TR-S-0014

Language Modeling and Statistical Machine Translation

Ruiqiang Zhang

2001.2.21

Two parts are contained in this report. The first is to report the latest results of language modeling in speech recognition. Detailed information including local N-gram, long distance constraints and linguistic question triggers are integrated in language models by maximum entropy approach. The experimental results prove our models are effective. The second is to discuss our statistical Ngram translation model. Perplexity test was made to evaluate effectiveness of the proposed Ngram translation models.

Contents

1. Preface

The contents contained in this document conclude my main fruitful work conducted in ATR from 5/1/1999 to 2/28/2001, when two different laboratories are spanned, ITL to SLT.

Two different topics are described in it. One is language modeling integrating higher level knowledge such as syntactic and semantic knowledge and sentence structures. The other is related to bilingual machine translation using statistical mechanism.

The motivation of the first topic is to aid performance of traditional trigram model with a helpful linguistic knowledge, which is a joint work with Dr.E.Black and A.Finch and Prof.Lafferty. Syntactic and semantic information and man-made linguistic knowledge were integrated into trigram model using a framework of maximum entropy modeling. The results by our methods prove our intuition and in line with other researchers. Most of this part of work were made in ITL and a detailed report has been presented in TR-IT-0334. Writing in this report is not redundant although some narrations are the same as TR-IT-0334. A new speech recognition experiment was carried out in last year and this paper mainly reports this work.

Another topic in this work is related to machine translation. My work here proposes a phrase-based translation model in contrast to IBM word translation model. Because of limited work, only translation templates are made and a simulated evaluation using perplexity are reported in this paper. It needs more work on implementing real translation.

In the end I would like to express my appreciation to many many people who help me to research and to live here. Especially send appreciations to Sagisaka-san, President Yamamoto-dan, Black-san, Finch-san and all the secretaries.

2. Language Modeling

2.1. Introduction

It is well known that trigram models are the main adopted language models in speech recognition. But it appears intuitively that information beyond left two words in a document ought to help reduce uncertainty of predicting word. Using additional information beyond trigrams have been studied in language modeling in recent years.

To date, by virtue of their flexibility, maximum entropy (ME) methods have gained popularity in language modeling. In particular, ME models allow the modeler to integrate a variety of additional features into a base model, such as a trigram language model. Rosenfeld ([8]) and Lau et al.([5]) integrated long history word triggers into trigram models. Zhang et al. ([11]) integrated information from semantic and syntactic tags, and the parse structure of previous sentences of the document being processed into trigram models. Wu et al.([10]) combined syntactic head, part-of-speech tags and utterance topics with trigram models. A reduction of word error rate over trigram models has been achieved in all their work, which demonstrate the effectiveness of maximum entropy methods in language modeling.

The present paper undertakes to demonstrate that semantic/syntactic part-of-speech tags, and parse structure of previous sentences of the document being processed, can add trigger information to a standard trigram language model, over and above the improvement delivered by word/word triggering along the line of the work by Rosenfeld and lau et al. We formulate linguistic question triggers to query information from part-of-speech tags and words in the current and previous sentences.

As the source of both tags and parses in the present experiments, we use a 181,000 word subset of the 1-million-word ATR Genereal English Treebank. This subset consists of text drawn from Associated Press newswire and Wall Street Journal articles, which is the domain of our experiments.

The present paper extend the work of Zhang([11]), with two notable differences. Firstly, a higher accurate part-of-speech tagger was built to aid language model, resulting in an improved performance in word error rate.

Secondly, in order to present convincible results, special test data are chosen from 1992 Wall Street Journal speech corpus, a standard test set for speech recognition. These new features bring out new results of the experiments.

The paper is organized as follows. Section 2 gives a short introduction of ATR Tagset and ATR Treebank, which is the whole training and testing data of the experiments in this paper. Section 3 introduces the basic mathematical formulas of language models. Section 4 describes the work of POS tagging; a new tagger integrating multiple sources of information is built by maximum entropy methods. Section 5 describes the language

modeling experiments on a real speech recognizer of ATRSPREC. Section 6 discusses the overall research, the conclusions and future research.

2.2. ATR English Tagset and ATR Treebank

The approximate one million word ATR English Treebank was designed as training data for general English processing. Divided into roughly 950 documents of length 30-3600 words, this treebank achieves a high degree of document variation along many different scales--document length, subject area, style, point of view, etc. (See Table 1 for titles of nine typical documents). Text is tagged and parsed using the ATR English Grammar .

Empire Szechuan Flier (Chinese take--out food)
Catalog of Guitar Dealer
UN Charter: Chapters 1--5
Airplane Exit--Row Seating: Passenger Information Sheet
Bicycles: How To Trackstand
Government: US Goals at G7
Shoe Store Sale Flier
Hair--Loss Remedy Brochure
Cancer: Ewing's Sarcoma Patient Information

Table 1: Nine Typical Documents From ATR/Lancaster Treebank

The ATR English Tagset, used in this treebank, is unrestricted in its coverage, and particularly detailed and comprehensive, vis-a-vis other existing tagsets such as UPenn Tagset. Each verb, noun, adjective and adverb in the ATR tagset includes a semantic label, chosen from 42 noun/adjective/adverb categories and 29 verb/verbal categories, some overlap existing between these category sets. Proper nouns, plus certain adjectives and certain numerical expressions, are further categorized via an additional 35 ``proper--noun" categories. These semantic categories are intended for any ``Standard--American--English" text, in any domain. Sample categories include: ``physical.attribute" (nouns/adjectives/adverbs), ``alter" (verbs/verbals), ``interpersonal.act" (nouns/adjectives/adverbs/verbs/verbals), ``orgname" (proper nouns), and ``zipcode" (numericals). The semantic categorization is, of course, in addition to an extensive syntactic classification, involving some 165 basic syntactic tags.
For detailed presentations, see [3][2]. An apercu of the characteristics of this tagset, however, can be gained from Figure 1, which shows a sample sentence from the ATR Treebank (and originally from a Chinese--food take--out flier), tagged with respect to the ATR General English Tagset.

(_(Please_RRCONCESSIVE Mention_VVIVERBAL-ACT this_DD1
coupon_NN1DOCUMENT when_CSWHEN ordering_VVGINTER-ACT

OR_CCOR ONE_MC1WORD FREE_JJMONEY
FANTAIL_NN1ANIMAL
SHRIMPS_NN1FOOD
Figure 1: Two ATR Treebank Sentences from Chinese Take--Out Food Flier (Tagged Only -- i.e. Parses Not Displayed)}

The generality and riched semantic and syntactic tagset of ATR treebank make it a very good training data for processing general English statistically. In this paper it was used for building a part-of-speech tagger and an advanced language model by maximum entropy methods.

