Tree-based Hybrid Machine Translation

Andreas Kirkedal
Centre for Computational Modelling of language
Institute for International Language Studies and Computational Linguistics
Copenhagen Business School
ask.isv@cbs.dk

Abstract

I present an automatic post-editing approach that combines translation systems which produce syntactic trees as output. The nodes in the generation tree and target-side SCFG tree are aligned and form the basis for computing structural similarity. Structural similarity computation aligns subtrees and based on this alignment, subtrees are substituted to create more accurate translations. Two different techniques have been implemented to compute structural similarity: *leaves* and *tree-edit distance*. I report on the translation quality of a machine translation (MT) system where both techniques are implemented. The approach shows significant improvement over the baseline for MT systems with limited training data and structural improvement for MT systems trained on Europarl.

1 Introduction

Statistical MT (SMT) and rule-based MT (RBMT) have complimentary strengths and combining their output can improve translation quality. The underlying models in SMT lack linguistic sophistication when compared to RBMT systems and there is a trend towards incorporating more linguistic knowledge by creating hybrid systems that can exploit the linguistic knowledge contained in hand-crafted rules and the knowledge extracted from large amounts of text.

Hierarchical phrases (Chiang, 2005) are encoded in a tree structure just as linguistic trees. Most RBMT systems also encode the analysis of a sentence in a tree. The rules generating hierarchical trees are inferred from unlabeled corpora and RBMT systems use hand-crafted rules based in linguistic knowledge. While the trees are generated differently, alignments between nodes and subtrees in the generation phase can be computed. Based on the computed alignments, substitution can be performed between the trees.

The automatic post-editing approach proposed in this paper is based on *structural similarity*. The tree structures are aligned and subtree substitution based on the similarity of subtrees performed. This knowledge-poor approach is compatible with the surface-near nature of SMT systems, does not require other information than what is available in the output, and ensures that the approach is generic so it can, in principle, be applied to any language pair.

2 Hybrid Machine Translation

Hybrid machine translation (HMT) is a paradigm that seeks to combine the strengths of SMT and RBMT. The different approaches have complementary strengths and weaknesses (Thurmair, 2009) which have led to the emergence of HMT as a subfield in machine translation research.

The strength of SMT is robustness - i.e. it will always produce an output - and fluency due to the use of language models. A weakness of SMT is the lack of explicit linguistic knowledge, which make translation phenomena requiring such information, e.g. long-distance dependencies, difficult to handle.

RBMT systems translate more accurately in cases without parse failure, since they can take more information into account e.g. morphological, syntactic or semantic information, where SMT only uses surface forms. RBMT often suffer from lack of robustness when parsing fails and

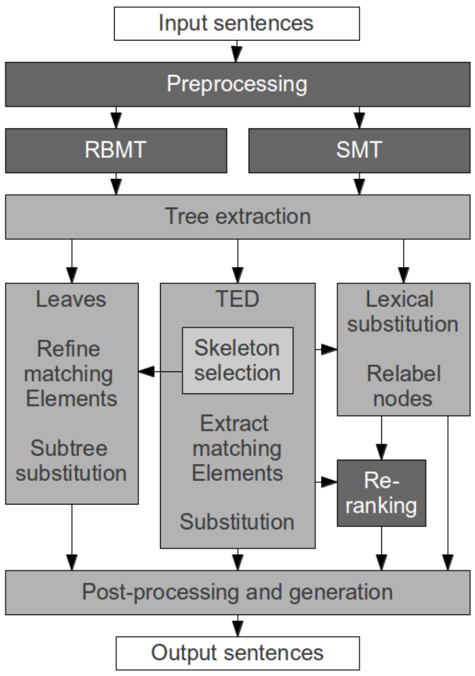

Figure 1: Hybrid system architecture.

in lexical selection in transfer. RBMT systems are also very costly to build, and maintenance and development can be very complex e.g. due to the interdependency of rules.

The post-editing approach attempts to incorporate the linguistic knowledge encoded in target-side dependency trees into hierarchical trees produced by an SMT system.

