

CS224n

Final Project Abstract

THE INTERLEAVING TRANSLATION GROUP
Francis Bond, Roger Levy,
Kyonghee Paik, Colleen Richey, and Ana Paula Quirino Simões

September 18, 2000

Abstract

We propose to use *inversion transduction grammars* (ITGs) for automatic constituency bracketing and identification of phrasal equivalents in bilingual, sententially-aligned Japanese-English corpus and German-English corpora. ITGs were originally developed by Wu (1995b,d,a,c), who tested them on an English-Chinese aligned corpus. An ITG consists of a binary context-free grammar generating output on two streams, but which allows concatenation order (left-right or right-left) for each phrase structure production rule to be specified independently for each stream. This facilitates the matching of corresponding phrasal constituents within sentence-aligned corpora.

Wu (1995b,a) developed a “grammarless” ITG for phrasal bracketing of parallel aligned sentences, which made use of the simplifying assumption of only one nonterminal category and equal probability for all non-terminal rewrite rules. We implement such an ITG for Japanese/English and German/English corpora, and evaluate the quality of bracketing outputs.

Wu used a translation lexicon automatically learned from collocations, but noted that adapting a translation lexicon pre-built for other purposes might yield better results. For German/English, we generated a translation lexicon using methods similar to Wu’s, and for Japanese/English, we used a commercially-used translation lexicon.

Wu tested the performance of his model by taking a random sample of phrasal translation pairs and evaluating them for their accuracy. We plan to use a similar procedure for evaluation, and we also hope to devise other means of testing ITG performance.

1 Background

With the increasing availability of large parallel corpora, statistical tools for parallel corpus analysis are increasingly important. Automatic sentence alignment, although not trivial, is already feasible to a reasonable degree of precision. The automatic alignment of sub-sentential structures, however, is less well-developed. Wu (1995b,d,a,c) developed the *Inversion Transduction Grammar* formalism (ITG) to model parallel sentence

production in two languages. An ITG production corresponding to a sentence pair in a parallel aligned corpus includes an associated *oriented bracketing* that captures not only sub-sentential structure but variation in constituent ordering between the two languages. Wu (1995c) overviews a variety of this formalism’s potential applications. In this project, we implement one of these applications, a *grammarless bracketing algorithm*, which automatically produces phrasal bracketings for the sentence pair given only a translation lexicon. We implemented this algorithm in two language pairs: German/English, using an automatically learned translation lexicon, and Japanese/English, using a translation lexicon designed for a (non-statistical) MT system.

Note that in this writeup, all bracketed parallel productions are given as English/English translation, for simplicity’s sake.

2 Brief introduction to ITG formalism

An ITG is a slightly modified context-free phrase structure grammar. It consists of a single *start node*, a finite number of *nodes* and *rewrite rules*, as does a context-free phrase-structure grammar. Furthermore, the ITG we implemented is that every rule is either a *terminal* or a *non-terminal* rewrite: that is, either every daughter of a rewrite rule is a *terminal* node, or every daughter is a *non-terminal* node. Unlike a CFG, however, every non-terminal rewrite rule has an associated *orientation–straight* or *inverted*—and every terminal node is a *pair* of symbols, representable as a/b , where a is the output in Language 1 and b the output in Language 2. *Singleton* terminals where output is empty in Language 1 or in Language 2 are allowed (represented by (ϵ/y) and (x/ϵ) respectively), but null terminals where output is empty in both languages is not.

A *stochastic* ITG is simply one where each rewrite is associated with a probability. In Section 6 we describe our implementation of Wu’s algorithm to find the optimal parse of a sentence pair given a stochastic ITG.

When a part-of-speech categorization of both languages is not readily available, the ITG can be generalized by using only one nonterminal category. Such an ITG should serve as a highly general tool for bracketing parallel corpora given no information about the grammar of the languages. We tested such an ITG on our German/English and Japanese/English corpora.

3 The Japanese-English data files

We prepared three sets of data files (in the directory `jp-en-txt`). The first, made from sentences taken from Japanese Newspapers were rejected as the average sentence length was too long (27 words) so the algorithm took too long.

The main set used for testing was example sentences from a Japanese-English dictionary (Gakken). All sentence pairs with each sentence less than 70 characters in length were combined into the file `kihon.euc`. The Japanese was split into separate words using a development version of NTT’s morphological analyzer ALTJAWS <http://www.kecl.ntt.co.jp/icl/mtg/resources/altjaws.html>. If the analyser re-

turned a canonical form for a word (the uninflected form used as an index in NTT’s lexicon Ikehara et al. (1997)), it was appended to the word separated by an underbar: *itta* “went” becomes *it_iku* “go” and *ta* “past”.

