \mathcal{L}

- \mathcal{L} must transform a parallel corpus into a string corpus, capturing potentially complicated correspondences between source and target words
- Preliminary non-stochastic version (Vidal 1989) which did not adequately cope with source-target correspondences
- Can improve by using statistical alignments - **Grammatical Inference and Alignments for Transducer Inference (GIATI)**
- An **alignment** of a translation pair (\mathbf{s}, \mathbf{t}) is a function that maps positions in \mathbf{t} to positions in \mathbf{s} (Brown et al 1993)

Step 1: Transforming Training Pairs to Strings

- Build a string of extended symbols (Γ) for each training pair (s, t) using its statistical alignment
- Basic idea: assign each word from t to the corresponding word form s using alignment a
- Problem: don't want to violate word order in t
- “Solution”: join words according to a unless word order of t is violated (in which case, join with the next word)
 - Avoids “reordering” the target string
 - Lower degree of nonmonotonicity (sort of)
 - It's wrong (linguistically) but they claim it works okay for limited-domain languages

Step 2: Grammar Inference

- Assume that we infer **n-grams with final states** (particular case of stochastic regular grammar)
- Infer n-gram models and compute probabilities from counts in the training set of extended strings
 - Use back-off smoothing
- Represent the n-gram model as a stochastic finite-state automaton

Step 3: Transforming a Stochastic Regular Grammar into a Stochastic FST

- Conversion of FSA (grammar) to FST is straightforward
- Might need to add (trivial) new states for back-off
- Inverse transformation $\Lambda(z)$ is based on:

if $(a, b_1 b_2 \dots b_k) \in \Gamma$ with $a \in \Sigma$ and $b_1, b_2, \dots, b_k \in \Delta$,

$$h_{\Sigma}((a, b_1 b_2 \dots b_k)) = a$$

$$h_{\Delta}((a, b_1 b_2 \dots b_k)) = b_1 b_2 \dots b_k$$

- If $\mathbf{z}_i = (a, b_1 b_2 \dots b_k) \in \Gamma$ is a transition in the inferred regular grammar, the corresponding FST transition is

$$(q, a, b_1 b_2 \dots b_k, q')$$

Example

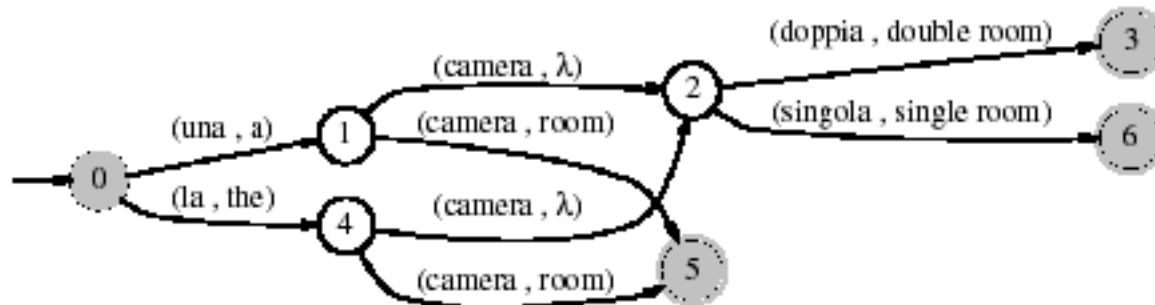


Figure 4

Bigram model inferred from strings obtained by the transformation \mathcal{L}_1 in example 2.

