Improving Online Machine Translation Systems

Bart Mellebeek
NCLT
School of Computing
Dublin City University
Dublin 9, Ireland
mellebeek@computing.dcu.ie

Anna Khasin
NCLT
School of Computing
Dublin City University
Dublin 9, Ireland
akhasin@computing.dcu.ie

Karolina Owczarzak
NCLT
School of Computing
Dublin City University
Dublin 9, Ireland
owczarzak@computing.dcu.ie

Josef Van Genabith
NCLT
School of Computing
Dublin City University
Dublin 9, Ireland.
josef@computing.dcu.ie

Andy Way
NCLT
School of Computing
Dublin City University
Dublin 9, Ireland
away@computing.dcu.ie

Abstract

In (Mellebeek et al., 2005), we proposed the design, implementation and evaluation of a novel and modular approach to boost the translation performance of existing, wide-coverage, freely available machine translation systems, based on reliable and fast automatic decomposition of the translation input and corresponding composition of translation output. Despite showing some initial promise, our method did not improve on the baseline *Logomedia*[1] and *Systran*[2] MT systems.

In this paper, we improve on the algorithm presented in (Mellebeek et al., 2005), and on the same test data, show increased scores for a range of automatic evaluation metrics. Our algorithm now outperforms *Logomedia*, obtains similar results to *SDL*[3] and falls tantalisingly short of the performance achieved by *Systran*.

1 Introduction

There can be hardly anyone in the machine translation (MT) community who has not used one of the freely available, commercial, wide-coverage systems that exist nowadays. Despite the often poor translation quality, these systems have been the biggest contributor to the use of MT in the recent past, mainly by facilitating communication between users with no common language, and allowing for gisting of documents where the user has no language competence. Furthermore, they have the potential to open up translation markets where human translation is not a feasible option; one example is the translation on an almost daily basis of newly published data on large, company websites.

Nonetheless, neither the MT community nor the wider public are, or should be, satisfied with the translation quality currently possible via these on-line systems. Let us not forget, however, that designing robust MT systems capable of translating wide-coverage, general language material is perhaps the hardest task in our field; it is noteworthy that those systems which have had the most success have been limited to distinct sublanguage areas (e.g. the *Météo* system (Chandioux, 1976)).

Why, then, do these freely available on-line systems often produce little more than 'word salad'? Usually, such systems only consider highly limited linguistic context. As a consequence, they produce much better translations when confronted with short sentences compared to longer, more complicated strings. It is not hard to conjecture why this is so: the longer the input sentence to be translated, the more likely that the automatic translation system will be led astray by the complexities in the source and target languages.

In (Mellebeek et al., 2005), we contended that better performance in terms of output quality can be achieved by processing the texts that these systems are required to translate at any one time into smaller chunks. We demonstrated that *BabelFish* (and other similar systems) may produce better translations using the English sentence in (1) as source, and using German as the target language:

[1] www.logomedia.net
[2] www.systransoft.co.uk
[3] www.freetranslation.com

(1) The chairman, a long-time rival of Bill Gates, likes fast and confidential deals.

BabelFish translates (1) as (2):

(2) Der Vorsitzende, ein langfristiger Rivale von Bill Gates, Gleiche fasten und vertrauliche Abkommen.

Here the system has identified *fast* as the main verb, and has translated *likes* as a noun. The consequence is that it is almost impossible to understand the semantic content of the source from (2). Nonetheless, *BabelFish is* able to correctly translate the shorter strings in (3):

(3) a. The chairman likes fast and confidential deals ⟹Der Vorsitzende mag die schnellen und vertraulichen Abkommen.

b. a long-term rival of Bill Gates ⟹ein langfristiger Rivale von Bill Gates.

If a method can be found whereby long input strings such as (1) can be broken down into smaller, simpler constituents, and the resultant translations (such as those in (3)) recombined, then perfectly acceptable translations can be derived, such as that in (4):[4]

(4) Der Vorsitzende, ein langfristiger Rivale von Bill Gates, mag die schnellen und vertraulichen Abkommen.