2.1 Related work

System combinations by coupling MT systems serially or in parallel have been attempted before e.g. via hypothesis selection (Hildebrand and Vogel, 2008), by combining translation hypotheses locally using POS tags (Federmann et al., 2010) or by statistical post-editing (SPE) (Simard et al., 2007). In hypothesis selection approaches, a number of MT systems produce translations for an n-best list and use a re-ranking module to rescore the translations. Using this approach, the best improvements are achieved with a large number of systems running in parallel and this is not feasible in a practical application, mostly due to the computational resources required by the component systems. The translations will also not be better than the one produced by the best component system. Tighter integration of rule-based and statistical approaches have also been proposed: Adding probabilities to parse trees, pre-translation word reordering, enriching the phrase table with output phrases from a rule-based system (Eisele et al., 2008), creating training data from RBMT systems etc. The factored translation models also present a way to integrate rule-based parsing systems.

```
Jeg [jeg] 1S NOM @SUBJ #1->2
arbejder [arbejde] <mv> V PR AKT @FS-STA #2->0
hjemme [hjemme] <aloc> ADV LOC @<ADVL #3->2
. [.] PU @PU #4->0
```

Figure 2: Disambiguated CG representation for *I work at home*. Dependency annotation is indicated by the #-character.

The automatic post-editing approach proposed here does not exactly fit the classification of parallel coupling approaches in Thurmair (2009). Other coupling architectures with post-editing work on words or phrases and generate confusion networks or add more information to identify substitution candidates, while the units focused on here are graphs and no additional information is added to the MT output. This approach does select a skeleton upon which transformations are conducted as in Rosti et al. (2007) and requires the RBMT system to generate a target side language analysis which must be available to the post-editing systems, but does not require a new syntactic analysis of noisy MT output. The architecture of the hybrid system used in this paper is parallel coupling with post-editing. A diagram of the implemented systems can be seen in Figure 1. The dark grey boxes represent pre-existing modules and open source software and the light grey boxes represent the additional modules developed to implement the post-editing approach.

2.2 RBMT Component

The Danish to English translation engine in GramTrans (Bick, 2007) is called through an API. The output is a constraint grammar (CG) analysis on the target language side after all transfer and target side transformation rules have been applied. Example output is shown in Figure 2. In the analysis, dependency information is provided and they form the basis for creating the tree used for structural similarity computation. Part-of-speech tags, source and target surface structure, sentence position and dependency information are extracted from the CG analysis.

GramTrans is created to be robust and produce as many dependency markings as possible to be used in later translation stages. Errors in the assignment of functional tags propagate to the dependency level and can result in markings that will produce a dependency tree and a number of unconnected subgraphs with circularities. This presents a problem if the dependency markings are the basis for creating a dependency tree because it is not straight-forward to reattach a subgraph correctly, when the grammatical tags cannot be relied upon.

2.3 SMT Component

A CKY+ algorithm for chart decoding is implemented in Moses (Koehn et al., 2007) for tree-based models and is used as the SMT component system in this paper.

Hierarchical phrases are phrases that can contain subphrases, i.e. a hierarchical phrase contains non-terminal symbols. An example rule from Danish to English:

$$X_1 \text{ i øvrigt } X_2 \longrightarrow \text{moreover, } X_1 \; X_2$$

X_n is a nonterminal and the subscript identifies how the nonterminals are aligned. The hierarchical phrases are learned from bitext with unannotated data and are formally productions from a synchronous context-free grammar (SCFG) and can be viewed as a move towards syntax-based SMT (Chiang, 2005). Since hierarchical phrases are not linguistic, Chiang makes a distinction between *linguistically* syntax-based MT and *formally* syntax-based MT where hierarchical models fall in the latter category because the structures they are defined over are not linguistically informed, i.e. unannotated bitexts.

A hierarchical model is based on a SCFG and the elementary structures are rewrite rules:

$$X \longrightarrow \langle \gamma, \alpha, \sim \rangle$$

As above, X is a nonterminal, γ and α are both strings of terminals and nonterminals and $\sim$ is a 1-to-1 correspondence between nonterminals in γ and α. As in shown previously, the convention is to use subscripts to represent $\sim$.

To maintain the advantage of the phrase-based approach, *glue rules* are added to the rules that are otherwise learned from raw data:

$$S \longrightarrow \langle S_1 X_2, S_1 X_2 \rangle$$
$$S \longrightarrow \langle X_1, X_1 \rangle$$

Only these rewrite rules contain the nonterminal S. These rules are added to give the model the option of combining partial hypotheses serially and they make the hierarchical model as robust as the traditional phrase-based approaches.

