The Survey of Statistics Machine Translation

Jiangming Liu

Due to Bayes theorem, the Noise Channel model $(P(S|T) \propto P(T|S) * P(S))$ is available and widely used. Given target language sentence T, we should choose the most probable source language sentence S to maintain $argmax_S P(T|S) * P(S)$.

Therefore, there are four main problems bellows

1, P(S) Language Model

2, P(T|S) Translation Model

3, parameter estimation -- training

4, the search algorithm -- decoding

Why P(T|S), not directly P(S|T) ?

The two factors (P(S) and P(T|S)) cooperate. This concentrates on well-formed English sentence.

● String2String

■ Word Alignment Peter F. Brown, 1990

Words near the beginning of the English sentence tend to align with words near the beginning of the French sentence and that words near the end of the English sentence tend to align with words near the end of the French sentence. But this is not always the case (distortion).

One way:

1, In IBM M1, it assumes all connections for each French position to be equally likely.

2, In IBM M2, the word alignment depends on the length of the target sentence and the position of the source word.

In IBM M1 and M2, it is over all possible position *i* in English sentence for a fixed position *j* in French sentence with *A*.

Another way:

1, In IBM M3, the probability of a connection depends on the tablet (probable translation word set), fertility (the number of words that can be aligned) and distortion (probable positions).

2, In IBM M4, take into consideration the words in an English string constituting phrases that are translated as units into French

3, In IBM M5, remove the deficiency in M3 and M4, which is that several words can lie on the top of one another, but words can be placed before the first position or beyond the last position in French string.

In IBM M3, M4 and M5, it is over a (or more which depends on fertility) probable position *j* in

French sentence for a fixed position i in English sentence with fertility, ignoring less probable alignment.

Tree2Tree

Inversion Transduction Grammars (ITG) Dekai Wu, 1995;1997

ITG is widely used for general corpora analysis, ITG is not to analyze ungrammatical input; rather the aim is to extract structure from the input data which is assumed to be grammatical for bilingual parsing. Meanwhile, the formalism in ITG is independent of the languages.

Context-free transduction grammar (CFTG) is useful for parallel bilingual sentences which have the same grammatical structure (or the same order of sentence). However, by minimally extending the generative power of context-free grammar (CFG), ITG yield a surprisingly effective result. During productions, ITG allows two possible orientations: the symbols generated by the right-hand side constituents of a production are concatenated in both the left-to-right order and the right-to-left order, which is extended from CFTG. In other words, notations in ITG allow combining graphical parse trees produced by bilingual language sentences with a possible to change the order of sub-trees. So ITG can generate many possible alignments between subsequences of sentence with a specific length. By the way, the constituent alignment includes a word alignment as a byproduct.

In my opinion:
Train for ITG: EM algorithm is for computing the probability of rules in both sides: source language and target language, furthermore dynamic programming algorithm is for the E-step.
Decode for translation using ITG: given a source sentence, we can obtain the target sentence with an optimal ITG tree using DP algorithm

Based on ITG Deyi Xiong, 2006

Based on ITG, a new approach of reordering phrase is proposed using ME classifier, which is to classify two types: *straight* and *inverse*. The classifier is trained over the features extracted from phrases.

Step 1: get the phrase

Step 2: link the phrase with one of types (straight and inverse)

Step 3: based on the word alignment, extract all the examples (o, b_1, b_2), where $o \in \{straight, inverse\}$, b_1, b_2 means two neighboring bilingual block from training data

Step 4: extract features using the function $h_i(o, b_1, b_2)$ to train a ME classifier model (reordering model)

Step 5: CKY style decoder that employ beam search algorithm to obtain the n-best result, which can be used in the phase where tuning the parameter using minimize error.

String2Tree

GHKM (Galley, 2004)

English parsing errors also cause trouble, and syntactic divergence is expected to be great. In the

face of these problems, there are three directions. The First direction is to abandon syntax in statistical machine translation. The Second direction is to abandon conventional English syntax and move to more robust grammars that adapt to the parallel training corpus. The third direction is to maintain English syntax and investigate alternate transformation models based on syntactic transformation.

GHKM follows the third direction. Given a source string S, a target tree T, and an alignment A, we can define the set $\rho_A(S,T)$ as the set of rules in any derivation $D \in \delta_A(S,T)$. We can regard this as the set of rules that we are entitled to infer from the triple (S,T,A)

To learn the set $\rho_A(S,T)$, we should be clear about the definition: *alignment graph*, *fragment*, *span*, *closure of span(n)*, *frontier set* and *frontier graph fragment*. *Alignment graph* is a rooted, directed, acyclic graph (this is a tree whose direction is top-down with an alignment to a string). *Fragment* is a sub-graph of nodes whose children are either all in the sub-graph or all not in the sub-graph. *Span(n)* is a set of nodes reachable from the subset of S that are reachable from the node *n*, *closure of span(n)* is shortest contiguous span which is a superset of *span(n)*. *Frontier set* is a set of nodes having a property that very node *n'* in graph connected to *n* and *n'* is neither ancestor nor descendant of *n*, in this condition, then *span(n')* and *closure of span(n)* has no intersection. *Frontier graph fragment* is a fragment where there is no other *n* whose span, *span(n)*, can cover or in the *span(frontier graph fragment)*.

Rule Extraction: 1, compute the *frontier graph fragment*; 2, compute the every possible combination of *frontier graph fragments* (e.g. every two, every three etc.); 3, form $\rho_A(S,T)$. This steps run in exponential time because of step 2. So how to get every *frontier graph fragment* in step2 running in linear time*:* 1, traversal all the node to get all span and implement span for each node which runs in linear time; 2, expand the node which is not in *frontier set* which runs in linear time.