In (Mellebeek et al., 2005), we presented such a method, together with initial results, for the TransBooster system, which takes as input strings from the Penn-II Treebank. Procedures identify constituents which are more easily translated one by one, and then recombine the resulting partial translations to generate the target string. The beauty of this is that throughout the process, the particular MT engine in question does all the translation *itself*—the system is helped to generate better translations than would otherwise have been produced to the benefit of the end user.

In (Mellebeek et al., 2005), we used the MT engines *Systran* and *Logomedia* to exemplify our method. Despite showing some initial promise, TransBooster did not improve on these baseline systems. Perhaps more noteworthy is the fact that our algorithm reverted to the baseline MT system in 85% of cases; that is, our methodology was only invoked for 15% of the input.

In this paper, we report on a number of improvements made to the algorithm, with the result that on the same test data of 800 sentences for English–Spanish, increased scores for BLEU (Papineni et al., 2002), NIST (Doddington, 2002) and F-Score (Turian et al., 2003) are seen. Thanks to these improvements, our algorithm now outperforms *Logomedia*, obtains similar results to *SDL* and falls tantalisingly short of the performance achieved by *Systran*. Furthermore, our improvements require backoff to the baseline systems in only in 23–40% of cases. We also provide a manual evaluation of a subset of the 800-sentence testset, from which we derive an upper bound of the improvements possible with our current architecture.

The remainder of this paper is organised as follows: in section 2, we provide details of related research, excluding (Mellebeek et al., 2005), the discussion of which appears in a separate section. We report on the improvements to our previously published method, and indicate why far fewer cases of backoff are required. In section 4, we reprise the experiments carried out in (Mellebeek et al., 2005), and compare the results published there with our improved scores. In addition to *Systran* and *Logomedia*, we provide an evaluation with *SDL*, and report on the relative contribution of our method to each of these three systems. We offer up some further improvements to our method in section 5, and finally we conclude.

2 Related Research

Other than (Mellebeek et al., 2005), which we present and comment on in detail in the next section, we were surprised to find that very little research has been published to investigate how such systems work, and how their obvious faults might be improved. The main point, of course, is that engines such as *BabelFish* are 'black box' systems, where any lexical and structural 'rules' are hidden from the user; the only way to figure out how the system is working is to compare the input strings against the generated translations.

(Pérez-Ortiz & Forcada, 2001) demonstrate a laboratory experiment they created in order to show students new to MT that these on-line systems are rather more sophisticated than what they term a 'Model 0' MT system, a basic word-for-word version of these on-line engines. In so doing the students infer that by iteratively pro-

[4] The only difference between this translation and that generated by a qualified translator might be the translation of the object noun phrase as *schnelle und vertrauliche Abkommen*, i.e. without the definite article (and corresponding adjective endings).

viding the MT system with more and more context, certain 'rule-based' processing is apparent.

As to seeking to improve on the output generated by such systems, the only previous (yet unpublished) research that we know of took place at the University of Leuven in the late-80s. Researchers experimented with a pre-processing system named 'Tarzan' in which a human translator identified certain clearly defined syntactic units in the input sentence which could be replaced by a syntactically similar placeholder for the purposes of simplifying the task of MT.

3 TransBooster Mark I & II

3.1 TransBooster Mark I

In (Mellebeek et al., 2005), we provided a detailed account of the TransBooster system. Essentially, there are a number of stages to the algorithm:

1. Flattening of Penn-II Trees;
2. Pivot Finding;
3. Locating Arguments and Adjuncts ('satellites') in the Source Language;
4. Use of Skeletons and Substitution Variables
5. Translation of Satellites;
6. Combining the Translation of Satellites into the Output String.

We will summarize each of these modules in the following sections.

3.1.1 Flattening Penn-II Trees

To date, we have used as input data the 49K sentences in the WSJ section of the Penn-II Treebank, although we are currently experimenting with previously unseen data which we feed into a range of state-of-the-art statistical parsers.

In order to prepare a Penn-II input sentence for translation with TransBooster, the tree for that string is flattened into a simpler structure consisting of a 'pivot' (meaningful head) and a number of 'satellites' (arguments and adjuncts), as in (5):

(5) SL: $[ARG_1]$ $[\mathsf{ADJ}_1] \ldots [ARG_L]$ $[\mathsf{ADJ}_l]$ **pivot** $[ARG_{L+1}]$ $[\mathsf{ADJ}_{l+1}] \ldots [ARG_{L+R}]$ $[\mathsf{ADJ}_{l+r}]$

where **pivot** = meaningful head for TransBooster, ARG = argument, ADJ = adjunct, {l,r} = number of ADJs to left/right of **pivot**, and {L,R} = number of ARGs to left/right of **pivot**.