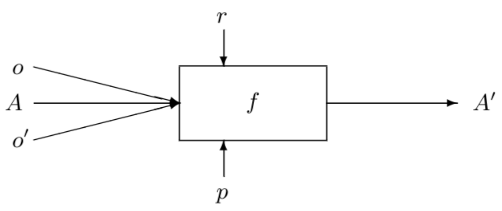

Figure 3: The matching process.

The Moses chart decoder was modified to output trace information from which the n-best hierarchical trees can be reconstructed. The trace information contains the derivations which produce the translation hypotheses.

The sentence–aligned Danish-English part of Europarl (Koehn, 2005) was used for training, and to tune parameters with MERT, the test set from the NAACL WMT 2006 was used (Koehn and Monz, 2006). GIZA++ aligns hierarchical phrases which were extracted by Moses to train a translation model and a language model was trained with SRILM (Stolcke, 2002). Moses was trained using the Experimental Management System (EMS) (Koehn, 2010) and the configuration followed the standard guidelines in the syntax tutorial.[1] To train SRILM, the English side of Europarl was used.

3 Matching Approach

The post-editing approach relies on structures output by the component systems. It is necessary to find similar structures to perform subtree substitution. Matching structures is a problem in several application areas such as semantic web, schema and ontology integration, query mediation etc. Structures include database schemas, directories, diagrams and graphs. Shvaiko and Euzenat (2005) provide a comprehensive survey of matching techniques.

The *matching operation* determines an alignment between two structures and an alignment is a set of *matching elements*. A matching element is a quintuple: $\langle id, e, e', n, R \rangle$:

id	Unique id.
e, e'	Elements from different structures.
n	Confidence measure.

[1]http://www.statmt.org/moses/?n=Moses.SyntaxTutorial

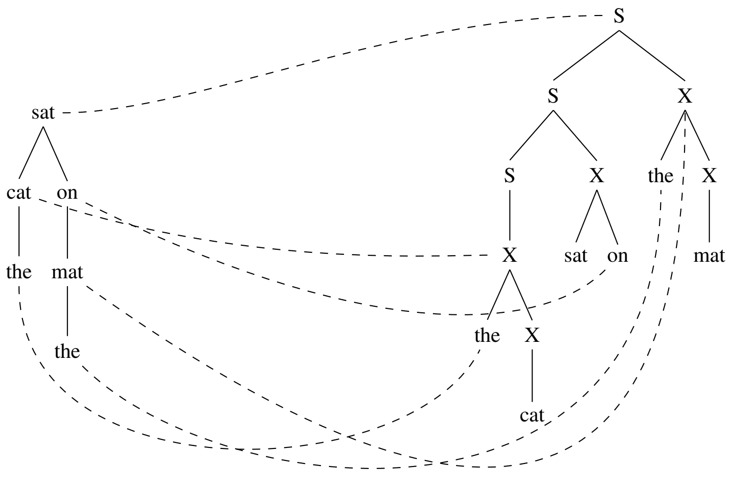

Figure 4: The refined alignment from dependency tree to hierarchical tree.

R The relation holding between the elements.

The resources that can be used in the matching process are shown in Figure 3. o and o' are the structures to be matched, A is an optional existing alignment, r is external resources, p is parameters, weights and thresholds and A' is the set of matching elements created by the process. In this paper, only matching elements with an equivalence relation $(=)$ are used.

The returned alignment can be a new alignment or a refinement of A. o will be a dependency tree and o' the hierachical trees from the SMT component system. To compute the initial alignment A between hierarchical and dependency trees, the source to target language phrase alignment output by the component systems is used. So the initial alignment between leaf nodes in target-side trees are computed over the alignment to the source language.

An important decision regarding this hybrid approach is how to compute the alignment and the size of the substituted subtrees. Irrespective of which technique is chosen to compute structural similarity, the resulting alignment should be refined to contain matching elements between internal nodes as shown in Figure 4.

3.1 Alignment Challenges

The change made to the chart decoder to output the n-best trace information is simple and does not output the alignment information. Currently, the tree extraction module computes an alignment between the source and target language phrases.

The segmentation of words into phrases done by Moses does not always correspond to the word-based segmentation required by the CG parser; phrases recognised by the CG parser rarely correspond to phrases in Moses and the hierarchical phrase alignment is not easy to handle.

