A SIMPLIFIED APPROACH TO WORD ALIGNMENT ALGORITHM FOR ENGLISH-TAMIL TRANSLATION

R. Harshawardhan [1], Mridula Sara Augustine [2]
Department of CEN, Amrita Vishwa Vidyapeetham,
Ettimadai, Coimbatore- 641105

Dr K. P. Soman [3]
Professor/HOD
Department of CEN
Amrita Vishwa Vidyapeetham, Ettimadai, Coimbatore- 641105

Abstract

In this paper, a recently proposed word alignment algorithm is simplified for easy understanding and tested for an Indian language. The word alignment problem is viewed as a simple assignment problem and is formulated as an Integer Linear Programming problem. The new objective function defined is tested for obtaining optimal alignment for English-Tamil translation pair. This alignment is necessary for creating the probabilistic bilingual dictionary and is also required for automatic machine translation. We have used this objective function to align words in 25 sentences of English-Tamil parallel corpora. The formulation is solved using the open source LP-Solver. Result obtained indicates that the methodology is applicable for all Indian languages. The work implemented is useful for pedagogical purposes, as it is a standard problem in computational linguistics. Accuracy of modern statistical machine translation depends on good word alignment. The document of the formulated model is available on request.

Keywords: Word alignment, assignment problem, integer linear programming, alignment variables and dictionary variables.

1. Introduction

Word alignment serves for many purposes. The standard approach to word alignment is to construct models from aligned sentences [Taskar, *et al*. (2005)]. For example in machine translation, the programs extract translation rules from word-aligned corpora and also the word alignment exploit dictionaries derived by alignment. Many methods of automatic word alignment have been proposed, especially for statistical machine translation (SMT) [Och and Ney (2004)]. Probabilistic generative models like IBM 1-5 [Guggilla and Singh, (2007)], HMM [Vogel, *et al*. (1996)], ITG, and LEAF [Fraser and Marcu, (2007)] define formulae for P (f | e) or P (e, f), with hidden alignment variables. EM algorithms estimate dictionary and other probabilities in order to maximize those quantities. One can then ask for Viterbi alignments that maximize P (alignment | e, f). EMD training combines generative and discriminative elements. Poor accuracy is a weakness for all these systems. Most translators still use 1990s algorithms to align their data. They have reasonable results for the translations of European languages but when it comes to any Indian language like Tamil, then numerous hurdles arises with these algorithms. Therefore we have to improve the accuracy of the word alignment algorithm in the translation of English sentences to Tamil sentences. Such an algorithm is described in [Bodrumlu, *et al*. (2009)]. We describe a simplified version of the same algorithm and show how it is applied for Indian Language.

2. Simplified objective function

The formulation of a word alignment problem as an Integer Linear Programming results in a good word alignment. This algorithm can be used for the translations of any language and in particular, Aryan and Dravidian languages .We have implemented this algorithm for English –Tamil translation pair. The beauty of the algorithm is that it is based on the common sense of human beings. The algorithm is well explained in the following example.

1. ***I came – naan vanthaen***
2. ***I went – naan ponaen***

When these translation-pairs of sentences are shown to a child, let us see how the child infers the alignment (translation) of words? On looking at the first sentence, the child start thinking that either *'I'* is assigned to *'naan'* or *'vanthaen'* and *'came'* is assigned to either *'naan'* or *'vanthaen'*. On looking at the second sentence, after reading the first one, the child immediately surmises that *'I'* has occurred in both the sentences and the

word *'naan'* has also occurred in both the translated sentences. So the child concludes that the translation pair of the source word *'I'* should be *'naan'* in the target language. Then the next word *'went'* has to be *'ponaen'*. Looking back at the previous sentences *'I'* should be *'naan'* and *'came'* is *'vanthaen'*. The answer is a perfect translation! But how can we model this problem mathematically, though it looks nothing more than common sense. Our human brain follows an algorithm with its own objective function and constraints. If we could possibly apply the same algorithm to a word alignment problem then we can get better results compared to other algorithms like Expectation maximization (EM) etc. In [Bodrumlu, *et al*. (2009)], the word alignment problem is formulated as a following linear program. The formulation of the objective function is such that in different sentences, as far as possible same source word is aligned to same target word if available. This new objective function forces the commonsensical solution for the word alignment problem.