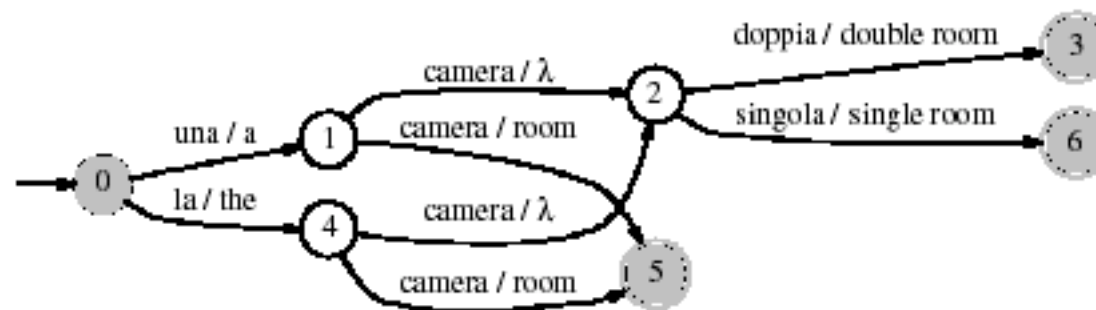


Figure 6

A finite-state transducer built from the n -gram of Figure 4.

Step 3: More Details

- Number of states in the FST: total # of k-grams ($k < n$) in the training set of extended strings, plus 1 (for the unigram)
- Number of transitions: # of k-grams ($k \leq n$) plus the number of states (back-off transitions)
- Actual number of k-grams depends on the degree of monotonicity of the corpus
 - Monotone corpus: number of expanded k-grams is close to the number of k-grams in either the source or target corpus
 - Non-monotone corpus: many more expanded strings
- N-grams are deterministic (non-ambiguous) but a fundamental property of the FSTs obtained in step 4 is that they can be non-deterministic (multiple paths corresponding to a single source string)

Experiments: EuTrans Project

- Domain: hotel reception desk conversations
- Spanish-English tasks:
 - EuTrans-0 - large, semi-automatically generated
 - EuTrans-I - more realistically-sized version of EuTrans 0
- Italian-English
 - EuTrans II - small, spontaneous speech
- Spanish-German
 - EuTrans-Ia - similar to EuTrans-I, but with much more non-monotonicity

Spanish-English Tasks

- Very small vocabulary (686 Spanish, 513 English, no OOVs)
- Seems rather contrived...
- Best results used Model 5 alignments and 11-grams for one domain, 6-grams for other
- Och and Ney's statistical alignment templates technique gets better results on EuTrans-I

Italian-English Task

- Transcribed telephone conversations and human transcriptions
- Similar domain as previous task (hotel reception desk) but much more spontaneous
- Vocab: 2.4k Italian, 1.7k English
- Very basic treatment of OOVs
 - If capitalized, keep word as own translation
 - If not capitalized, translation is empty string
- Results: WER 27%, BLEU 0.56 for 2, 3, 4-grams
- Many long sentences consist of short segments w/punctuation
 - Obtain better results by doing segmentation with dynamic programming first (WER 24.9-25.5%, BLEU 0.62) comparable to Och and Ney

Spanish-German Task

- Word order is more different for this language pair
- Vocab: 6.6k Spanish, 4.9k German
- More states and transitions due to more word reordering between source and target languages
- Delays of 1-5 word positions for 20% of target words produced (compare to 15% for English target)
- WER 10.6, BLEU 0.83 for 4, 5, 6-grams

Causes of Errors

1. Correct translations which are different from (single) reference
2. Wrong alignments of training pairs
3. Insufficient or improper generalization of n-gram-based GIATI learning
4. Wrong approximation of Viterbi-score-based search results

Most errors are due to reasons 1-3

Conclusions

- Comparable to other finite-state techniques when data is plentiful
- Better performance when data is rare
- Non-smoothed m-gram models are just a translation memory
 - Unigrams provide maximal generalization
 - $1 < n < m$ - other order n-grams provide varying degrees of generalization
- Smoothed n-grams/transducers can handle previously unseen source data
- Quality of generalizations depends on alignment quality and the extent of word-order preservation between source and target

Questions, Comments, Etc

- Search (sections 3.1 and 4.4)
 - Viterbi approximation
 - Byrne also discussed search last week
- Reordering of words/phrases
 - Kumar and Byrne's work
 - Other possibilities?
- How to handle out-of-vocabulary words?