Aligning hierarchical phrases like **(a)** in Figure 5 is not complicated. The ordering is identical and the Danish word *offentliggøres* is aligned to *will be published*. The numbers `1-3` refer to the alignment of non-terminal nodes based on phrase positions.

It is more complicated to align **(b)** in Figure 5. There are two methods of handling this type of alignment appropriate for the component systems. Because there are an equal number of tokens in the English phrase and Danish phrase, aligning the tokens 1-1 monotonically would be a solution that, in this case, results in a correct alignment.

Another approach relies on weak word reordering between Danish and English and would align *findes* with *there are*. This reduces the alignment problem to aligning *vi der* with *we*. In this case, the alignment is noisy, but usable for creating matching elements. Both approaches are implemented in the hybrid system and the first approach supercedes the second due to the advantage of correlating with the CG approach.

An initial element-level alignment between nodes in a dependency tree and a hierarchical tree is computed over the source language and creates a set of *matching elements* containing aligned nodes.

3.2 Alignment Refinement

Between a dependency and an hierarchical tree, an element-level alignment needs to be refined to

```
(a)    offentliggøres X : X -> will be published X : 1-3
(b)    vi X der X findes : X -> X, we X there are : 1-3 3-0
```

Figure 5: Simplified example of a simple alignment.

a structure-level alignment similar to the one in Figure 4.

Not all matching elements in an initial alignment should be refined e.g. if both nodes in a matching element are leaf nodes, no refinement is needed. Criteria for selecting initial matching elements for refinement are needed.

In the RBMT output, there are no indications of where the parser encountered problems. If a surface form is an out-of-vocabulary (OOV) word, the morphological analyser is used to assign a lexical category based on the word form, hypothesise additional tags based on the analysis and proceed with parsing. In the SMT output, an OOV marker is appended to a surface form to indicate that the word has not been translated. The marker gives an indication of where enriching a hierarchical tree with RBMT output can result in improvement of translation quality.

Based on these observations, hierarchical trees are chosen to function as skeletons. Substituting dependency subtrees into a hierarchical tree is more straightforward than using dependency trees as skeletons. It was not possible to identify head-dependent relations based solely on the information contained in hierarchical subtrees while removing subtrees from hierarchical trees and inserting dependency subtrees does not destroy linguistic information in the tree and dependency subtrees can easily be transformed into a hierarchical-style subtree.

Leaves Based on the OOV marker, a matching technique based on leaf nodes is implemented to refine matching elements and based on this alignment, substitute hierarchical subtrees with dependency subtrees.

The dependency subtree is identified by collecting all descendants of a node. The descendants are handled as leaf nodes because both leaf and nonterminal nodes contain surface forms in a dependecy tree.

The dependency trees provided by GramTrans are not always projective. Subtrees may not represent a continuous surface structure and a continuous subtree must be isolated before an alignment between subtrees can be found because the hierarchical trees resemble phrase structure trees and discontinuous phrases are handled using glue rules.

To identify the corresponding subtree in the hierarchical tree, the matching elements that contain the nodes in the dependency subtree are collected and a path from each leaf node to the root node is computed. The intersection of nodes is retrieved and the root node of the subtree identified as the lowest node present in all paths. It is not always possible to find a common root node besides the root node of the entire tree. To prevent the loss of a high amount of structural information, the root node cannot be replaced or deleted.

3.3 Substitution based on an edit script

An algorithm for computing structural similarity is the *Tree Edit Distance* (TED) algorithm, which computes how many operations are necessary for transforming one tree into another tree. Following Zhang and Shasha (1989) and Bille (2005), the operations are defined on nodes and the trees are ordered, labelled trees. There are 3 different edit operations:

rename Change the label of a node in a tree.

delete Remove a node n from a tree. Insert the children of n as children of the parent of n so the sequence of children are preserved. The deleted node may not be the root node.

insert Insert a node as the child of a node n in a tree. A subsequence of children of n are inserted as children of the new node so the sequence of children are preserved. An insertion is the inverse operation of a deletion.

A cost function is defined for each operation. The goal is to find the sequence of edit operations that turns a tree T_1 into another tree T_2 with minimum cost. The sequence of edit operations is called an *edit script* and the cost of the optimal edit script is the tree edit distance.