Let *'i'* denote sentence number in the corpus, *'j'* denote index of target word in corpus and *'k'* denote index of source word in the corpus. *Let align(i,j,k)* denote a binary variable representing alignment of *j'th* target word to *k'th* source word in *i'th* sentence. So alignment problem can be viewed as an assignment problem. However we have many sentences and therefore many assignment problems. Now our task is to find a single objective function that connects or combines all assignment problems. This is achieved through the introduction of extra dictionary variables which are again binary. *dictionary(j,k)* denotes whether *j'th* word of target language is an alignment of *k'th* word in source language. This takes a value 1 if this happens in at least one sentence pair. Inclusion of dictionary variables *dictionary(j,k)* is the most innovative aspect in the new formulation.
The objective function is defined as:

$$min \sum_{j,k} dictionary(j,k) \quad (1)$$

Subject to the constraints:

$$\forall i,j \sum_{k} align(i,j,k) = 1 \quad (2)$$

$$\forall i,k \sum_{j} align(i,j,k) \leq 1 \quad (3)$$

$$\forall i,j,k align(i,j,k) \leq dictionary(j,k) \quad (4)$$

The objective function is minimization of summation of all dictionary variable values. The first constraint "Eq. (2)" says that every word of the target language (i.e., Tamil) is aligned to any one of the words of the source language (i.e., English). This ensures that each word in given Tamil sentence is associated with at least one word in the corresponding English sentence. The second constraint, "Eq. (3)" says that all words in a given English sentence are not necessarily aligned to some words of corresponding Tamil sentence. Because sometimes there are no specific translations for certain words of English such as auxiliary verbs. The third constraint, "Eq. (4)" says that the alignment variables *align(i,j,k)*are linked to a dictionary variable *dictionary(j,k).* It must be noted that every unique word in a language is given a unique index. This index is used in naming the alignment variables. The whole algorithm is based on the repetition of the words in various sentences. For getting minimum value for objective function, the algorithm must repeatedly align a repeating word in source sentences to a single target word in the corresponding target sentences so that only one dictionary variable assumes a value of *'1'*. It should align to a different target word if and only if the repeated target word is not present in the target sentence under consideration. A small corpus of sentences with repeated words is enough to test this algorithm.

3. LP - Formulation

The corpus used for testing the algorithm is shown as follows.
i. i came
naan vanthaen
ii. he came
avan vanthaan

iii. i went
naan ponaen
iv. he went
avan ponaan
v. i am good
naan nallavan
vi. he is good
avan nallavan
vii. i like her
naan avalai virumbugiraen
viii. he likes me
avan ennai virumbugiraan
ix. i love you
naan unnai kadhalikkiraen
x. he loves me
avan ennai kadhalikkiraan
xi. we love him
naam avanai virumbugirom
xii. she loves him
aval avanai kadhalikkiraal
xiii. she likes all
aval anaivaraiyum virumbugiraal
xiv. all like me
anaivarum ennai virumbuvar
xv. he is better than me
avan ennai vidanallavan
xvi. she is better than him
aval avanai vidanallaval
xvii. all love nature
anaivarum iyarkaiyai virumbuvar
xviii. he loves nature
avan iyarkaiyai virumbugiraan
xix. i love poem
naan kavithaiya ivirumbugiraen
xx. poet writes poem
kavignarkavithai yaiezhuthuvar
xxi. i am a poet
naan oru kavignar
xxii. she loves a poet
aval oru kavignarai kadhalikkiraal
xxiii. poet loves nature
kavignariyar kaiyai virumbuvaar
xxiv. he is a good poet
avar oru nalla kavignar
xxv. i told you
naan unnidam koorinaen

The corpus is made in such a way that the words are repeated in many sentences. The unique words of both the languages are extracted from the corpus and they are indexed as given in the Fig.1.The alignment algorithm is modeled as an assignment problem. *"In a standard assignment problem, 'm' work has to be assigned to 'n' persons, such that one work is assigned to exactly one person, and assignment should be such that the cost incurred is minimum, assuming that* $m \geq n$*."* Again note that, in literature, the assignment problem is formulated as an Integer Linear Programming problem.