2.3. Mathematical Fundamentals

In this section two mathematical formulas, maximum entropy and mutual information, are described because of their core roles in this paper. The first one is used to building language models integrating multiple information formalized as triggers. The second is to find the most informational triggers with complementary information.

2.3.1. Maximum Entropy Modeling

We consider a random process that produces an output symbol y, a member of a finite set Y. In generating y, the process may be influenced by some contextual information h. Our task is to estimate the conditional probability $p(y|h)$.

If given a training data, we can collect a large number of samples $(h_1, y_1), (h_2, y_2), \cdots, (h_N, y_N)$, each sample (h, y) consists of a predicted output symbol y and contextual information h of y.

Sometimes we want to choose some interesting 'triggers' from the training data. A trigger pair is formulated as (s,t), s can be any kind of triggering feature. In the case of long history trigger, s is the presence of a particular word in the history. For linguistic question triggers, s is the boolean answer to a question. t is synonymous with y, a predicting feature triggered.

Suppose we select a large number of triggers $(s_1,t_1),(s_2,t_2),\cdots,(s_M,t_M)$ we define a trigger function as follows:

$$f_i(h,y)=\begin{cases}1 \text{ if } s_i \text{ occurs in } h \text{ and } t_i \text{ is } y\\ 0 \quad \text{otherwise}\end{cases}$$

The above equation means the trigger function $f_{s,t}(h, y)$ is a binary-valued function. If and only if the trigger (s,t) occurs in the sample (h, y), the value of the trigger's corresponding trigger function is set 1.
Our task is to estimate the probability $p(y|h)$ if given training data $(h_1, y_1),(h_2, y_2),\cdots,(h_N, y_N)$
and triggers $(s_1,t_1),(s_2,t_2),\cdots,(s_M,t_M)$, The presence of triggers constraints the probability distributions $p(y|h)$ as follows:

$$\sum_{h,y} p(h, y) f_k(h, y) = \sum_{h,y} \tilde{p}(h, y) f_k(h, y) \qquad (1)$$

where:

$$p(h, y) \approx \tilde{p}(h) p(y|h)$$

Here, $\tilde{p}(h)$ and $\tilde{p}(h, y)$ are empirical probability distributions, defined by

$$\tilde{p}(h) = \frac{\#(h)}{N}, \quad \tilde{p}(h, y) = \frac{\#(h, y)}{N} \qquad (2)$$

$\#(\bullet)$ means number of times that $(\bullet)$ occurs in the sample.
Then the maximum entropy solution satisfying the constraint equations 1 and 2 is derived as follows:

$$p(y|h) = \frac{\exp(\sum_{i=1}^{M} \lambda_i f_i(h, y))}{Z(h)} \qquad (3)$$

where $Z(h)$ is a normalizing constant determined by the requirement that $\sum_y p(y|h) = 1$ for all y,

$$Z(h) = \sum_y \exp(\sum_{i=1}^{M} \lambda_i f_i(h, y)) \qquad (4)$$

A predefined initial distribution may be used in order to reduce heavily training burden. If given an initial model $p_0(y|h)$, we add another constraint

$$p = \arg\min D(p \| p_0) \qquad (5)$$

where: D is the Kullback-Leibler distance.

Then the the maximum entropy solution satisfying the constraint equations 1, 2 and 5 is,

$$p(y \mid h) = \frac{\exp(\sum_{i=1}^{M} \lambda_i f_i(h, y)) p_0(y \mid h)}{Z(h)} \qquad (6)$$

Clearly if we choose the initial distribution as uniform distribution, Equation 3 is a special case of equation 6.

In equation 6 λ_i is a weight of trigger f_i. An improved iterative scaling algorithm is used to train model 6 to obtain λ_i (See [7]).

2.3.2. Mutual Information

When considering a particular trigger pair (s,t), we are interested in the correlation between s and t. We can assess the significance of the correlation between s and t by measuring their mutual information. We use the same formula as [8] to calculate mutual information.

$$\begin{aligned} MI(s,t) = & P(s,t) \log \frac{P(t \mid s)}{P(t)} \\ & + P(s,\tilde{t}) \log \frac{P(\tilde{t} \mid s)}{P(\tilde{t})} \\ & + P(\tilde{s},t) \log \frac{P(t \mid \tilde{s})}{P(t)} \\ & + P(\tilde{s},\tilde{t}) \log \frac{P(\tilde{t} \mid \tilde{s})}{P(\tilde{t})} \end{aligned} \qquad (7)$$

2.4. Part-of-speech Tagging

In our work, part-of-speech tags played an important role because the language models require part-of-speech information. There are many methods to build a tagger such as
N-gram models([6]), decision trees([3]), transformations([4]) and maximum entropy approach([1]).
But the tagge we used in this paper is distinguished from the previous by adopting some new features .
(1) We use a much more detailed tagsets(see Section 2). There are over 3,000 tags in ATR Tagset, far more than the rudimentary, 45-tag UPenn Tagset. This kinds of tagging with such a bigger tagset have been seldom done before.

(2)The information we used to build the tagging model is extremely riched, vis-a-vis other taggers. In our tagger, we integrated into the tagger model local word and tag information, as opposed to other taggers where only one type of information of those mentioned above was used.
Our tagging model is a maximum entropy(ME) model similar as Equation 6. For the case of predicting tag t based on the tag history h, Equation 6 is rewritten as

$$p(t \mid h) = \frac{\exp(\sum_{i=1}^{N}\sum_{j=1}^{M} \lambda_{ij} f_{ij}(h,t)) p_0}{Z(h)} \qquad (8)$$

where:
N is the number of trigger types. We defined 17 trigger types in the model.
M_i is the number of the $i-th$ triggers. The trigger was selected if its mutural information value was higher than a predefined threshold.
p_0 is a uniform distribution model.