Step 1: parse source language into phrase tree and each leaf is mapped to the word in target language using word-alignment

Step 2: extract rules with two algorithms capturing the minimal frontier graph fragment

- **GHKM-extend (Galley, 2006)**

Problems in GHKM: 1, just one derivation of alignment graph. 2, unaligned words. In GHKM-extend, all derivations will be represented by a form of derivation forest with a special node 'OR' to solve problem 1 and the problem 2 will be solve during forming derivation forest. GHKM-extend also assign the rules with probability, which is divided into two parts: Probability of syntactic parsing and syntax-based translation model

Tree2String

- **Liang Huang, 2006**

It is used Directly with a truly syntactic parse tree, not just SCFG, ITG, etc. and it will be more expressive than the hierarchical phrase model and ITG. Mostly the constitute parse tree (CPT) is used. The dependency parse tree (DPT) is closer to the semantic analysis then CPT does, although

it is difficult to specify reordering information

So its difference with the pattern-match? I think there is a deep syntactic parse in some string which are simply regard as a sequence

Step 1: get parse tree and related rules which are defined as usual

Each rule has a rank which is the number of variable and it is phrase rule if its rank is zero, in other word, there is no variable in that rule. And the derivation is defined

Step 2: training model and get the k-best derivation (encoder)

Depart from the noisy-channel approach, the model directly calculate the probability of target language given source language. The count of rule is used to estimate the probability of rules

Step 3: using the language model to re-rank the result

To maintain the unique translation output for each k-best derivation, keep a hash-table of unique string at each vertex in the hypergraph, and when asking for the next-best derivation of a vertex, keep asking until we get a new string, and then add it into the hash-table.

- Yang Liu, 2006

A Tree2String template for smt. In a word, in syntax-based xRs model, the translation quality is deeply affected on the quality of word alignment and parsing. Without the restriction, large memory and time should be need.

- Jun Xie, 2011

A dependency tree in source language is used and target is string. In the past, dependency structure based models (Xiong, 2007) typically employ both substitution and insertion operation to complete translation. As a result, they have to resort to heuristics or separate ordering models to control the word order of translation. However, in this paper, a head-dependents relation model will specify the order information in head-dependents rules. Furthermore, generalize the lexicalized words with their categories to alleviate data sparseness problem.

Train:

Step 1: get fragments in dependency parse tree. There are many pre-definitions: *head span* and its *consistent*, its *closure*; *dependency span* which mean a span of a dependency tree rooted *n*. furthermore, *acceptable head set* is a set of nodes, which are consistent; *acceptable dependent set* is a set of nodes, whose $dep(n) \neq \emptyset$; *acceptable head-dependents fragments* is a sub-tree, whose root is in acceptable head set and all sinks are in acceptable dependent set. Why? I think root is consistent so as to maintain continuous sub-string in target language can be translated using this rule and the sinks are not overlap from each other.

Step 2: extract rules

Lexicalized rules can be extracted directly from acceptable head-dependents fragments by replacing substitution with variable

Unlexicalized rules can be obtained by generalizing lexicalized rules. this process is that 1, generalization from lexicalized rules (using POS) and 2, filter the rules.

Step 3: assign rules with probability (model)

Decoder: bottom-up algorithm (CKY), pseudo translation rules, which means there is no reordering, will be used only if there are no acceptable translation rules. Furthermore, use a cube pruning and beam search algorithm.

I think the assign rules with probability is separated with extract rules, so the information of generalization is not used to the probability.

• Phrase-based method

■ Franz Josef Och, 2002, 2004; Philipp Koehn 2003

Step 1: get phrase

1, Phrase-level alignment is obtained from two direction word-level alignment (giza) and the phrase translation probability is computed as follow: $\phi(\bar{f}|\bar{e}) = \frac{count(\bar{f},\bar{e})}{\sum_{\bar{f}} count(\bar{f},\bar{e})}$. However, there is no smooth performance

2, based on 1, filter out non-intuitive pairs using syntax parse tree. Parse both sides of the corpus with syntactic parser and check if both phrases are subtrees in the parse tree.

3, phrase-based joint probability model is used and in additional EM algorithm, on one hand, a joint distribution $\phi(\bar{f},\bar{e})$ is obtained, and then conditional probability is also obtained; on the other hand, a joint distribution $distort(i,j)$ is obtained

Step 2: decoder (beam search)

• Hierarchical phrase-based method

■ David Chiang 2005

In phrase-based models, the phrases longer than three words improve performance little, suggesting that data sparse takes over the long phrase. Reordering model is too simple, that is reorder in the inner of one phrase obtained in many models. So the inversion between two groups of phrase is not captured.

The hierarchical phrase pairs come from formally productions of synchronous CFG. Anyway in the system (Yamada and Knight 2001) is both formally and linguistically syntax-based.

Step 1: Based on the rules extraction strategy, the rule set can be obtained. This phase is used to generalize. However, the number of rules is large, leading to two main problems: training and decoding is slow and spurious ambiguity leading to same *n-best*

Question: the initial phrase is too general to produce so many rules? Although there are some solutions. These solutions are just for limitation of rules, not consideration of syntax (case grammar or others?)

Step 2: Decoder algorithm is CKY with beam search

Sum them up, machine translation model: String2String translation (IBM models), Tree2Tree (in synchronous parse, rewrite a formalized Grammar, mostly rules), string2Tree (GHKM), Tree2String (phrase-based and hierarchical phrase-based, which is general; Dependency2String). So I think two points below:

1, only using statistics is not enough, like IBM models, phrase-based. With syntax (formal rules, which is assigned as probability), quality of machine translation will be improved.

2, generalization is necessary and effective, like that phrase-based goes to hierarchical phrase-based or it is possible for dependency to go to some models in future?