The cost functions should return a distance metric and satisfy the following conditions:

1. $\gamma(i \rightarrow j) \geq 0$ and $\gamma(i \rightarrow i) = 0$

2. $\gamma(i \to j) = \gamma(j \to i)$

3. $\gamma(i \to k) \leq \gamma(i \to j) + \gamma(j \to k)$

γ is the cost of an edit operation.

The *edit distance mapping* is a representation of an edit script. A rename operation is represented as $(i_1 \to j_2)$ where the subscript denotes that the nodes i and j belong to different trees. $(i_1 \to \epsilon)$ represents a deletion and $(\epsilon \to j_2)$ an insertion.

The cost of an edit distance mapping is given by:

$$\gamma(M) = \sum_{(i,j)\in M} \gamma(i \to j) + \sum_{i \in T_1} \gamma(i \to \epsilon) + \sum_{j \in T_2} \gamma(\epsilon \to j)$$

$j \in T_2$ means j is in the set of nodes in T_2.

It is important to note that the trees are ordered trees. The unordered version of the tree edit distance problem is NP-hard, while polynomial algorithms based on dynamic programming exist for ordered trees.

The algorithm does not require an input alignment or external resources. The cost functions for deletion, insertion and renaming must be defined on the information present in the nodes and a unique id must be assigned to the nodes. This id is assigned by traversing the tree depth-first and assigning an integer as id. The algorithm visits each node in the trees in post order and determines based on the cost assigned by the cost functions, which edit operation should be performed.

To generate matching elements that align dependency nodes to nonterminal hierarchical nodes, cost functions for edit operations are modified to assign a lower cost to rename operations where one of the nodes is a hierarchical nonterminal node. If two nodes have the same target and source phrase, a rename operation does not incur any cost and neither does the renaming of untranslated phrases. This ensures that matching elements from the initial alignment that does not require refinement are not altered. Also, if the source is the same and the difference in sentence position is no more than five, the renaming cost is reduced. Experiments showed that a window of five words was necessary to account for differences in sentence position and prevent alignment to nodes later in the sentence with the same source phrase.

This technique is independent of the OOV marker and creates a structure-level alignment. The substitutions performed can be of very high quality but some untranslated words might not be handled. If the system finds any OOV words in the hierachical tree after substitution, a rename operation is carried out on the node.

The extracted matching elements are noisy because they rely on the noisy source to target language alignment and the RBMT engine can also produce an inaccurate translation making the substitution counter-productive. Further limitations on the cost functions become too restrictive and produce too few matching elements. To avoid some of the noise, all permutations of applying substitutions based on the edit script are generated, re-ranked and the highest scoring hypothesis chosen as the translation.

3.4 Generation

To ensure that the surface string generated from the newly created tree will have the correct word ordering, the dependency subtree is transformed before being inserted into the hierarchical tree. To create the insertion tree, the dependency nodes are inserted as leaf nodes of a dummy node. The dummy node is inserted before the root node of the aligned hierarchical subtree and the information on the root node copied to the new node. Subsequently, the hierarchical nodes are removed from the tree. If both nodes in a matching element are leaf nodes, the hierarchical node is relabeled with information from the dependency node.

4 Experiments

The experiments have been conducted between Danish and English. The language model trained with EMS is used to re-rank translation alternatives. BLEU (Papineni et al., 2002), TER (Snover et al., 2006) and METEOR (Banerjee and Lavie, 2005) scores will be reported.

4.1 Experimental Setup

Two sets of five experiments have been conducted. The first set of experiments use the initial 100,000 lines from Europarl for training Moses and the second set of experiments use the full Europarl corpus of ca. 1.8 mio sentences. The SMT baseline is the hierarchical version of Moses.

TED Skeleton Selection The impact of choosing the translation hypothesis with a minimal edit

Metrics:	BLEU	TER	METEOR
RBMT baseline	19.35	64.54	53.19
SMT baseline	30.16 (22.63)	57.16 (63.10)	59.51 (50.72)
Lexical substitution	**30.53** (**25.28**)	**56.40** (60.56)	**61.22** (57.24)
Leaves technique	29.06 (21.96)	57.96 (64.80)	60.09 (54.32)
TED skeleton(any bias)	30.16 (22.63)	57.08 (62.98)	59.46 (50.75)
TED-R 1-best	29.78 (25.16)	57.25 (60.51)	59.87 (57.31)
TED-R skeleton(any bias)	29.99 (25.18)	56.72 (**60.44**)	60.79 (**57.34**)

Table 1: Automatic evaluation. 100k experiments in parentheses

distance to the dependency tree from the rule-based system is investigated. In one setting, the cost functions adhere to the constrictions of computing a distance metric. Two settings test the impact of biasing the insertion and deletion cost functions to assign a lower cost to inserting/deleting nonterminals, i.e. turning the dependency tree into the hierarchical tree and vice versa.