In this experiment, the history h is defined as $(w_{-2}, w_{-1}, w, w_1, w_2)$, whose meaning are shown in Table 2. The last collumn of Table 2 is from an example in Figure 2.

w	word whose tag we are predicting	lose
t	tag we are predicting	VVOCONTROL
t_{-1}	tag to the left of tag t	NP2GROUP
t_{-2}	tag to the left of tag t_{-1}	AT
w_{-1}	word to the left of word w	Cowboys
w_{-2}	word to the left of word w_{-1}	the
w_1	word to the right of word w	the
w_2	word to the right of word w_1	game

Table 2:illustration of symbles

the	Cowboys	lose	the	game
AT	NP2GROUP	VVOCONTROL	AT	NN1COMP-B

Figure 2: a tagger example

We use the treebank data described in Section 2 as our training and test data. It contains one million words and was separated into two parts:

a set of 850,000 words, the training data, which was used to build the models

a set of 53,000 words, the test data, which was used to test the quality of the models.

The tagger of Equation 8 was trained and tested by the above data. The experimental results are shown in Table 3.

The first column shows the trigger type used. For example, trigger $(w_{-1}w,t)$ means a combination of current word w and left of this word w_{-1} is used to trigger predicting tag t. As shown in Figure 2, combination (*the Cowboys, VVOCONTROL*) fall into this trigger type.

The second column is the number of triggers of the first column.

The third and the fourth column are the perplexity and tagging accuracy of test data, respectively. Firstly we presented both the results of using single type trigger and then give the results of using all the trigger types together in the last row.

The tagging results obtained are better than the N-gram tagger, previously used in [9], 10% improvement with regard to accuracy. It shows maximum entropy approach is a powerful method to integrate multiple information sources. When we combined all the triggers into one model, the results are much better than only one single type trigger is used. Our tagger model is improved by using ME over N-gram model.

Trigger Type	Number of triggers	Test PP	Accuracy(%)
(w,t)	73162	3.59	75.06
$(w,t)+(w_{-2}w_{-1}w,t)$	73162+15957	3.56	75.30
$(w,t)+(w_{-1}ww_1,t)$	73162+16667	3.54	75.90
$(w,t)+(ww_1w_2,t)$	73162+16345	3.54	75.60
$(w,t)+(w_{-1}w,t)$	73162+14708	3.51	76.12
$(w,t)+(ww_1,t)$	73162+15789	3.47	76.52
$(w,t)+(t_{-1},t)$	73162+18520	3.15	76.14
$(w,t)+(t_{-1},t)+(t_{-2}t_{-1},t)$	73162+18520+15660	3.11	76.24
$(w,t)+(t_{-1}w_1,t)$	73162+12302	3.40	76.26
$(w,t)+(t_{-1}ww_1,t)$	73162+21564	3.51	76.12
$(w,t)+(w_{-1}w_1,t)$	73162+12496	3.47	76.14
$(w,t)+(w_{-1},t)$	73162+28415	3.33	76.90
$(w,t)+(w_1,t)$	73162+27380	3.34	76.78
$(w,t)+(t_{-1}w,t)$	73162+14212	3.44	75.78
$(w,t)+(t_{-2}t_{-1}w,t)$	73162+18699	3.47	75.40
$(w,t)+(w_{-2}w_{-1},t)$	73162 +9811	3.53	75.92
$(w,t)+(w_1w_2,t)$	73162+9733	3.52	76.01
ALL		3.07	78.80

Table 3:Experimental Results of Tagging Using Detailed Local Constraints

2.5. Language Modeling of ATRSPREC

2.5.1. Linguistic Question Triggers

The motivation of this paper is to evaluate additional information in aiding a conventional trigram model. The additional information mentioned here points to part-of-speech tag and words in the long history left to the predicting word, including current and previous sentences. By using long history word triggers, long history word information have been addressed by Rosenfeld [8] and Lau [5]. For part-of-speech information in the long history, we introduce a special designed trigger, linguistic question triggers to express it. In addition to part-of-speech, linguistic question triggers also contain information from long history words. In detail, linguistic questions were written by a professional grammarian, which query either:(a) the tags of the words to the left of, and in the same sentence as, the word being predicted; or (b) tags within any or all of the previous sentences of the document to which the word belongs that is being predicted; or both of (a) and (b) together. Each of these questions then triggers a particular word in the vocabulary. Some examples of linguistic questions are shown in Table 4.

The first question queries whether there exists a verb having semantic meaning ``help" to the left. For example, words like helped/helps/assisted/fixed and et al. In fact, this question searchs for a subset of part-of-speech tag( VVDHELP, VVIHELP, VVNHELP, ...) in the left history. If any of tags in the brackets occurs, the answer of this question is ``YES".

The second question queries if a noun with semantic meaning ``money" lies within two words to left. This question searchs for part-of-speech tag either NN1MONEY or NN2MONEY. Words such as revenue, money, cost, prices make this answer yes.

The third question asks whether the occuring times of a noun with semantic meaning ``animal" in the last 6 sentences to the left is greater than 2 less than 10000. If so, the answer is ``YES".

The fourth question query sentence structure of sentences to the left. Although questions related to sentence structures were written by the grammarian, we did not use them in the experiments because these questions depend on the availability of the full parse for previous sentences that are queried. In current, our ATR English Parser runs too slow to be incorporated into speech recognition.

From examples of linguistic questions in Table 4, it is understood that questions embody information from part-of-speech and words in current and previous sentences. But, these part-of-speech and word information were not used direcly as other people did, they are combined and refined by a special grammarian and expressed in a format of question. Based on differing contextual history with regard to the predicting word, the answers of

questions are either ``YES" or ``NO", which indicate a correspondence of question with the predicting word. This correspondence may be expressed by linguistic question triggers and selected from ATR treebank based on their mutual information values. And then used in the same way as long history word triggers, linguistic question triggers are integrated in language models by maximum entropy models.

#	Question Description
1	v sem help to left
2	n sem money within two words to left
3	2 ≤ FREQ_6_n_sem_animal_to_left ≤ 10000
4	current or recent node is sd with n subject n sem person verb v sem verbal act

Table 4:Examples of Questions

Considering language modeling for predicting words, Equation 6 is rewritten as

$$p(w|h) = \frac{\exp(\sum_{i=1}^{M} \lambda_i f_i(h,w)) p_0(w|h)}{Z(h)} \qquad (9)$$

In this paper we make a comparision of 4 language models:
1. A base trigram model
2. A ME model using trigram model as a base model integrated long history word triggers
3. A ME model using trigram model as a base model integrated linguistic question triggers.
4. A ME model using model 2 above as a base model integrated linguistic question triggers.

Model 1, a trigram model, acts as a baseline model for the later advanced model. It is a standard back-off trigram model built by CMU toolkit. Model 2 is used to evaluate effectiveness of long history word triggers. Model 3 is used to measure effectiveness of linguistic question triggers. Model 4 is to check the joint results by long history word triggers and linguistic question triggers.