We also tested on the Ikehara MT test suite Ikehara and Shirai (1990), converted to `kinou.lis`.

42 sentences from the test suite, from the section dealing with verb valence, were hand parsed as a gold standard (by Kyonghee Paik). The sentences in the input format are `test-data.euc`. The hand parsed sentences are `test-good.euc`.

4 The Japanese-English lexicon

5 German-English lexicon

For the bracketing algorithm to be run on the German/English data set, a lexicon had to be built which would return the translation probability for each given bilingual word pair within the given corpus. The corpus used is constituted of dialogues extracted out of the **VERBMOBIL** corpus, containing dialogues aligned by speech turns.

The lexicon was built applying a variant of the EM-algorithm described in Brown et al. (1993)¹. A variant of this model was used by Wu and Xia to induced the Chinese/English lexicon for the bracketing algorithm, as decribed in Wu and Xia (1994), Wu and Fung (1994).

5.1 The bilingual training procedure

The output of the algorithm to induce the lexicon is a table listing all possible pairs of words (where the first word is in the first language, and the second word is in the second language) extracted out of the training corpus and the probabilities estimates for translating the first word into the second one.

The basic model of Brown et al. (1993) assumes that every string \mathbf{g} of the second language (here German) is a possible translation of \mathbf{e} , a given string in the first language (here English). To every pair of strings (\mathbf{e}, \mathbf{g}) , a number $Pr(\mathbf{g}|\mathbf{e})$ will be associated, which is the probability that a translator, when given \mathbf{e} as input, will produce \mathbf{g} as its translation. Given this, the best translation of any \mathbf{e} can be found choosing the string $\hat{\mathbf{e}}$ for which $Pr(\mathbf{g}|\mathbf{e})$ is greatest. Using Bayes’ theorem, we find that:

$$Pr(\mathbf{g}|\mathbf{e}) = \frac{\mathbf{Pr}(\mathbf{e})\mathbf{Pr}(\mathbf{e}|\mathbf{g})}{\mathbf{Pr}(\mathbf{g})} \quad (1)$$

Since the denominator is independent of \mathbf{e} , finding $\hat{\mathbf{e}}$ is the same as finding \mathbf{e} so as to make the product $\mathbf{Pr}(\mathbf{e})\mathbf{Pr}(\mathbf{e}|\mathbf{g})$ as large as possible. This brings us then to the *Fundamental Equation of Machine Translation*:

$$\mathbf{e} = \underset{e}{argmax} Pr(\mathbf{e})\mathbf{Pr}(\mathbf{e}|\mathbf{g}) \quad (2)$$

¹We implemented here a variant of the first model described in the paper.

$Pr(\mathbf{e})$ is just a simple count of the number of occurrences of the given string in the corpus, $Pr(\mathbf{e}|\mathbf{g})$ is an enormous table which associates every possible pair of words in the corpus analysed to a number between 0 and 1. Since counting the occurrences of a word is a trivial task, in the following we will concentrate on finding the conditional probabilities $Pr(\mathbf{e}|\mathbf{g})$ for our lexicon.

If we take m to be the length of the German sentence \mathbf{g} and l the length of the English sentence \mathbf{e} , the basic model proposed by Brown et al. (1993) assumes that the probability of translation for \mathbf{e} into \mathbf{g} following a particular word alignment \mathbf{a} between both sentences can be approximated by:

$$Pr(\mathbf{g}, \mathbf{a}|\mathbf{e}) = \frac{\epsilon}{(l+1)^m} \prod_{j=1}^m t(c_j|e_{a_j}) \quad (3)$$

where $t(c_j|e_{a_j})$ are translation probabilities for individual word pairs and ϵ is a small constant.

Under this assumption, the expected number of times that a given English word e in a training English sentence \mathbf{e} generates a particular word g in a given German sentence \mathbf{g} can be estimated using the following equation:

$$c(g|e; \mathbf{g}, \mathbf{e}) = \frac{t(g|e)}{t(g|e_0) + \dots + t(g|e_l)} \sum_{j=1}^m \delta(g, g_j) \sum_{i=0}^l \delta(e, e_i) \quad (4)$$

where $\delta(x, y)$ is the *Kronecker delta function*, which equals to one when both of its arguments are the same and otherwise equals to zero.