TED is computed for 20 translation hypotheses and the best performing setting reported.

Leaves An experiment using the leaves technique has been conducted. The experiment is performed using the best hypothesis from Moses and also using TED to chose the most structurally similar skeleton. The best setting will be reported.

Lexical substitution To be able to compare a more naive approach, subtree substitution based on the initial element-level alignment between leaf nodes is used. In this approach, a subtree is one node. The technique is identical to using the RBMT lexicon to lookup untranslated words and inserting them in the translation.

TED-R An experiment where the mappings that represent a rename operation, which are produced during TED computation, are extracted and used as matching elements is conducted. Mapping elements containing only punctuation or the root node of either tree are discarded. All combinations of substitutions based on the extracted matching elements are performed and the highest ranking hypothesis according to a language model is chosen as the final translation.

The extracted matching elements may not incorporate all the untranslated nodes. All untranslated nodes are subsequently translated using lexical substitution as mentioned above. The subtrees inserted into the hierarchical tree will undergo the same transformation as the subtrees inserted using the leaves technique.

This experiment is evaluated using both the 1-best hierarchical tree as skeleton and choosing the skeleton using TED. All three settings are tested and the best performing experiment reported.

4.2 Evaluation

The results of the automatic evaluation can be seen in Table 1. *Skeleton* indicates that TED was used to pick the hierarchical tree. The best evaluations are in bold.

100k The RBMT baseline is outperformed by all hybrid configurations, though it does have a higher METEOR score than the SMT baseline and skeleton selection. Lexical substitution and TED-R obtains an increase of ca. 2.5 BLEU, 4 TER and 4 METEOR points over the best baseline scores. The leaves technique decreases the metrics except for METEOR and the skeleton selection only shows an insignificant improvement.

Europarl Only lexical substitution improve all metrics over the baseline. Using the leaves technique again results in a decrease in BLEU and TER, but improves METEOR. The impact of skeleton selection is similar to previous experiments, but the use of skeleton selection in TED-R has become larger.

Manual Evaluation The evaluators rank 20 sentences randomly extracted from the test set on a scale from 1-5 with 5 being the best and it is possible to assign the same score to multiple translation alternatives. This evaluation was inspired by the sentence ranking evaluation in Callison-Burch et al. (2007). The five sentences to be evaluated will come from the RBMT and SMT baselines, lexical substitution, leaves technique and TED-R skeleton and the evaluators are 5 Danes who have studied translation with English as second language and 3 native English speakers.

The baseline systems make up 85% of the lowest ranking. The distribution between systems is more even for the second lowest ranking with the baselines only accounting for 52.6%. In the middle ranking, the top scorer is lexical substitution

System	1	2	3	4	5	Avg. rank
SMT	**53**	**64**	30	12	1	2.025
RBMT	14	48	61	29	8	2.806
Lex. sub.	3	33	**63**	**58**	3	3.156
Leaves	6	33	61	55	5	3.125
TED-R	3	35	46	55	**21**	3.35

Table 2: Rankings from the manual evaluation of the second set of experiments.

with a small margin to the RBMT baseline and the leaves technique. The many assignments of rank *3* could indicate that many of the translations produced can be used for gisting, i.e. get an impression of what information the source text conveys, but not enough to give a complete understanding, but can also be a result of being the middle value and chosen when the evaluators are in doubt. Lexical substitution is also the top scorer in the second-best ranking, followed closely by the other hybrid configurations and the hybrid systems account for 80.3% of the second-best rankings. TED-R recieves more top rankings than the other systems combined (55.3%). The RBMT baseline achieves second-most top-rankings. This can be attributed to the cases where the rules did not encounter unknown words and created very accurate translations, as is the hallmark of RBMT.