2.5.2. Perplexity Evaluation

The well—established trigram LM was used as the base LM for our experiments. This model was selected because it represents a respectable language model which most readers

will be familiar with. The ME framework was used to build the derivative models since it provides a principled manner in which to integrate the diverse sources of information needed for these experiments.

For training a baseline trigram language model, we used a corpus of newspaper text drawn from 1987--1996 Wall Street Journal and Associated Press Newswire in equal proportion. Certain types of words were mapped to generic tokens representing the class of word. These were: words representing time of day (e.g. 12:21), dates (e.g. 11/02/64), price expressions (e.g. ¥$100) and year expressions (e.g. 1970--1999). The mapping was done using simple

regular-expression pattern matching. The substitutions were implemented to assist the trigram model, which is unable to ask questions about the internal structure of words and cannot be expected to form useful n-grams from this class of words. The linguistic questions, however, being able to query the word's internal structure, were more effective on the raw words themselves and were used in that way. The vocabulary, and therefore the words being predicted, was constructed from data in which these tokens had been mapped. In total, the training data consisted of 20 million words.

Using the same 20 million training words as for the trigram model, and the techniques described in section~¥ref{sec:mathematics}, About 60,000 long history word triggers were selected. A language model (referred to as trigram + WTmodels), integrating long history word triggers with a baseline trigram model was built. We then used the ME model built by adding word--triggers to the base model as the base model for a second ME model which incorporated our question--based triggers. We found this approach effective in

dealing with the large number of triggers involved. The number of question--based triggers used was 95,486 and the question set size from which the triggers were produced was 6,455.

As the training data for the tagger used in these experiments, we use a 181,000--word subset of the approximately--1--million--word ATR General English Treebank. This treebank subset consists exclusively of text drawn from Associated Press newswire and Wall Street Journal articles. The 181,000 words are partitioned into a training set of 167,000 words and a test set of 14,000 words. We utilize this portion only of the treebank, as opposed to the entire corpus, in order to match the text type of the raw data set used to train our baseline n--gram language model, which composed of AP and WSJ text in roughly the same proportions. This tagged data was used to train the following two language models. The first referred to as ``trigram+Q's", integrated question triggers with a base trigram model; the second referred to as ``trigram+Wtmodel+Q's" integrated question triggers with a base long history word trigger model (ie. with "trigram+WTmodel"). The number of triggers used in the various model components is shown in Table 5.

Perplexity evaluation data consisted of 14,000 words of hand--labelled and --parsed ATR treebank, again drawn in the same proportion from Wall Street Journal and Associated Press.

In Table 5, ``trigram" is the perplexity of the base trigram model before any ME training. ``trigram+WTModel" is the perplexity of the ME model which combines long history word triggers with the base trigram model. ``trigram+Q's" is the perplexity of the ME model combining question triggers with the base trigram model. ``trigram+WTModel+Q's" is the perplexity of the full ME model (using all the features) after training.

trigram models	20001(uni) 395663(bi) 527782(tri)
word triggers	37567
question triggers	95486
question numbers	6455

Table 5: Number of constraints in the models

Model	PP	Change(%)
trigram	153.0	-
trigram + WTModel	130.0	15.0
trigram + Q's	133.6	12.7
trigram + WTModel + Q's	118.7	22.4

Table 6: Perplexity Results

2.5.3. Recognition Error Rate

To evaluate our technique we used the ATRSPREC speech recognition system, designed by ATR for speaker-independent English recognition. We carried out this experiment using the Wall Street Journal Corpus (WSJ0). The acoustic model was trained by ATRSPREC toolkit, using speaker independent training data---SI84 in the WSJ0, consisting of 7193 sentences from 84 different speakers.

Our test data are from the WSJ0 Evaluation data. The sentences in the evaluation data were grouped into (usually very short) paragraphs. Since the language models are relying on the long-range history for contextual information, and our questions may query information from the 6 sentences before, only paragraphs containing 6 or more sentences were used. A total of 21 paragraphs, containing 184 sentences, were selected from the directory si_et_jr of the WSJ0 as test speech data. A example paragraph with 8 sentences Some example sentences are listed in Figure 3.

Our language model was used to rescore the best N hypotheses output by the speech recognition system, yielding a new, reordered N-best list of hypotheses. In this experiment, we set N the number of hypotheses output from ATRSPREC at 200.

The rescoring process proceeds as follows and is shown algorithmically in Figure 4. The tag for each word in each hypothesis is predicted using our tagger. This tag sequence is then combined with the predicted tags from

previous sentences to form a history for this hypothesis. Linguistic questions are asked of the history and the answers are combined with the base LM (Equation 9) to obtain probabilities for each word in the hypothesis. And then, the chain rule is used to combine these, to produce the language model's probability of the hypothesis. This probability is linearly interpolated with the probability from the acoustic model to yield the probability used to rank the N-best list of hypotheses. Finally the hypothesis with the highest probability is saved as the scoring result of this sentence and used as a history for next sentence hypothesis.

Querying the full parse of the sentences would be desirable, and although possible, the parsing process is much slower than tagging, and possibly too inaccurate to be useful (the parses will be on top of already errorful predicted tags). Thus for the purposes of these experiments we restricted ourselves to questions over a tagged history.

All the LMs used in this experiment for rescoring were the same as those in the perplexity experiment. The experimental results are shown in Table 7.

Model	WER(%)
trigram	26.5
trigram + WTModel	24.8
trigram + Q's	24.3
trigram + WTModel + Q's	23.8

Table 7: The recognition results

#	
1	gum chewers in singapore were in a sticky situation wednesday
2	that was when the nationwide ban on gum chewing went into effect
3	the gum chewing ban was a direct result of the country's new emphasis on hygiene
4	in fact retailers who handle bootleg bubblegum could face a fine as high as two thousand dollars
5	it has already been illegal to spit in singapore for the last fifteen years
6	and littering has been a capital offense in this asian country since the end of the second world war
7	the ban on chewing gum is the latest effort for the government to live up to the motto that graces every state flag
8	it reads cleanliness is next to buddha

Figure 3: An example of tagged (test) data

FOREACH hypothesis $hp_0, hp_1, \cdots, hp_N (N = 200)$

(The hypothesis hp_i is a word sequence $w_0 w_1 \cdots w_L$ where $L+1$ is the number of words in this hypothesis)