To give a real example, consider the Penn-II tree in (6) corresponding to the string in (1):

(6) (S (NP-SBJ (NP (DT the) (NN chairman)) (, ,) (NP (NP (DT a) (JJ long-time) (NN rival)) (PP (IN of) (NP (NNP Bill) (NNP Gates)))) (, ,)) (VP (VBZ likes) (NP (ADJP (JJ fast) (CC and) (JJ confidential)) (NNS deals))))

Once the pivot *likes* is found (see next section), and the arguments *the chairman, a long-time rival of Bill Gates*, and *fast and confidential deals* are replaced by substitution variables (cf. section 3.1.4)—syntactically similar strings, the translations of which are known in advance—the flattened structure in (7) is automatically obtained:

(7) (S (NP-SBJ The man) (VBZ likes) (NP dogs))

This is submitted to the client MT system in order to derive the most appropriate translation for the pivot, i.e. we hypothesize that a direct MT system is far more likely to obtain the correct translation *mag* for *likes* from (7) than (1), cf. the poor translation in (2) above.

3.1.2 Finding Pivots

Most of the time, the pivot is the head terminal of the Penn-II node currently being examined. In some cases, we derive a 'complex pivot' consisting of this head terminal together with some of its rightmost neighbours, e.g. phrasal verbs or strings of auxiliaries. Another example consists of adjectival phrases, where the head dominates a PP (*close to the edge*)—here our algorithm would extract *close to* as the pivot. In (Mellebeek et al., 2005), we only considered contiguous pivots, but in our latest work some non-contiguous pivots have been incorporated (cf. section 3.2.1).

3.1.3 Locating Satellites

In TransBooster, it is essential to be able to distinguish between required elements and optional material, as adjuncts can safely be omitted from the simplified string that we submit to the MT system. This can clearly be seen in (7), where the apposition *a long-time rival* can freely be omitted. This simple method considerably reduces the complexity of the source strings. More complex cases involve the replacement of constituents by syntactically similar material (see next section).

The procedure used for argument/adjunct location is an adapted version of Hockenmaier's algorithm for CCG (Hockenmaier, 2003). The nodes we label as arguments include all the nodes Hockenmaier labels as arguments together with some of the nodes (e.g. VP children of S where S is headed by a modal verb; quantitative adjectives) which she describes as adjuncts. In ongoing research, we wish to compare this procedure with the annotation of Penn-II nodes with LFG functional information (Cahill et al., 2004).

3.1.4 Skeletons and Substitution Variables

The simplified source strings such as (5) are submitted to the MT systems, and these output target strings of the form *TL* in (8):

(8) TL: $[ARG'_1]$ $[\mathsf{ADJ}'_1]\ldots[ARG'_L]$ $[\mathsf{ADJ}'_l]$ **pivot′** $[ARG'_{L+1}]$ $[\mathsf{ADJ}'_{l+1}]\ldots[ARG'_{L+R}]$ $[\mathsf{ADJ}'_{l+r}]$

Of course, the position of the translation SAT'_i does not necessarily have to be identical to the position of the constituent SAT_i in the source. In order to find the position of the satellites in the target language, we replace each of them with an appropriate substitution variable whose translation is known in advance. Returning to (7), simple NPs such as *the man* can replace singular NPs, for instance. Submitting (7) to *BabelFish* derives the translation in (9):

(9) Der Mann mag Hunde.

Subtracting known translations from (9), and the substitution variables from (7) gives us the translation pair *likes* $\Longrightarrow$*mag* for the pivots.[5]

3.1.5 Translating Satellites

In order to retrieve the correct translation of *fast and confidential deals* in (1), we need to ensure that this constituent is inserted into a template in direct object position. Such a template is the string *The man sees*, which most of the time in German would be translated as *Der Mann sieht*, as in (10):

(10) [The man sees] fast and confidential deals. $\Longrightarrow$[Der Mann sieht] schnelle und vertrauliche Abkommen.