The translation probabilities for any given pair of words are given by:

$$t(g|e) = \lambda_e^{-1} \sum_{\mathbf{g}, \mathbf{e} \text{ in corpus}} c(g|e; \mathbf{g}, \mathbf{e}) \quad (5)$$

where:

$$\lambda_e = \sum_g \sum_{\mathbf{g}, \mathbf{e} \text{ in corpus}} c(g|e; \mathbf{g}, \mathbf{e}) \quad (6)$$

5.2 This is how the EM algorithm is supposed to work...

The training algorithm uses equations 4, 5 and 6 iteratively to re-estimate the values for $t(g|e)$, as follows:

1. First, consistent initial values for all $t(g|e)$ are chosen.
2. Compute the counts for all word translation pairs using equation 4, summing over all sentence pairs in the corpus.
3. Compute λ_e for each English word using equation 6.
4. Re-estimate the values for $t(g|e)$, this time using equation 5.

5. Repeat the three last steps until the probabilities converge.

The algorithm as described above was applied to a (fairly) small sample of the data, namely one dialogue extracted out of the corpus, containing 19 aligned sentences. The algorithm iterated 10 times (the number of iterations was set to be 10, no stopping criteria was defined for this test), and it produced reasonable results.

Unfortunately, we never let the algorithm described as above run on a representative sample of the data until it could terminate. The reason for that is the long training time for the algorithm - for the data from one out of 8 cdroms with biligual dialogues, it took the algorithm around 10 hours to run through its first iteration.

5.3 ...and this is how our lexicon was produced

In order to cope with the time problem, we decided to let the algorithm run only through one iteration. Although we are aware of the fact that the probabilities estimated this way would not reach the precision factors exposed in Wu and Xia (1994) and Wu and Fung (1994), we hoped to get an approximation good enough to let the bracketing algorithm find the correct mappings.

The algorithm was then implemented, with the counts being re-estimated once, and the translation probabilities are calculated after this single re-estimation. This version of the algorithm was applied to a representative sample of the data, with approximately 7500 sentence pairs. Table 1 shows some of our results.

The sample entries show that the algorithm mapped inflected forms, like the German determiners and the personal pronouns, correctly to the English translation, distributing high probabilities among good candidates, and assigning to bad pairs a significantly lower probability. Note that the algorithm could make the distinction between definite and indefinite determiners, mapping those correctly between the two languages. Words which were not so frequent within the corpus got high probabilities when mapped to *ich* and commas, since those are the most frequent tokens in the corpus.

Some optimizations of the actual converging algorithm are possible, as managing the access to the databases where the probabilities and the sentence pairs are stored in a faster way, so that the algorithm can converge to the correct results in about 24 hours, which is the time that Wu and Xia (1994) and Wu and Fung (1994) needed for the induction of their Chinese/English lexicon.

5.4 What else can be done in order to improve the lexicon induction

Wu and Xia (1994) and Wu and Fung (1994) describe some filtering criteria to get rid of translation entries with small or negligible probabilities. One of them is to set thresholds on probability, but in an intelligent way, since absolute thresholds work poorly for sparse data.

In order to cope with this problem and still prune the lexicon in an adequate way, they included only words which occur at least a certain amount of times in their lexicon,

unfortunately	leider	0.0158862729845391
unfortunately	ich	0.02647712164089819
unfortunately	sollte	3.54948390313868e-07
I	,	0.000278760996993383
I	ich	0.000119027970813151
I	wir	6.33160832563274e-06
you	Sie	0.000249177306955483
you	ich	9.63692041269916-05
he	er	1.29642633712784e-06
she	er	3.11035212241217e-06
he	ihn	1.29642633712784e-06
he	ihm	0.0100287700176989
she	sie	0.108273430949249
the	er	1.78710571537698e-12
the	der	1.35355717346005e-05
the	die	8.30178910063669e-05
the	das	3.02764109974647e-06
the	dem	1.90444979633192e-05
the	ein	1.83056307515398e-06
the	einer	1.78710971537698e-12
a	dem	1.2185198435789e-05
a	ein	0.000167543316561741

Table 1: Sample entries of the English/German lexicon induced

and retained only the translations accounting for the top 0.75 of the probability mass. Any translation with probability less than 0.11 was eliminated of their lexicon.