5 Discussion

It is not surprising that lexical substitution achieves a significant increase in all metrics. The approach only translates untranslated words using the RBMT lexicon. This can improve the translation or, because of noisy matching elements, introduce wrong words but the penalty incurred for untranslated words and wrongly translated words is the same if the number of tokens is similar. Further, lexical substitution does not rely on structural similarity and can avoid the potential sources of errors encountered at a later processing stage.

Skeleton selection has little impact on the metrics and distinct derivations can result in the same surface structure, giving the same scores, but it is evident that finding the most similar tree improves substitution.

The improvements observed in the 100k experiments are not evident in the metrics when the full Europarl data is used. The more powerful SMT system is able to handle more translations but manual evaluation reveals a distribution where the majority of rankings for the baseline systems are in the lower half and rankings for the hybrid systems tend more towards the mid-to-upper rankings, with TED-R having more distribution around the second-best and highest score. This indicates that the approach creates more accurate translations.

SMT baseline	(COM (1999) 493 - C5-0320 / 1999 - 1999 / 2208 (COS))
Leaves	(came (1999) 493 - C5-0320/1999-1999/2208 (COM COS)) - C5-0320 / 1999 - 1999 / 2208 (
TED-R	(COM (1999) 493 - C5-0320/1999-1999/2208 / 1999 - 1999 / 2208 (COS))

Table 3: Substitution of numbers.

The leaves technique consistently underperforms lexical substitution, but manual evaluation shows a high correlation between the two methods and their average ranks are similar. TED-R is ranked higher than the leaves technique in the metrics and manual evaluation also ranks TED-R higher than lexical substitution. This suggests that the extra surface structure removed is not present in the reference translation and that TED-R is a better implementation of the post-editing approach.

Subtree substitution, whether using leaves or TED, does not handle parentheses, hyphens and numbers well. The structure severely degrades when performing substitution near these environments. The example in Table 3 shows the errors made by the substitution algorithm. An entire subphrase is duplicated using the leaves technique which introduces an opening parenthesis with no closing counterpart and includes the erroneous translation *came*, while TED-R duplicates */ 1999 - 1999 / 2208*.

The reason for these wayward substitutions can be found in the dependency tree. The matching parentheses are not part of the same subtree and this is the root cause of the problem. The leaves technique is very sensitive to these errors and there is no easy way to prevent spurious parentheses from being introduced. Re-ranking in TED-R could filter these hypotheses out, but because the re-ranking module cannot model this dependency, the sentences with these errors are not always discarded. In the manual evaluation campaign, the sentence from Table 3 was included in the sample sentences. It would seem that the many evaluators did not view this error as important or it was ignored. It would be impossible to find the referenced Council decision based on the translations and dates or monetary amounts might change drastically, which would not be acceptable if the translated text should be ready for publishing after translation. For gisting, where the user knows that the translation is not perfect, this may constitute less of a problem.

6 Future work

The initial alignment is based on the source to target language alignment. In the RBMT module, it is mostly word-based while in Moses, the alignment must be recomputed due to the simplicity of the modification and that the Moses chart decoder cannot output word alignment. The modelling only handles alignment crossing one nonterminal and reduces alignment problems to these cases by assuming a weak reordering.

Future work should include extracting the word alignment from the SMT system to improve source to target language alignment. The MT decoder Joshua can output complete derivations including word-based alignment which would eliminate the need to recompute source to target language alignment which currently produces noisy matching elements. Experiments using a different RBMT engine should also be conducted. The RBMT module does not always produce one complete tree structure for a sentence and the reattachment algorithm handles this by adding any additional graphs to the root node of the tree structure. A RBMT engine that produces complete derivations is likely to improve the translation quality. This will require different tree extraction modules for Joshua and the RBMT engine, but otherwise the system can be reused as is.

6.1 Languages and formalisms

The chosen languages are closely related Germanic languages. While the results seem promising, the applicability of the approach should be tested on a more distant language pair, e.g. Chinese-English or Russian-English if you wish to preserve the possibility of using METEOR for evaluation, but any distant pair for which an RBMT system exists can be used — provided a tree output is available.

The implementation substitutes dependency subtrees into a hierarchical CFG-style tree. A second test of the hybridisation approach is to combine systems where the structures are not as diverse. Hierarchical systems are derived from a SCFG so a RBMT system based on a CFG formalism such as LUCY, could be used to test the generality of the hybridisation approach.