[5]For reasons of space, we omit here a detailed description of the research which culminated in the ultimate choice of substitution variables, together with a report on how the obtained translations of the pivots can be verified using 'pivot skeletons'. See (Mellebeek et al., 2005) for more details.

3.1.6 Forming the Translation

Compared to the other modules just described, this routine is a very simple one. Nodes containing less than 5 lexical items are not submitted to the process just described, and are translated 'as is'. Nodes containing more than 4 lexical items (established empirically as the best threshold) are recursively decomposed until all satellites are small enough to be submitted to the MT engines, and these partial translations are recombined to yield the target string output to the user.

3.2 TransBooster Mark II

For each of the routines in the previous section, we have implemented a number of improvements.

3.2.1 Pivot Finding

Most of the pivot finding improvements are related to the verbal structures. Where necessary, we now include intervening material in the verbal pivot to prevent a verbal structure with auxiliaries from being handled incorrectly. For example, in the string *The doctor has never seen this before*, it is necessary to include *never* as part of the verbal pivot (*has never seen*). In addition, personal pronouns in subject position are included in the pivot to account for zero-subjects in Spanish, as shown in (11):

(11) He has written a book $\Longrightarrow$Ha escrito un libro.

If the pronoun *he* were excluded from the pivot, then the incorrect translation *El ha escrito un libro* would be generated. For the same reason, we also include expletives in the verbal pivot.

Better string comparison methods to extract arguments from the argument skeleton have also increased the quality of the extracted pivots. This is one of the reasons why the number of backoffs are reduced considerably with respect to TransBooster Mark I.

3.2.2 Pivot Skeletons

In TransBooster Mark I, the pivot skeleton only contained arguments. However, in many cases the presence of certains adjuncts (e.g. determiner or a number in an NP) is necessary in order that a correct translation is derived.

3.2.3 Substitution Variables

Ideally, a substitution variable should have a syntactic structure similar to the constituent it

replaces. If a specific structure is not recognised, we use a non-word string such as $SAT1$ as the substitution variable, which has a high probability of leading to a bad translation of the argument skeleton. By recognising more syntactic structures, we have been able to reduce the use of these default strings from 40% in TransBooster Mark I to 5% in the current implementation.

3.2.4 Context templates

In (Mellebeek et al., 2005), only one translation for each context template was recognised. We have augmented the number of translations of each template by including variations that are dependent on the MT system used. For instance, the template *according to*, which is used to embed NPs dominated by an *-ing* form followed by a preposition (e.g. *coinciding with the conference*), is translated as *según* or *de acuerdo con* into Spanish depending on the MT system used.

3.2.5 Other improvements

Other improvements include refinements of the ARG/ADJ distinction, a better treatment of coordination and a postprocessing module that takes care of items such as punctuation, capitalisation and contraction of prepositions.

3.3 A Worked Example

In this section, we will illustrate the entire TransBooster process on the Penn-II sentence *Grumman Corp. received an $18.1 million Navy contract to upgrade aircraft electronics.* The algorithm is summarised below:

```
QUEUE = {S}
While (QUEUE not empty) {
   Node N = shift QUEUE;
   If (# leaf nodes of N <= 4) {
       translate N in context;
   }
   else{
       find pivot N;
       find satellites N;
       substitute satellites;
       build skeleton(s);
       translate skeleton(s);
       find translation pivot;
       if (translation pivot not OK) {
             translate N in context;
             break;
       }
       find loc. of translation satellites;
       add satellites to QUEUE;
   }
Recompose translations;
```

The input to the algorithm above is (12):

(12) (S (NP-SBJ (NNP Grumman) (NNP Corp.)) (VP (VBD received) (NP (NP (DT an) (ADJP (QP () (CD 18.1) (CD million)) (-NONE- *U*)) (NNP Navy) (NN contract)) (SBAR (WHNP-1 (-NONE- 0)) (S (NP-SBJ (-NONE- *T*-1)) (VP (TO to) (VP (VB upgrade) (NP (NN aircraft) (NNS electronics)))))))) (. .))