None of these pruning criteria was applied here. Given the small data used to induce the lexicon, no pruning criteria of this sort would produce a lexicon of useful size. It is also not clear that a pruned lexicon would serve the purposes of the bracketing algorithm, since for it to run, it requires a probability for each sentence pair it encounters, so that there is no reason to prune these probabilities out of the lexicon (reducing the storage space would be a big benefit with a bigger lexicon, though).

Here some factors which could improve the translation estimates for our German/English lexicon. Since German is a highly inflecting language, mapping word inflection classes (based on the stems, for instance) instead of the words themselves would probably produce much more consistent results. For this purpose, a mapping function would have to be defined, which would map each German word to its stem or uninflected form, ignoring the inflection endings.

Another factor which has had an enormous impact in our lexicon was the abusive usage of commas in the transcription of the **VERBMOBIL** corpus, which caused the probability of each pair containing one comma to be higher than any other combination of words. Finding a way to intelligently take the commas into account would probably diminish inconsistency in the lexicon produced. One way of achieving this would be to

get rid of all commas before inducing the lexicon. This wasn't realized here, since the bracketing algorithm uses the mapping $(, |,)$ as one of its criteria.

6 Details of bracketing ITG algorithm implementation

As mentioned above, the version of the ITG we implemented has only one non-terminal category. Furthermore, we implemented a binarized ITG in which there are only two non-terminal rewrite rules:

$$\begin{aligned} A &\rightarrow [A A] \\ A &\rightarrow \langle A A \rangle \end{aligned}$$

These are taken to have equal probability a . Note that we did not calculate true probabilities for these rewrite rules, nor did we adjust the probabilities in the translation lexica to account for the probability mass taken by the non-terminal rewrites. This has no effect on the results, however, except in the interaction between singleton translation probabilities, non-terminal rewrite probability, and unattested word-pair translation probability in determining the number of singletons in the optimal parse. We discuss this interaction in section 6.2.

Given such a stochastic ITG, it is evident that the optimal parse is that where the product of probabilities of all rewrites (including the translation probabilities) is highest. Wu (1995a) discusses the optimal-bracketing algorithm in greater detail, but we present a summary of the algorithm here.

This algorithm is similar to calculating PCFG parse probabilities through chart parsing. The highest-probability parse for each node is recursively calculated, starting at terminal nodes and working up toward the starting node. The probability of such a parse at a given node N can be written as δ_{stuv} , where st is the vertex coverage of N in Language 1 (from vertex s to vertex t) and uv the coverage in Language 2. Also associated with non-terminal nodes are an orientation θ_{stuv} , and Language 1 and Language 2 splits σ_{stuv} and ν_{stuv} for the two daughters. The steps to the algorithm are as follows.

First: initialize δ_{stuv} s for vertex coverage less than 2 in both languages. Note that vertex coverage of zero in both languages is ruled out by the prohibition on null productions. Following Wu, we assume that all L₁- or L₂-singleton probabilities are, equally, a very small constant.

Second: recursion on δ s, σ s, and ν s. It's important to recurse in the right way. In order to calculate δ_{stuv} , we already need to have $\delta_{ss'uv'}$, $\delta_{s'tuv'}$, $\delta_{ss'u'v}$, and $\delta_{s'tu'v}$ for all $s \leq s' \leq t$ and all $u \leq u' \leq v$. Note that in initialization, we calculate all δ s for $t - s \leq 1$ and $v - u \leq 1$. We need to systematically expand the $t - s$ and $v - u$ values little by little, working up. This is implemented in the program by using `for` loops over τ and ν values, which correspond to $t - s$ and $v - u$ values, respectively. The ensuing picture looks like this:

$\tau \parallel \nu$	1	2	...	$V-1$	V
1	$(T+1)(V+1) - 1[init]$	$\rightarrow (T+1)V$	$\rightarrow \dots \rightarrow$	$2(T+1)$	$\rightarrow T+1$
2	\downarrow $T(V+1)$	$\rightarrow \downarrow$ TV	$\rightarrow \dots \rightarrow$	\downarrow $2(T-1)$	$\rightarrow \downarrow$ $T-1$
...	\downarrow ...	\downarrow	\downarrow ...	\downarrow ...
$T-1$	\downarrow $2(V+1)$	$\rightarrow \downarrow$ $2V$	$\rightarrow \dots \rightarrow$	\downarrow 4	$\rightarrow \downarrow$ 2
T	\downarrow $V+1$	$\rightarrow \downarrow$ V	$\rightarrow \dots \rightarrow$	\downarrow 2	$\rightarrow \downarrow$ 1

Each entry in this matrix represents the number of δ s for each (τ, ν) pair. $[init]$ marks the initialization step, which calculates δ s for the $(\tau, \nu) = (1, 1)$ equivalent position.