TAG this word sequence, to obtain tag sequence $t_0 t_1 \cdots t_L$

FOREACH $j \in \{0,1,\cdots,L\}$

Create history $h_j = w_0 w_1 \cdots w_{j-1} + t_0 t_1 \cdots t_{j-1}$ + history derived from previous sentences

Find active word triggers and question triggers based on h_j and use Equation 1 to get $P(w_j \mid h_j)$

Calculate $P_{LM} = \prod_{j=1}^{L} P(w_j \mid h_j)$

Calculate $P_{\text{int}erP}(hp_i) = \lambda_1 P_{LM}(hp_i) + \lambda_2 P_{AM}(hp_i)$

OUTPUT the best hypothesis: $hp_{best} = \max_{i=0,N} P_{\text{int}erP}(hp_i)$

ADD the word sequence and tag sequence of hp_{best} to the history derived from previous sentences

where:

- $P_{LM}(hp), P_{AM}(hp)$ are the language/acoustic model probabilities of hypothesis hp;
- h_j is the word and tag history preceding word w_j;
- Interpolation weights λ_1 and λ_2 were chosen empirically.

Figure 4: The rescoring process

2.6. Discussion

The experiments presented here have focused on showing that we can glean useful information from the linguistic analysis of syntactic and semantic tags in the history of the word being predicted. The experiments demonstrated that this information gives a reduction in perplexity, of 90% of that provided by the long--range word triggers used by [8] . Moreover, when these triggers are used in conjunction with a model incorporating long--range word triggers, 91% of the perplexity gain from both sources is inherited by the new model. When the technique (restricted to questions about tagged text in the history) is integrated into a speech recognition system, the word error rate of the system is reduced. The gain in word error rate using linguistic questions is approximately the same as that of the long history word triggers. When both sets of features are used, most of the error rate improvement from both techniques is passed on. This indicates that the information we are providing is new and complementary. This is in line with our intuition, given the nature of the questions we ask.

It is clear from the results of these experiments that there is a significant amount of useful information for a language model in the parses and tags in the history. We feel that further improvements can be made by developing the language we are using to ask these questions and thereby improving their expressive power. Although impracticable at present, it would also be desirable to extend the technique to query the parse structure of the history. Finally, we feel that large gains will follow improvements in the base speech recognition system accuracy.

3. Machine Translation
3.1. INTRODUCTION

Since Brown et al. [12] introduced a statistical model for machine translation, which is, a translation model of modeling a noise channel and a language model of modeling a target language, and change the problem of machine translation into finding a maximum posterior solution under such source-channel formalism, there have been a lot of papers dealing with the problem of machine translation by following this road [14][16][20].

Although many kinds of translation models have been applied, all these models tried to model word-to-word correspondences between source and target and further restricted that each source word was assigned exactly one target word. From the viewpoint of language models, these translation model are no more than unigram translation models that have been proved less efficient. Moreover, considering the results reported in [21] the alignment template system of using phrase-to-phrase correspondences shows much better results than word-to-word EGYPT models. This encouraged us to build a translation model under the consideration of contextual information.

In this paper, we address the problem of building a N-gram translation model in machine translation. The aim of these doings is two-fold: to better model translation models by inducing strong contextual constraints, and to predict probabilities of word combinations of source language other than a single word by translation models. In addition, This N-gram translation model has an advantage of using word-to-word bilingual alignment model to implement phrase-to-phrase correspondence, other than using the complicated phrase-to-phrase bilingual alignment mode in [14][19].

In what follows, section 2 introduces the idea of N-gram translation template. Section 3 describes how to use these translation templates to build a N-gram translation model. Section 4 presents the experiment results on perplexity of using these translation templates. Section 5 discusses the future research direction.

3.2. N-GRAM TRANSLATION TEMPLATES

As shown in Fig.5, the problem of statistical machine translation is to decode out a target language sentence given a source language sentence. A successful solution of this problem depends on: (a) information abstraction from a huge training bilingual data and (b) methods of applying the information to generate a translation of an unseen source sentence resembling the translation of the seen source sentences in the training data as closely as possible. This gives rise to a requirement that information gleaned from training bilingual data should be as rich as possible. But after reviewing methods used in [14][16][20], translation models did not integrate detailed information. Although Brown proposed a complicated basket of translation models such as fertility models and alignment models, it is controversial how well these models work.

In this paper we proposed a new method called as N-gram translation templates to convey information from bilingual data. An automatic alignment program was applied to complete word-to-word correspondences.

And then N-gram translation templates were built based on the correspondences.

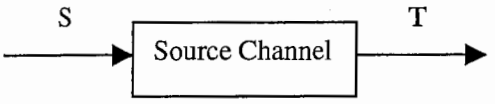

Figure 5: The source-Channel Model of translation

A N-gram translation template is formalized as a triple (s,t,p), s is a neighbored N words in a source sentence. t is the aligned target words of s. p is the probability of t given s, that is, $p(t \mid s)$.

As shown in Fig.6, it is an example of Chinese/English aligned sentences(automatic alignment by GIZA), in which source and target language are Chinese and English respectively. (为了, for) is an unigram translation template, (为了/明天, for tomorrow) is a bigram translation template. (为了/明天/预约, reservation for tomorrow) is a trigram translation template.

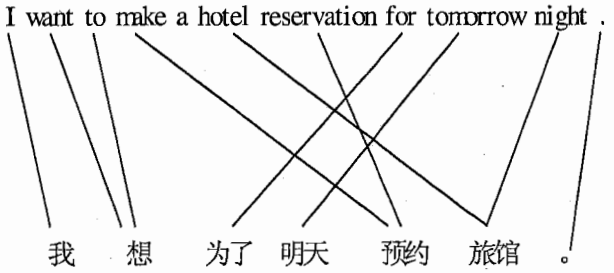

Figure 6: Example of Chinese/English alignment

Given aligned bilingual training data, we can collect a set of N-gram translation templates,

$U = \{(s_0,t_0,p_0),(s_1,t_1,p_1),\cdots,(s_K,t_K,p_K)\}$

K is the whole number of translation templates. We presented a method of building N-gram translation templates in the experiment.

