Learning Dependency Transduction Models from Unannotated Examples

Hiyan Alshawi and Shona Douglas

Presented by Karim Filali
March 4th, 2004
Outline

• Weighted Head Transducers
• Dependency Transduction Models
• Training
• Experiments
Weighted Head Transducers

- Weighted Head Transducer: \((A,B,Q,I,F,T)\)
  - \(A\): input alphabet
  - \(B\): output alphabet
  - \(Q\): set of states \(q_0,q_1,\ldots\)
  - \(I\) and \(F\) subsets of \(Q\): initial and final states resp.
  - \(T\): set of transitions of the form \(<q,q',w',v',\alpha,\beta,c>\)
    - transition from state \(q\) to state \(q'\)
    - \(w'\): input symbol
    - \(v'\): output symbol
    - \(\alpha\): input position
    - \(\beta\): output position
    - \(C\): weight of the transition
How WHT Transitions Work?

- Consider transition \(<q,q',w',v',\alpha,\beta,c>\). In transitioning from \(q\) to \(q'\), the WHT reads input symbol \(w'\) at position \(\alpha\) and outputs symbol \(v'\) at position \(\beta\).

- If another transition is taken with the same input position \(\alpha\) (or output position \(\beta\)) as a previously taken transition, a symbol is read from (resp. written to) the next square adjacent to \(\alpha\) (or \(\beta\)) away from the head.
How Does a WHT Work?

• Non deterministic
  – Choose input symbol $w$ from input string together with initial state $q$ in $I$ associated with $(w,v)$ for some output symbol $v$. $w$ and $v$ will be at squares 0 of the input and output tapes resp.
  – Take a sequence of transitions until a final state (in $F$) is reached.
  – The derivation is valid if each symbol in the input string is read exactly once.
  – Output string is formed by taking sequence of symbols on target tape, ignoring white space.
Relationship to FSTs

• Head transducer can simulate a left-to-right transducer

• More expressive than FST
  – Example: reversing a string of arbitrary length
Dependency Transduction Models

• Collections of weighted head transducers

• For MT model, the transducers are applied hierarchically:
  – Takes advantage of locality of phrasal structure in natural language
  – Derives pairs of dependency trees (source and target): nodes are the words. Parents are called heads, children, dependents
  – Each dependency tree is ordered => sentence can be generated by recursive traversal
  – One-to-one mapping between source and target local trees

• Problem with using single transducer: insufficient generalization => large models, and data sparseness
Synchronization Example
Local Tree Derivation Using Head Transducers

• Each pair of local trees is derived by a head transducer
  – Input to transducer is flattened local source tree and output is flattened local target tree
  – Empty symbol $\epsilon$ to cope with different lengths of source and target strings

• Dependency transduction models compared to recursive transition networks for transduction of stochastic phrase structure grammars:
  – Strict left-to-right processing in RTNs requires delaying output with epsilon transitions
  – In DTM, use of transition positions relative to heads allows corresponding source and target words to be present in the same transitions => lexical translation and dominance probability relate directly to the model network structure.
Parameterization of the Dependency Transduction Model

- \( P(\text{transition with head words } w \text{ and } v \text{ and dependents } w' \text{ and } v') = P(q',w',v',\alpha,\beta|w,v,q) \)
- \( P(\text{choosing initial state } q_0' \text{ for subderivation headed by } w' \text{ and } v') = P(q_0'|w',v') \)
- \( P(\text{choosing } w_0 \text{ and } v_0 \text{ as roots of the two trees}) = P(\text{roots}(w_0,v_0)) \)
- Probability of derivation = \( P(\text{roots}(w_0,v_0)) \ P(\text{D}_{w_0,v_0}) \) where
  \( P(\text{D}_{w,v}) = P(q_0,q_{i+1}|w,v) \prod_{1 \leq i < n} P(q_i+1,w_i,v_i,\alpha_i,\beta_i|w,v,q_i)P(\text{D}_{w_i,v_i}) \)
- Cost of a derivation by a DTM = sum of all weights of the head transducer derivations involved
- For MT, target string is obtained by flattening of lowest cost target tree
- Dynamic programming to find the lowest cost dependency derivation
- When there is no derivation spanning the whole input string, the minimal length sequence of partial derivations with the lowest total cost spanning the whole lattice is chosen
Training

• Four stages:
  a) Compute co-occurrence statistics from data
  b) Search for optimal synchronized alignment
  c) Record hypothesized head-transducer transitions which can generate the alignments
  d) Compute maximum likelihood head-transducer weights from transition counts
a) Word Correlation Statistics

- For each $w$, assign a cost $r(w,v,b)$ for all possible translations $v$ in the context of the bitext $b$
  - $w$ and $v$ can be words, or compounds (contiguous words; here limited to 2)
  - $r(w,v,b) = \Phi(w,v) + d(w,v,b)$
  - $\Phi$: correlation measure normalized to $[0,1]$ with 0 indicating perfect correlation
  - $\Phi(w,v)$ initially computed from counts of bitexts in which $w$ and $v$ co-occur, $w$ occurs alone, $v$ alone, and neither $w$ nor $v$ occur. $\Phi(w,v)$ is refined during alignment
  - Pairing cost above works better than log probability of target word given source word (IBM models)
b) Hierarchical Alignments