Third: Reconstruction. Start with the root node and recursively travel down the daughter nodes. Since the coverage of each non-terminal node's optimal daughters is recorded during the calculation of optimal inside parses, this is a fairly simple step. During this process, we output the resulting bracketings.

6.1 Bracket Flattening

Every word alignment between sentences in two languages corresponds to at least one inversion transduction parse, but often more than one (see Wu (1995b) for details). For this reason, many of the bracketings of an entirely binary-branching ITG parse are often unnecessary structural commitments. Wu (1995a) deals with this problem through a bottom-up post-processing algorithm that takes advantage of the following word-alignment associativity properties of ITGs:

$$[A[BC]] \Leftrightarrow [[AB]C] \Leftrightarrow [ABC]$$

$$\langle A \langle BC \rangle \rangle \Leftrightarrow \langle \langle AB \rangle C \rangle \Leftrightarrow \langle ABC \rangle$$

We have implemented a similar algorithm that is top-down, however, and also further generalizes the associativity equivalences. We note that the following hold:

$$[A_1 \dots A_k [B_1 \dots B_k] A_{k+1} \dots A_n] \Leftrightarrow [A_1 \dots A_k B_1 \dots B_k A_{k+1} \dots A_n]$$

$$\langle A_1 \dots A_k \langle B_1 \dots B_k \rangle A_{k+1} \dots A_n \rangle \Leftrightarrow \langle A_1 \dots A_k B_1 \dots B_k A_{k+1} \dots A_n \rangle$$

These equivalences are easy to implement directly during the final output: brackets are only printed for a node if its orientation is not that of its mother. The result is an ITG of minimal structural commitment given the word alignment it encodes.

6.2 Interaction of parameters

Initial results revealed an important interaction for a pair of words (x, y) whose match is unattested in the translation lexicon, between the non-terminal rewrite probability

a , the singleton-production probabilities $b(\epsilon/y)$ and $b(x/\epsilon)$, and the unattested word-pair probability (call it $b_0 = b(x/y)$). Wu (1995a) appears to treat b_0 as equal to zero, although he does not explicitly state so; we treated b_0 as equal for all unattested pairs, but in one run we set $b(\epsilon/y) = (x/\epsilon) = 2(b_0^2) \ll 1$. a was set arbitrarily to 0.1; realistically, a should probably be closer to about 0.9, given the average length of sentences, but as a 's only role is in determining how many singletons will be produced, we were not too concerned. With this set of parameter values, our output included many spurious word matches, and in the small sample we examined, a sentence of E English and J Japanese tokens was parsed with only $|E - J|$ singletons, typically much lower than the number of singletons we expected for those sentences.

This result is less surprising than it appears. Our parse-optimizing algorithm does not permit the split of a node $\delta_{s-1stu-1u}$ into two singletons, but it does permit considerable flexibility in the split of, say, a node $\delta_{s-2stu-2u}$. For example, the node covering the English-English pair

...like dogs ...
 ...I like ...

assuming that *like* is matched between the two streams, can be parsed as \langle like/like dogs/I \rangle , or as $[$ e/I like/like dogs/e $]$ (or as another tree with two singletons and equivalent word order). The probability of the former parse is $ab(\textit{like}/\textit{like})b_0$ while the probability of the former is $a^2b(\textit{like}/\textit{like})b(\epsilon/y)b(x/\epsilon)$. Which of these will be chosen as the preferable parse, then, is dependent on the ratio of $\frac{b_0}{ab(\epsilon/y)b(x/\epsilon)}$. It is not certain that this ratio will *always* determine whether mismatched token pairs are mapped to each other or as singletons in output, but in our experimentation with the system, this always turned out to be the case. Note also that this is the *only* place in the algorithm where the probability a of nonterminal rewrite plays a role.

7 Results and Evaluation

See separately attached.

8 Discussion and Further Work

In this project, we only implemented a truly grammarless ITG. As a result, phrasal bracketing results were dependent only on the translation lexica we used. Furthermore, the ITG was not constrained to the use of a constituent structure consistent across parsed sentences. One of the major assumptions that ITGs operationalize is that binarized constituent structure is identical across languages. This was unfortunately not tested in our project. To conduct such a test would require that we implement a more sophisticated grammar with more non-terminal categories and more parameters for rewrite rule probabilities, which would be constrained to fit training data. As Japanese and German word order are considerably freer than in English, implementing such a grammar would likely give interesting results.