QUEUE = {S}

- Step 1:
  - S contains more than 4 leaf nodes $\Longrightarrow$not ready for translation $\Longrightarrow$decompose
  - Find pivot S
    pivot = 'received'
  - find satellites
    ARG1 = 'Grumman Corp.'
    ARG2 = 'an $18.1 million Navy contract to upgrade aircraft electronics.'
  - substitute satellites
    ARG1_subst = 'The man'
    ARG2_subst = 'a car'
  - build skeleton(s)
    arg. skel = 'The man received a car.'
  - translate skeleton(s)
    trans. arg. skel. = 'El hombre recibió un automóvil'
  - find translation pivot
    trans. pivot = 'recibió'
  - pivot skel = 'Grumman Corp. received a contract.'
    trans pivot skel = 'Grumman Corp. recibió un contrato.'
    'recibió' is present in trans pivot skel $\Longrightarrow$continue
  - find location of translation satellites
    ARG1' left of pivot', ARG2' right of pivot'
  - add satellites to QUEUE
    QUEUE = {ARG1, ARG2}
- Step 2:
  - ARG1 'Grumman Corp.' contains less than 5 leaf nodes $\Longrightarrow$ready for translation $\Longrightarrow$translate in context
  - 'Grumman Corp. is sleeping.' $\Longrightarrow$'Grumman Corp. está durmiendo'
  - ARG1' = 'Grumman Corp.'
    QUEUE = {ARG2}
- Step 3:
  - ARG2 'an $18.1 million Navy contract to upgrade aircraft electronics' contains more than 4 leaf nodes $\Longrightarrow$not ready for translation $\Longrightarrow$decompose
  - pivot = 'an $18.1 million Navy contract'

- ADJ21 = 'to upgrade aircraft electronics'
- ...
- QUEUE = {ADJ21}

- Step 4:
  - ADJ21 'to upgrade aircraft electronics' contains less than 5 leaf nodes $\Longrightarrow$ready for translation $\Longrightarrow$translate in context
  - 'A man to upgrade aircraft electronics' $\Longrightarrow$'Un hombre actualizar equipo electrónico de aeronave'
  - ADJ21' = 'actualizar equipo electrónico de aeronave'
  - QUEUE = { }
- Step 5:
  - Recompose translation:
    'Grumman Corp. recibió uno $18.1 millón contrato de la marina de guerra actualizar equipo electrónico de aeronave.'
  - Original translation by Logomedia:
    'Grumman Corp. Recibió uno $18.1 millón marina se compromete por contrato actualizar equipo electrónico de aeronave.'

3.3.1 Analysis of the example sentence

Although the context template of ADJ21 *to upgrade aircraft electronics* in step 4 is not perfect, the translation output of TransBooster clearly outperforms that of *Logomedia*. The main reason for this improvement is the fact that our reduction of syntactic complexity forces *Logomedia* to disambiguate *contract* correctly as a noun rather than as a verb.

4 Experiments, Results and Evaluations

The effectiveness of our algorithm is measured against an 800-sentence testset (min. 1 word, max. 54 words, ave. 19.75 words) from Section 23 of the Penn-II Treebank. In what follows, we present results of an automatic evaluation, using BLEU, NIST and F-Score as metrics, against the full testset. [6] We then conduct a manual evaluation of 321 sentences for which some difference in translation between TransBooster and *Logomedia* is found. The manual evaluation allows us to identify the phenomena for with TransBooster is most successful. Finally, we extract a 121-sentence subset from these 321 sentences where the manual evaluation scores for TransBooster are higher than for the baseline *Logomedia* system, corroborate these scores with automatic evaluation metrics and use the scores to derive an upper bound to the improvements possible with our current architecture.

[6] The statistical significance of these results was corroborated in each case by using the NIST/BLEU resampling toolkit described in (Zhang et al., 2004).

4.1 Automatic Evaluation

4.2 Full 800-Sentence Testset

We present here results of an automatic evaluation of the TransBooster method against the baseline systems *Logomedia, Systran* and *SDL*, for English–Spanish.