3.3. N-GRAM TRANSLATION MODELS

Before we derived N-gram translation models, we described our modeling to source sentences and target sentences. We segment a source sentence using a length N window from left to right and view this source sentence as an integration of overlapped N-gram word sequences. That is,

For a source sentence $X = x_0 x_1 x_2 \ldots x_M$

It is modeled as an overlapped connection of $M-N$ word sequences XC,

$$\begin{aligned} XC &= x_0 x_1 \cdots x_{N-1} + x_1 x_2 \cdots x_N \\ &+ x_2 x_3 \cdots x_{N+1} + \cdots + x_{M-N-1} x_{M-N} \cdots x_M \\ &= xc_0 + xc_1 + \cdots xc_{M-N} \end{aligned}$$

We defined this modeling of source sentences as N-gram cross-source models.

Provided that a set of translation templates U and an unseen target sentence are known, based on translation templates we can take out a set of phrases from the known target sentence, which are target part of translation templates and appear in the unseen target sentence. That is, we decompose the unseen target sentence into a set of subunit defined by translation templates. We formalized our words as:

Given a target sentence

$$Y = y_0 y_1 y_2 \cdots y_H$$

Its translation template decomposition is defined as

$$YC = yc_0 + yc_1 + yc_2 + \cdots + yc_L$$

yc_i is a decomposition of Y. L is the number of all decompositions. yc_i is formed from words of Y and yc_i is one of target part of a translation template U, i.e.

$yc_i \in Y$ and $yc_i \in U$

As a consequence of modeling source sentence X and target sentences Y as XC and YC, we defined the joint probability of a source sentence and a target sentence as

$$P(X,Y) = P(XC,YC) = P(YC)P(XC \mid YC)$$

The equation above is derived by Bayes' law. But in the case of N-gram translation model, we replace $P(XC \mid YC)$ with $P(YC \mid XC)$, although it is incorrect from strict mathematical viewpoint. But it is more convenient for models. The same usage can be found in [4]. We rewrite out our model of bitext machine translation as:

$$P(X,Y) = P(XC,YC) \Rightarrow P(YC)P(YC \mid XC) \qquad (10)$$

Furthermore, we calculate the translation model as:

$$\begin{aligned} P(YC \mid XC) &= P(yc_0, yc_1, \cdots, yc_L \mid xc_0, xc_1, \cdots, xc_{M-N}) \\ &= \prod_{j=0}^{M-N} \sum_{i=0}^{L} P(yc_i \mid xc_j) \end{aligned} \qquad (11)$$

xc_j is a N-gram source word sequence. We compute the probability $P(yc_i \mid xc_j)$ as an interpolation from unigram to N-gram. For a case of trigram, it is written as:

$$\begin{aligned} P(yc_i \mid x_1 x_2 x_3) &= \lambda_1 f(yc_i \mid x_1 x_2 x_3) + \lambda_2 f(yc_i \mid x_1 x_2) \\ &+ \lambda_3 f(yc_i \mid x_2 x_3) + \lambda_4 f(yc_i \mid x_1) \\ &+ \lambda_5 f(yc_i \mid x_2) + \lambda_6 f(yc_i \mid x_3) \end{aligned} \qquad (12)$$

Where:

$$f(y \mid x) = C(x, y) / C(x)$$

$$\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 + \lambda_5 + \lambda_6 = 1$$

Equation (10)(11)(12) are the mechanism of our proposed machine translation. Some concluding points are:

(1) In the mechanism of proposed translation, we use a language model and a translation model.

(2) A language model is a conventional N-gram language model.

(3) A translation model is a N-gram translation model defined by our modeling of source and target sentences and computed by Equation (11).

3.4. EXPERIMENT

3.4.1. Resources

Our translation task was restricted in the domain of travel-related topics such as hotel reservation, sightseeing information query and so on. ATR collected a huge multilingual language database related to travel task. The training and test data used for the present experiment were selected from this database, containing 169 conversions, 19402 sentences and 192,430 words.

Each sentence consists of 4 different translations described by four different language, English, Chinese, Japanese and Korean, respectively. Every two of four can be used as training data for different language translation. The bilingual data used for current experiment were Chinese/English part. Our task was to translate Chinese into English.

The tools available for the Chinese/English translation were: a Chinese word segmentation program, an English tokenizer and a machine translation toolkit, EGYPT released by CLSP/JHU.

The Chinese word segmentation program was built by statistical N-gram methods and trained by adopting People Daily news corpus. This program has been used for building word-based language models in Chinese speech recognition. Its segmentation accuracy was acceptable with the measure of speech recognition. But if it is used for language processing such as POS tagging and translation, this accuracy seems lower than expected, getting even worse when used to segment travel related data. As a consequence, we did a lot of manual work to correct segmentation errors and adding many new words such as Japanese name and place name.

The English tokenizer used was made by ATR. After Chinese word segmentation and English tokenization, we input the Chinese/English data to Whittle for preparing training and test data for GIZA. Table 8 shows a summary of data used in the GIZA after Whittle. Fig. 7 shows the training process described.

3.4.2. Model Building

Released by CLSP/JHU, GIZA is a program for training IBM's model from bilingual data. It uses EM method to iteratively train IBM Model 3 described in [12]. During the training of Model 3, it passes Model 1 and Model 2. For detail description of GIZA, please refer to [20].

	Chinese	English
TRAIN Sentences	18922	
Words	178,443	196,263
Vocabulary	4736	6078
TEST Sentences	200	
Words	1755	1893

Table 8: a Chinese/English database

In this experiment we were only interested in getting the Viterbi allignment of bitexts, which is one of the results output by GIZA. In the option we set for running GIZA, the iteration times for each model (1/2/3) is 10. Fig. 7 shows the perplexity of test data after each iteration.

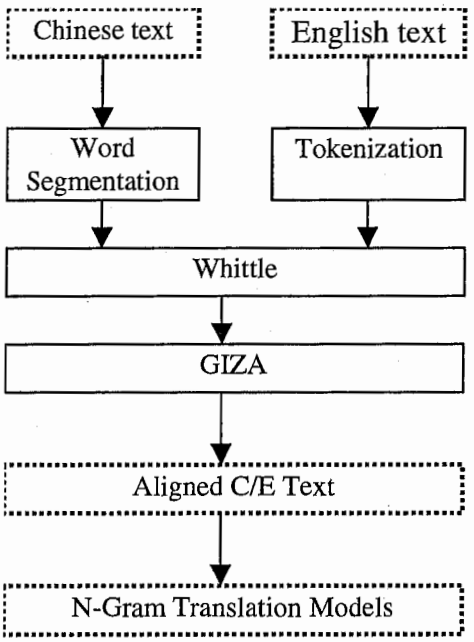

Figure 7: Training framework of N-gram Translation Model

Our aim was to get a better alignment of bilingual data for building N-gram translation model. But the alignment results given in the previous

experiment were not good visually. This alignment was made by GIZA without a bilingual dictionary. Our next experiment was made with a dictionary. We made a Chinese/English dictionary by the following steps. First, we applied GIZA to align Chinese/English without a dictionary. Second, from this alignment take out source words and its target matched words. Each pair of source and target is an item of an initial dictionary. Third, manually remove from the initial dictionary those items where source and target translation were wrong, resulting the final Chinese/English dictionary. As to our experiments, the initial dictionary consisted of 38,000 items chosen from the alignment without a dictionary. After manual work we got a final dictionary containing 4879 items, 3098 Chinese words and 2714 English words.