To do so would also require a more sophisticated translation lexicon that includes part of speech information. The automatically learned lexicon we used for the German/English corpus, for example, does not contain this information. On the other hand, detailed translation probabilities, which are lacking in many already-available non-statistically oriented translation lexica, are important for stochastic ITGs. In future work, it would be important to work on more closely integrating part-of-speech and probabilistic information to improve bracketing. Part-of-speech tagged bracketed output would also contain more information, such as the headedness of parallel aligned phrases, that is readily transferrable to other applications.

Finally, it is evident from our work that more sophisticated methods of dealing with the phrasal positioning of singletons is necessary, as singletons are highly underconstrained positionally. Wu (1995a) describes a postprocessing algorithm for “sinking” singletons to positions as structurally low as possible. Due to time constraints, we did not implement this algorithm, but we note that Wu’s argument that singleton-sinking is linguistically well-founded is on shaky ground. Wu notes that in Chinese/English alignment, many singletons will be function words in one language or another, and that these function words are generally structurally low. However, under many conceptions of syntax, functional categories are typically quite high in the phrase structure. An example given in Wu (1995a) is of the positioning of singleton “be/e” in the English/Chinese phrase

will be accountable to the Financial Secretary
will to Financial Secretary accountable

Wu’s singleton-sinking algorithm drops the singleton to the lowest possible phrase [be/e accountable/accountable]. However, it appears to us that the proper bracketing of this phrase is [will/will be/e < accountable/accountable [to/to the/e Financial/Financial Secretary/Secretary>]. On the other hand, some sentence pairs in our corpus included parenthetical material in one language that is completely omitted in the other language. In the parsing of such a sentence, the parenthetical material is not constrained to constituenthood. A more sophisticated part-of-speech system would likely address this, but we note that such a part-of-speech system needs to be quite sophisticated since parts of speech ultimately need to be assigned to word *pairs*, not single-language words. In closing, we note that the elucidation of an ITG category system is an important direction for further research to take, but that adapting existing phrase structure to a satisfactory ITG category system may be quite difficult.

References

- Brown, P. F., DellaPietra, S. A., DellaPietra, V. J., and Mercer, R. L. (1993). The mathematics of machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.
- Ikehara, S., Miyazaki, M., Shirai, S., Yokoo, A., Nakaiwa, H., Ogura, K., Ooyama, Y.,

- and Hayashi, Y. (1997). *Goi-Taikai — A Japanese Lexicon*. Iwanami Shoten, Tokyo. 5 volumes/CDROM.
- Ikehara, S. and Shirai, S. (1990). A function test system for Japanese to English machine translation. In *IEICE Technical Report NLC90-43*, pages 17–24. IEICE. (in Japanese).
- Wu, D. (1995a). An algorithm for simultaneously bracketing parallel texts by aligning words. In *ACL-95: 33rd Annual Meeting of the Assoc. for Computational Linguistics*, volume 33, pages 244–251. Cambridge, MA.
- Wu, D. (1995b). Grammarless extraction of phrasal translation examples from parallel texts. In *TMI-95, Sixth International Conference on Theoretical and Methodological Issues in Machine Translation*, volume 2, pages 354–372. Leuven, Belgium.
- Wu, D. (1995c). Stochastic inversion transduction grammars, with application to segmentation, bracketing, and alignment of parallel corpora. In *IJCAI-95: 14th Intl. Joint Conf. on Artificial Intelligence*, volume 14, pages 1328–1335. Montreal.
- Wu, D. (1995d). Trainable coarse bilingual grammars for parallel text bracketing. In *WVLC-3: 3rd Annual Workshop on Very Large Corpora*, volume 3, pages 69–82. Cambridge, MA.
- Wu, D. and Fung, P. (1994). Improving Chinese tokenization with linguistic filters on statistical lexicon acquisition. In *Proceedings of the Fourth Conference on Applied Natural Language Processing*, pages 180–181. Stuttgart.
- Wu, D. and Xia, X. (1994). Learning an english-chinese lexicon from a parallel corpus. In *AMTA-94*, pages 206–213. Association for Machine Translation in the Americas, Columbia, Maryland.