Let us frame the word alignment model as an assignment problem by considering an example of two sentences from the corpus. Also we could see how the constraints are formed and the objective function is created. Here, Tamil words (whose index is *j*'s*)* are taken in rows, English words (*k*'s*)* are in columns and they are arranged according to their index numbers.

The word alignment model for sentence 1 is given in Table 1,

Table 1. Alignment model for first sentence.

i=1	*k→*	*10*	*5*
j↓		*i*	*came*
19	*naan*	*align(1,19,10)*	*align(1,19,5)*
29	*vanthaen*	*align(1,29,10)*	*align(1,29,5)*

Constraints created for sentence 1:

$$align(1,19,10) + align(1,19,5) = 1 \quad (5)$$

$$align(1,29,10) + align(1,29,5) = 1 \quad (6)$$

$$align(1,19,10) + align(1,29,10) \leq 1 \quad (7)$$

$$align(1,19,5) + align(1,29,5) \leq 1 \quad (8)$$

$$align(1,19,10) \leq dictionary(19,10) \quad (9)$$

$$align(1,19,5) \leq dictionary(19,5) \quad (10)$$

$$align(1,29,10) \leq dictionary(29,10) \quad (11)$$

$$align(1,29,5) \leq dictionary(29,5) \quad (12)$$

The word alignment model for sentence 3 is given in Table 2,

Table 2. Alignment model for third sentence.

i=3	*k→*	*10*	*24*
j↓		*i*	*went*
19	*naan*	*align(3,19,10)*	*align(3,19,24)*
25	*ponaen*	*align(3,25,10)*	*align(3,25,24)*

Constraints created for sentence 3:

$$align(3,19,10) + align(3,19,24) = 1 \quad (13)$$

$$align(3,25,10) + align(3,25,24) = 1 \quad (14)$$

$$align(3,19,10) + align(3,25,10) \leq 1 \quad (15)$$

$$align(3,19,24) + align(3,25,24) \leq 1 \quad (16)$$

$$align(3,19,10) \leq dictionary(19,10) \quad (17)$$

$$align(3,19,24) \leq dictionary(19,24) \quad (18)$$

$$align(3,25,10) \leq dictionary(25,10) \quad (19)$$

$$align(3,25,24) \leq dictionary(25,24) \quad (20)$$

Here, *align (i,j,k)*are alignment variables and *dictionary(j,k)* are dictionary variables. From the previous example, we observe that for each sentence, there is a separate assignment problem. For each assignment problem, we have separate variables. It is clear from the example that in both the sentences, the aligning of word *'i'* to *'naan'* has unique *alignment variable* for each sentence (i.e.,) *align(1,19,10)* and *align(3,19,10).* These variables differently in the sentence number index. Suppose the alignment of *'i'* to *'naan'* occurs in *'n'*

sentences of the corpus, then, we have *'n'* alignment variables for this single word alignment. Corresponding to these *'n'* alignments, we have one single dictionary variable. This dictionary variable is independent of the sentence number (*index 'i'*). Therefore we are indirectly forcing alignment of *'i'* to *'naan'* maximum number of times. The objective function is optimized by minimizing the summation of all dictionary variable values of the sentences. We are minimizing the objective function because the same source word *'I'* should be aligned to one target word' *naan'* in most of the sentences where it occurs. *'i'* in a sentence should be aligned to some other word if and only if *'naan'* does not occur in the corresponding targets sentence.

4. Results

The dictionary table created is shown in the Fig. 1, where the values of *'1'* indicate the word alignment of translation pairs. The total number of sentences in the corpus is 25. The total number of unique words of English is 26. The total number of unique words of Tamil is 36. We convert word alignment problem into a linear program. We have used the LP Solver to solve this problem. The objective is to minimize *dictionary* variables subjected to some constraints. The constraints correspond to *aligning* variables and *dictionary* variables. For our problem with 25 sentences there are 636 constraints and 1183 variables. The time to load data was 0.110 seconds; pre-solve used 0.014 seconds, 0.121 seconds in simplex solver, in total it took 0.245 seconds. The optimal solution obtained is 40 after 557 iterations.