We rerun GIZA using the same option as the previous but a Chinese/English dictionary. The perplexity in this case was also shown in fig. 8.

From the results we observed the minimum perplexity with the dictionary was around 210, whereas the minimum perplexity without the dictionary was 227. Perplexity reduced 7.8%. In addition, when the alignment with a dictionary was compared with the alignment without dictionary, it looked more satisfactory. So we decided to choose this alignment as the training data of building our N-gram translation templates. More specifically, we chose the alignment with the minimum perplexity, which was the second iteration of Model 2.

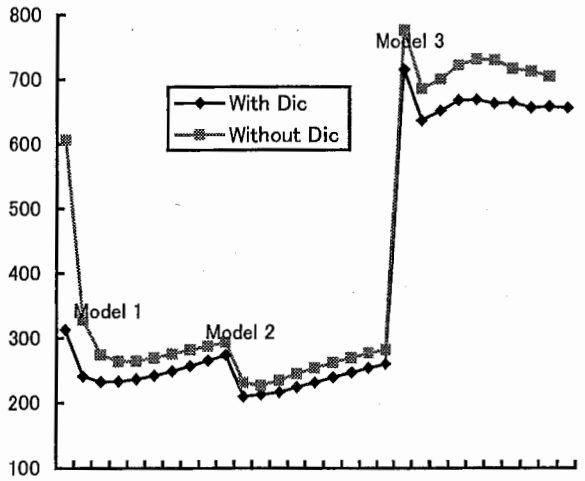

Figure 8: perplexity as iteration times

We made the following rules to get N-gram translation templates.

(1) First build unigram translation templates. As to the alignment output by GIZA, each source word aligned to multiple target words, see fig.5. We searched an adjacent word sequences in the multiple word sequences and regarded it as an unigram translation template if this adjacent word sequences contained at least one word that had been defined in the

3.4.2. Model Building

Released by CLSP/JHU, GIZA is a program for training IBM's model from bilingual data. It uses EM method to iteratively train IBM Model 3 described in [12]. During the training of Model 3, it passes Model 1 and Model 2. For detail description of GIZA, please refer to [20].

	Chinese	English
TRAIN Sentences	18922	
Words	178,443	196,263
Vocabulary	4736	6078
TEST Sentences	200	
Words	1755	1893

Table 8: a Chinese/English database

In this experiment we were only interested in getting the Viterbi allignment of bitexts, which is one of the results output by GIZA. In the option we set for running GIZA, the iteration times for each model (1/2/3) is 10. Fig. 7 shows the perplexity of test data after each iteration.

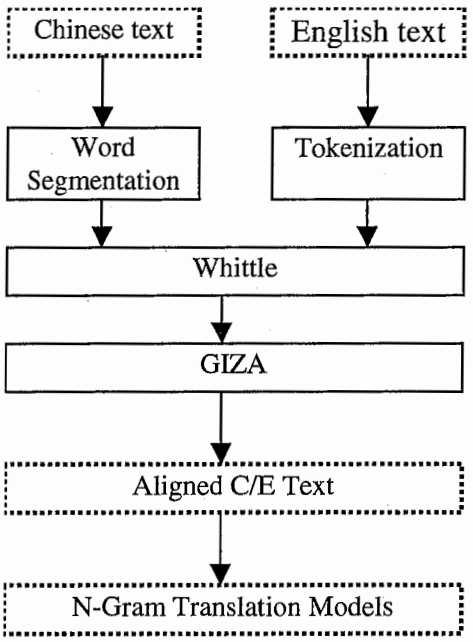

Figure 7: Training framework of N-gram Translation Model

Our aim was to get a better alignment of bilingual data for building N-gram translation model. But the alignment results given in the previous

experiment were not good visually. This alignment was made by GIZA without a bilingual dictionary. Our next experiment was made with a dictionary. We made a Chinese/English dictionary by the following steps. First, we applied GIZA to align Chinese/English without a dictionary. Second, from this alignment take out source words and its target matched words. Each pair of source and target is an item of an initial dictionary. Third, manually remove from the initial dictionary those items where source and target translation were wrong, resulting the final Chinese/English dictionary. As to our experiments, the initial dictionary consisted of 38,000 items chosen from the alignment without a dictionary. After manual work we got a final dictionary containing 4879 items, 3098 Chinese words and 2714 English words.

We rerun GIZA using the same option as the previous but a Chinese/English dictionary. The perplexity in this case was also shown in fig. 8.

From the results we observed the minimum perplexity with the dictionary was around 210, whereas the minimum perplexity without the dictionary was 227. Perplexity reduced 7.8%. In addition, when the alignment with a dictionary was compared with the alignment without dictionary, it looked more satisfactory. So we decided to choose this alignment as the training data of building our N-gram translation templates. More specifically, we chose the alignment with the minimum perplexity, which was the second iteration of Model 2.

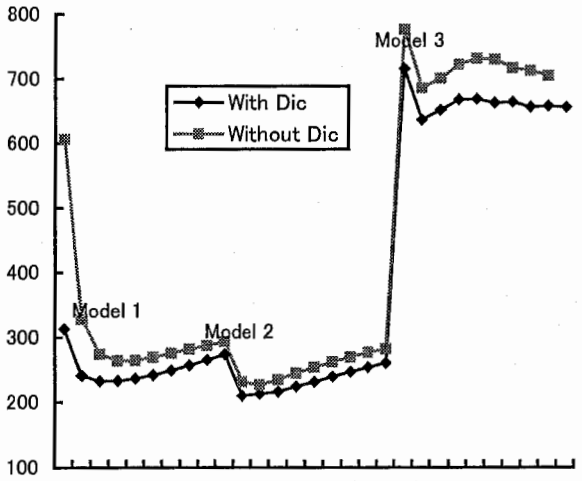

Figure 8: perplexity as iteration times

We made the following rules to get N-gram translation templates.