4.2.1 Logomedia

	BLEU	NIST	GTM
Logomedia	.3108	7.3428	.5740
TransBooster	.3163	7.3901	.5753
Percent. of Baseline	101.7%	100.6%	100.2%

Table 1: TransBooster vs. Logomedia: Results on the full 800-sentence testset

The results using *Logomedia* as the test MT system are shown in Table 1. TransBooster improves on the baseline system according to all 3 automatic evaluation metrics.The TransBooster algorithm was used for 545 (68%) of the 800 sentences, and for the remaining 255 (32%) we backed off to the baseline system. Off the 545 sentences for which the TransBooster algorithm was invoked, 224 translations were identical to those produced by the baseline system, with 321 containing some differences.

4.2.2 SDL

	BLEU	NIST	GTM
SDL	.2988	7.3000	.5738
TransBooster	.2971	7.3438	.5688
Percent. of Baseline	99.4%	100.6%	99.1%

Table 2: TransBooster vs. SDL: Results on the full 800-sentence testset

The results using *SDL* as the test MT system are shown in Table 2. TransBooster improves on SDL according to NIST, while the BLEU score and the F-Score are slightly lower than the baseline system's score. TransBooster algorithm was used for 615 (77%) of the 800 sentences, with the remainder backing off to the baseline system. Off these 615 sentences, 242 translations were identical to those produced by the baseline system, with 373 containing some differences.

4.2.3 Systran

The results using *Systran* as the test MT system are shown in Table 3. In this case, TransBooster

	BLEU	NIST	GTM
Systran	.2963	7.1781	.5631
TransBooster	.2891	7.0983	.5584
Percent. of Baseline	97.6%	98.9%	99.1%

Table 3: TransBooster vs. Systran: Results on the full 800-sentence testset

fails to improve on the baseline system, but the scores are only slightly lower than the results obtained by *Systran*. The TransBooster algorithm was used for 481 (60%) of the 800 sentences, with the remainder backing off to the baseline system. Off these 481 sentences, 167 translations were identical to those produced by the baseline system, with 314 containing some differences.

4.3 Manual Evaluation

We observed above that for *Logomedia*, 321 out of the 800-sentence testset received different translations via TransBooster. After a manual evaluation based on an average between accuracy and fluency, we considered 121 (38%) of these to be better when TransBooster was used, 79 (24%) being worse, and the remaining 122 (38%) adjudged to be similar. These judgements are backed up by submitting the 121-sentence subset to an automatic evaluation, as shown in Table 4. Since the testset consists only of sentences that were considered better when TransBooster was used, these results indicate an upper boundary on the automatic evaluation improvements that can be achieved using the current architecture of TransBooster.

	BLEU	NIST	GTM
Logomedia	.3102	6.2346	.5454
TransBooster	.3442	6.5427	.5621
Percent. of Baseline	110.9%	104.9%	103.0%

Table 4: TransBooster vs. Logomedia: Results on the 121-sentence subset of the full testset where using TransBooster improved translation quality

4.3.1 Discussion

In general, where improvements are seen by invoking the TransBooster method on *Logomedia*, these can be divided into 4 classes: (i) word order (approx. 20% of cases); (ii) better target language lexical selection (50%); (iii) better agreement (10%); and (iv) better homograph resolution (20%). Examples of some of these appear in (13):

(13) **Better Lexical Selection**:

Orig: One week later, Leonard H. Roberts, president and chief executive officer of Arby's, was *fired* in a dispute with Mr. Posner. $\Longrightarrow$
Logomedia: Uno semana después, Leonard H Roberts, presidente y Funcionario en Jefe Ejecutivo de Arby's, fue *disparado* en una disputa con el Sr. Posner.
TransBooster: Uno semana después, Leonard H Roberts, presidente y Funcionario en Jefe Ejecutivo de Arby's, fue *despedido* en una disputa con el Sr. Posner.

Homograph resolution and agreement:

Orig: "I find it hard to conceive of people switching over to CNN for what, at least in the public's mind, is the same news," *says Reuven Frank*, the former two-time president of NBC News and creator of the Huntley-Brinkley *Report*.
Logomedia: "Lo encuentro difícil concebir de personas que cambian to CNN para qué, por lo menos en la mente del público, es las mismas noticias", *decir que Reuven Frank*, los ex dos veces presidente de NBC News y creador del Huntley - Brinkley *presentan un informe*.
TransBooster: "Lo encuentro difícil concebir que las personas cambien a CNN para lo que, por lo menos en la mente del público, es las mismas noticias," *dice Reuven Frank*, el ex presidente dos veces de NBC News y creador del Huntley - Brinkley *Report*.