- Four functions:
  - Alignment mapping $f$
  - Inverse alignment mapping $f^{-1}$ (to handle mapping target words to $\varepsilon$: otherwise coincides with $f$)
  - source head map $g$
  - target head map $h$
Conditions for Synchronized Hierarchical Alignments

1. Non-overlap: If $w_1 \neq w_2$, then $f(w_1) \neq f(w_2)$, and similarly for $v$ and $f'$

2. Synchronization: if $f(w) = v$ and $v \neq \varepsilon$, then $f(g(w)) = h(v)$, and $f'(v) = w$. Similarly, if $f'(v) = w$ and $w \neq \varepsilon$, then $f'(h(v)) = g(w)$, and $f'(w) = v$

3. Phrase contiguity: image under of $f$ of the maximal substring dominated by a head word $w$ is a contiguous segment of the target string
Optimal Hierarchical Alignments

- Cost of hierarchical alignment = sum of costs $r(w,v,b)$ of each pairing $(w,v)$ in $f$ (alignment function) (cost also includes penalties for the distance between heads and dependents)

- Dynamic programming to find complete hierarchical alignment that minimizes the cost function
  - Start with all possible subalignments with at most one source word (or compound) and one target word (or comp.)
  - Combine adjacent source substrings: one of the two subphrases is added as a dependent of the head of the other subphrase. This choice forces selection of a target dependent phrase because of synchronization
  - Subphrase selection: subphrase with highest alignment cost is dependent. Advtge: badly correlated segments at bottom of tree

- $\Phi(w,v)$ is reestimated from alignment pairings obtained after each DP round
c) Transduction Network
Toplogy: States and Transitions

- Construct head transducer consistent with hierarchical alignment
- Sharing of some model states arising from different training instances
- Example of construction:
  - Assume all source dependents are to left of head and no null source dependents
  - \( \sigma \): state naming function, takes sequence of strings to transducer states
  - For each \( w \) and \( v = f(w) \), construct states \( q_0 = \sigma(w,v,initial) \) and \( q_{w,v} = \sigma(w,v,final) \)
  - For each dependent \( w'_i \), \(-n \leq i \leq -1\), of \( w \) construct states \( q_i = \sigma(w,v,w'_i, f(w'_i), i) \) and transitions \( <q_0, q_{-1}, w'_{-1}, f(w'_{-1}), -1, \beta_1> \ldots <q_{i+1}, q_i, w'_i, f(w'_i), i, \beta_i> \ldots <q_{1-n}, q_{-w}, v, w'_{-n}, f(w'_{-n}), -n, \beta_n> \)
d) Transition Weights

- ML estimation of $P(q', w', v', \alpha, \beta \mid w, v, q)$ from the transition counts
- For this particular construction $P(q_0' \mid w', v') = 1$
Data Sets

• Human transcriptions of English sentences paired with their translations
• English-Spanish corpus
  – Air travel information enquiries, ~14000 bitexts and ~1200 held-out test bitexts
• English-Japanese corpus
  – ATT customer-operator conversations (half through operators), ~12000 bitexts + ~3000 for testing
  – Japanese hand segmented to correspond to English words
• A few thousand word vocabulary
• Short sentences (average length =7 words), spoken language
• Typical Spoken language errors
Evaluation Metrics

• Simple accuracy: edit distance
  \[ 1 - \frac{(I+D+S)}{R} \]

• Translation accuracy: adds transposition
  \[ 1 - \frac{(I+D+S+T)}{R} \]
Experiments

- WHT model better than baseline (word-for-word transducers which replace each source word with its most correlated target word in training set): ~70% versus ~40%
- Error reduced for both Spanish and Japanese but more so for Spanish
- Translation accuracy can be improved to 76% (from 74.2%) for Spanish and to 73.9% (from 72.2%) for Japanese using N-gram and case-based methods
Remarks

- Three assumptions underlying translation model:
  - Natural language strings decompose hierarchically into contiguous phrases
  - One of the words of a phrase, the head, determines how the phrase combines with other phrases
  - Decomposition of source string is strongly related to decomposition of target string
- Model lies between IBM models and hand-crafted ones
- Hierarchical decomposition results into faster search algorithms
- A priori knowledge such as a bilingual lexicon to guide construction of alignments might improve accuracy
- Room for improvement using a priori linguistics knowledge in the selection of head words during training
- Authors argue they are not ignoring role of semantics: based on the hypothesis that natural language strings decompose recursively into meaningful phrases, they find “natural” meaning representations
  - Advantage of avoiding expensive annotation of natural lang. strings
  - Deriving unambiguous meaning representations is challenging