(1) First build unigram translation templates. As to the alignment output by GIZA, each source word aligned to multiple target words, see fig.5. We searched an adjacent word sequences in the multiple word sequences and regarded it as an unigram translation template if this adjacent word sequences contained at least one word that had been defined in the

dictionary as a translation of the source word. If more than one adjacent word sequence satisfied such condition, all these adjacent word sequences connected by an empty slot inserted artificially was regarded as a unigram translation template. The empty slots separated the adjacent word sequences.

(2) If we had unigram translation template, just combined N unigram translation template according to the relative position in the aligned target sentences to get N-gram translation templates.

(3) Using maximum likelihood methods to get statistics of N-gram translation templates distribution.

Using the rules described above, we built bigram translation templates based on the alignment results of the second iteration of Model 2 of GIZA. Table 9 shows some numbers of the built bigram translation templates and source language model constraints. Compared with the source language model constraints, the bigram translation templates covered 70% unigram and 22% bigram of source language. These numbers were not high. But we can improve it by introducing POS tagging or using more training data. Some examples of these bigram translation templates were shown in Table 10. This bigram translations templates included both unigram(多少) and bigram(多少/钱) templates.

	Source (>1)		Target	(S,T)>1
	Unigram	Bigram		
Bigram templates	2213	3936	7670	39663
Languag e model	3278	17946	--	--

Table 9: Statistics of bigram templates and LM

Source	Target	P(t|s)
多少	How much	0.5224
多少 钱	How much	0.3561
多少 钱	How much is # charge	0.05481
多少 钱	How much is it	0.2055
多少 钱	How much is # fare	0.04110

Table 10: Examples of N-gram translation templates

It is very desirable to present the translation results using our proposed N-gram translation models. But till the last minute of submitting this paper, we have not finished this work. We would like to offer a perplexity evaluation of our N-gram translation models. We used the same resource as GIZA to calculate the perplexity. When we used only unigram templates, perplexity is 220. When we used bigram templates, the value reduced to 142.

The perplexity reduced more than 35%. This result is very exciting. It indicates our N-gram models works well in combining detailed contextual information.

3.5. CONCLUSION

In this paper we show readers our insight on building N-gram translation models for machine translation. We introduced new concepts such as N-gram translation templates and described the approach of using the alignment output by GIZA to build N-gram translation templates and combined with N-gram language models into translation. We reported the experiment results of using GIZA in aligning travel-related data. A perplexity evaluation was made to support our ideas of N-gram translation models. We will report the translation results in the workshop or report it in a follow-up paper.

4. REFERENCES

1. Ratnaparkhi. A maximum entropy part-of-speech tagger. In Proceedings of the Empirical Methods in Natural Language Processing Conference, 1996.
2. E.Black, S.Eubank, H.Kashioka, R.Garside, G.Leech, and D.Magerman. Beyond skeleton parsing: producing a comprehensive large—scale general--english treebank with full grammatical analysis.In Proceedings of the 16th Annual Conference on Computational Linguistics, pages 107--112, 1996.
3. E.Black, S.Eubank, H.Kashioka, and J.Saia. Reinventing part-of-speech tagging. Journal of Natural Language Processing (Japan), 5(1), 1998.
4. E.Brill. Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging Computational Linguistics, 1995.
5. R.Lau, R.Rosenfeld, and S.Roukos. Trigger-based language models: a maximum entropy approach. In Proceedings of the International Conference on Acoustics Speech and Signal Processing, pages II:45--48, 1993.
6. B.Merialdo. Tagging english text with a probabilistic model. Computational Linguistics, 20(2):155--172, 1994.
7. S.Della Pietra, V.Della Pietra, and J.Lafferty. Inducing features of random fields. IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(4):380--393, 1997.
8. R.Rosenfeld. A maximum entropy approach to adaptive statistical language modelling. Computer Speech and Language, 10:187--228, 1996.
9. R.Zhang, E.Black, A.Finch, and Y.Sagisaka. Integrating detailed information into a language model.In Proceedings of ICASSP'2000}, pages III--1595--1598, Istanbul, Turkey, June 2000.
10. Jun Wu and S.Khudanpur. Syntactic heads in statistical language modeling. In Proceedings of ICASSP'2000, pages III--1699--1702, Istanbul, Turkey, June 2000.

11. R.Zhang, E.Black, and A.Finch. Using detailed linguistic structure in language modelling. In Eurospeech'99, pages 1815--1817, 1996.

12. P.F.Brown, J.Cocke, S.Della Pietra, V.Della Pietra, F.Jelinek, J.Lafferty, R.Mercer, and P.Roosin(1990), A Statistical approach to machine translation. Computational Linguistics, 16, 79-85

13. P. F. Brown and V. J. Della Pietra and S. A. Della Pietra and R. L. Mercer. The Mathematics of Statistical Machine Translation: Parameter Estimation. Computational Linguistics, vol. 19, 2, 1993

14. F. J. Och and H. Weber. Improving Statistical Natural Language Translation by Categories and Rules. Proc. of the 17th Int. Conf. on Computational Linguistics, 1998

15. F. J. Och and C. Tillmann and H. Ney. Improved Alignment Models for Statistical Machine Translation. Proc. of the Joint SIGDAT Conf. On Empirical Methods in Natural Language Processing and Very Large Corpora, 1999

16. A. Berger and P. Brown and S. Della-Pietra and V. Della-Pietra and J. Gillett and J. Lafferty and R. Mercer and H. Printz and L. Ures. The {Candide} System for Machine Translation, proceedings of the ARPA Human Language Technology Workshop, 1994

17. H. Alshawi and A. Buchsbaum and F. Xia, A Comparison of Head Transducers and Transfer for a Limited Domain Translation Applications, Proc. ACL, 1997

18. I. Dagan and K. Church and W. Gale, Robust Word Alignment for Machine Aided Translatio, Proceedings of the Workshop on Very Large Corpora: Academic and Industrial Perspectives, 1993

19. I. Dan Melamed, Automatic Discovery of Non-Compositional Compounds, Proceedings of the Second Conference on Empirical Methods in Natural Language Processing, 1997

20. Y.Y.Wang and A.Waibel. Decoding Algorithm in Statistical Machine Translation. Proc. Of COLING-ACL, 1998

21. EGYPT. A Statistical Machine Translation Toolkit. Released by JHU/CLSP. Available at http://www.clsp.jhu.edu/ws99/projects/mt/index.html