5 Further Improvements

Where invoking the TransBooster algorithm results in worse translation quality, in most cases this is due to missing context. We want to improve the treatment of context in two ways: (i) Use real heads as substitution variables, as substituting constituents with their real heads instead of with a syntactically similar variable is likely to yield better results. However, this

method poses a number of retrievability problems, since we do not know the translation of the real heads beforehand. A possible solution would be to use tracker material inside the substitution variable (e.g. numerals, determiners, etc.) to retrieve its translation; (ii) Extending and refining the context templates that we currently use to mimic the original syntactic structure of the sentence is also likely to lead to improved results.

6 Conclusions

The translation quality obtained from on-line MT systems deteriorates with longer input strings. We have presented a method where we recursively break down sentences from the Penn-II Treebank into smaller and smaller constituents, and confront the MT system with these shorter sub-strings. We keep track of where those individual parts fit into the overall translation in order to stitch together the translation result for the entire input string. Throughout the process the commercial MT engine does all the translation *itself*: our method helps the system to improve its own output translations.

We have detailed a number of improvements to the original TransBooster system presented in (Mellebeek et al., 2005). Thanks to these improvements, TransBooster now outperforms *Logomedia* and obtains results similar to *SDL* and only slightly below the scores achieved by *Systran*.

Also, the improvements have led to a massive reduction in the amount of times we back off to the default MT system; currently, this ranges from 40% for *Systran*, to just 23% for *SDL*, whereas the backoff figure for the work in (Mellebeek et al., 2005) was around 85%. Of course, it remains a major thrust of current research to lower these figures still further.

We have outlined a number of ways to further improve the system and we are confident that future versions of TransBooster will obtain net improvements on all 3 baseline systems.

Acknowledgements

This research was funded by an Enterprise Ireland Basic Research grant *SC/2003/282*.

References

Cahill, A., M. Burke, R. O'Donovan, J. van Genabith and A. Way. 2004. Long-Distance Dependency Resolution in Automatically Acquired Wide-Coverage PCFG-Based LFG Approximations. In *Proc. 42nd Annual Meeting of the Assoc. for Computational Linguistics*, Barcelona, Spain, pp.319–326.

Chandioux, J. 1976. MÉTÉO: un système opérationel pour la traduction automatique des bulletins météorologiques destinés au grand public. *META* **21**:127–133.

Doddington, G. 2002. Automatic evaluation of MT quality using n-gram co-occurrence statistics. In *Proc. Human Language Technology*, San Diego, CA., pp.128–132.

Hockenmaier, J. 2003. Parsing with Generative models of Predicate-Argument Structure. In *Proc. 41st Annual Conference of the Association for Computational Linguistics (ACL-03)*, Sapporo, Japan, pp.359–366.

Mellebeek, B., A. Khasin, J. Van Genabith and A. Way. 2005. TransBooster: Boosting the Performance of Wide-Coverage Machine Translation Systems. In *Proceedings of the 10th Annual Conference of the European Association for Machine Translation (EAMT-05)*, Budapest, Hungary, pp 189–197.

Papineni, K., S. Roukos, T. Ward and W-J. Zhu. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proc. 40th Annual Meeting of the Association for Computational Linguistics (ACL-02)*, Philadelphia, PA., pp.311–318.

Pérez-Ortiz, J. & M. Forcada. 2001. Discovering Machine Translation Strategies: Beyond Word-for-Word Translation: a Laboratory Assignment. In *Workshop on Teaching Machine Translation, MT Summit VIII*, Santiago de Compostela, Spain, pp.57–60.

Turian, J., L. Shen. and D. Melamed. 2003. Evaluation of Machine Translation and its Evaluation. *MT Summit IX,* New Orleans, LA., pp.386–393.

Zhang, Y., and S. Vogel. 2004. Measuring Confidence Intervals for the Machine Translation Evaluation Metrics. *Proceedings of the Tenth Conference on Theoretical and Methodological Issues in Machine Translation* Baltimore, MD., pp.85–94.