As the TED-R approach does not rely on markers for OOV words, an implementation where hierarchical subtrees are inserted into the RBMT output should also be conducted. The problem of inserting CFG-style subtrees into a dependency tree and generating the correct surface structure must be resolved or a different RBMT system which produce CFG-style trees implemented.

The implementation of the leaves technique relies on the diversity of the tree structures, i.e. that there are element-level similarities between hierarchical leaf nodes and both terminal and nonterminal dependency nodes and that the subtree rooted in a dependency node can be aligned to a hierarchical subtree. The refinement method would have to be altered. The relations and children techniques (Shvaiko and Euzenat, 2005) are good candidates for similar tree structures.

A change of formalism would not require alterations of the tree edit distance approach, as long as the structures are in fact tree structures.

7 Conclusion

The post-editing approach proposed in this paper combines the strengths of statistical and rule-based machine translation and improve translation quality, especially for the least accurate translations. The structural and knowledge-poor approach is novel and has not been attempted before. It exploits structural output to create hybrid translations and uses the linguistic knowledge encoded in structure and on nodes to improve the translation candidates of hierarchical phrase-based MT systems.

Automatic evaluation shows a significant increase over the baselines when training data is limited and also improvement in TER and METEOR for lexical substitution and TED-R with a SMT system trained on the Europarl corpus.

Manual evaluation on test data shows that hybrid translations were generally ranked higher, indicating that the hybrid approach produces more accurate translations.

References

S. Banerjee and A. Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. *Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, page 65.

E. Bick. 2007. Dan2eng: Wide-coverage danish-english machine translation. *Proceedings of Machine Translation Summit XI*, pages 37–43.

P. Bille. 2005. A survey on tree edit distance and related problems. *Theoretical computer science*, 337(1-3):217–239.

C. Callison-Burch, C. Fordyce, P. Koehn, C. Monz, and J. Schroeder. 2007. (Meta-) evaluation of machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 136–158. Association for Computational Linguistics.

D. Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270. Association for Computational Linguistics.

A. Eisele, C. Federmann, H. Uszkoreit, H. Saint-Amand, M. Kay, M. Jellinghaus, S. Hunsicker, T. Herrmann, and Y. Chen. 2008. Hybrid machine translation architectures within and beyond the EuroMatrix project. In *Proceedings of the 12th annual conference of the European Association for Machine Translation (EAMT 2008)*, pages 27–34.

C. Federmann, A. Eisele, H. Uszkoreit, Y. Chen, S. Hunsicker, and J. Xu. 2010. Further experiments with shallow hybrid mt systems. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 77–81. Association for Computational Linguistics.

A.S. Hildebrand and S. Vogel. 2008. Combination of machine translation systems via hypothesis selection from combined n-best lists. In *MT at work: Proceedings of the Eighth Conference of the Association for Machine Translation in the Americas*, pages 254–261. Citeseer.

P. Koehn and C. Monz. 2006. Manual and automatic evaluation of machine translation between european languages. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 102–121. Association for Computational Linguistics.

P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 177–180. Association for Computational Linguistics.

P. Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5. Citeseer.

P. Koehn. 2010. An experimental management system. *The Prague Bulletin of Mathematical Linguistics*, 94(-1):87–96.

K. Papineni, S. Roukos, T. Ward, and W.J. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

A.V.I. Rosti, N.F. Ayan, B. Xiang, S. Matsoukas, R. Schwartz, and B. Dorr. 2007. Combining outputs from multiple machine translation systems. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 228–235.

P. Shvaiko and J. Euzenat. 2005. A survey of schema-based matching approaches. *Journal on Data Semantics IV*, pages 146–171.

M. Simard, N. Ueffing, P. Isabelle, R. Kuhn, et al. 2007. Rule-based translation with statistical phrase-based post-editing. In *ACL 2007 Second Workshop on Statistical Machine Translation*.

M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231. Citeseer.

A. Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *Proceedings of the international conference on spoken language processing*, volume 2, pages 901–904. Citeseer.

Gregor Thurmair. 2009. Comparing different architectures of Hybrid Machine Translation systems. In *Proceedings of the MT Summit XII*, pages 340–347.

K. Zhang and D. Shasha. 1989. Simple fast algorithms for the editing distance between trees and related problems. *SIAM J. Comput.*, 18(6):1245–1262.