5. Applications

One important application of word alignment is in Translation Memory for building probabilistic dictionary. A translation memory consists of text segments in a source language and their translations into one or more target language stored in such a way that we can retrieve similar sentences for a given sentence in source language or target language. These segments can be blocks, paragraphs, sentences, or phrases. It is an invaluable tool for manual translation. Similar model can also be developed for making dependency parser based on Paninian Grammar [Bharati, *et al*. (1995)].

6. Conclusion

Good word alignments in Statistical machine translation yield good translation accuracy. The new objective function for word alignment given in [Bodrumlu, *et al*. (2009)] is simplified for easy understanding and implemented using open source LP-Solver for English-Tamil word alignment problem. The model developed can be used as a good demonstration tool. The LP formulated model is available on request.

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
		a	all	am	better	came	good	he	her	him	i	is	like	likes	love	loves	me	nature	poem	poet	she	than	told	we	went	writes	you
1	anaivaraiyum	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	anaivarum	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
3	aval	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
4	avalai	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	avan	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	avanai	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
7	avar	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	ennai	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
9	ezhuthuvar	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
10	iyarkaiyai	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
11	kadhalikkiraal	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
12	kadhalikkiraan	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
13	kadhalikkiraen	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	kavignar	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
15	kavignarai	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
16	kavithaiyai	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
17	koorinaen	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
18	naam	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
19	naan	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	nalla	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	nallaval	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	nallavan	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	oru	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	ponaan	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
25	ponaen	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
26	unnai	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	unnidam	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
28	vanthaan	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	vanthaen	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	vida	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	virumbugiraal	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32	virumbugiraan	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
33	virumbugiraen	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
34	virumbugirom	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
35	virumbuvaar	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
36	virumbuvar	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 1. Dictionary table showing the alignment of words

7. Future work

The future works include, reformulation of constraints in LP for many to many word alignments; implementation of the problem in faster and reliable GPU environment, with provision for larger number of constraints and variables, to get better results with good accuracy; we have to test the algorithm for large bilingual corpora; the model can be extended to multilingual word alignment; the same linear program can be solved in Excel; the word alignment model can be applied for phrase based translations [Zhang, *et al*. (2003)]; the word alignment model can be created using quadratic assignment [Julien, *et al*. (2006)].

References

[1] Akshar Bharati, Vineet Chaitanya, Rajeev Sangal. (1995): "Natural Language Processing: A Paninian Perspective", Publisher: Prentice Hall of India.

[2] Alexander Fraser and Daniel Marcu. (2007): "Getting the structure right for word alignment: LEAF", in Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 51–60.

[3] Ben Taskar, Simon Lacoste-Julien, Dan Klein. (2005): "A discriminative matching approach to word alignment", in Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP), pages 73–80, Vancouver.

[4] Chinnappa Guggilla and Anil Kumar Singh. (2007): "A Java Implementation of an Extended Word Alignment Algorithm Based on the IBM Models", In Proceedings of the 3rd Indian International Conference on Artificial Intelligence, Pune, India.

[5] Dan Melamed, I. (1997): "A word-to-word model of translational equivalence", in Proc. ACL .

[6] Franz Josef Och and Hermann Ney. (2004): "The alignment template approach to statistical machine translation", Computational Linguistics, 30(4).

[7] Simon Lacoste-Julien, Ben Taskar, Dan Klein, and Michael I. Jordan. (2006): "Word alignment via quadratic assignment", in Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL, pages 112–119,New York.

[8] Stephen Vogel, Hermann Ney, and Christoph Tillmann. (1996): “HMM-based word alignment in statistical translation”, in Proc. ACL.
[9] Tugba Bodrumlu, Kevin Knight, Sujith Ravi. (2009): “A New Objective Function for Word Alignment”.
[10] Ying Zhang, Stephan Vogel, Alex Waibel. (2003): “Integrated phrase segmentation and alignment algorithm for statistical machine translation”, in Proc. Intl. Conf. on NLP and KE.