Incorporating Translation Quality-Oriented Features into Log-Linear Models of Machine Translation

Sergio Penkale

A dissertation submitted in fulfilment of the requirements for the award of

Doctor of Philosophy (Ph.D.)

to the

Dublin City University
School of Computing

Supervisor: Prof. Andy Way

June 2011

Contents

List of Figures

List of Tables

Abstract

The current state-of-the-art approach to Machine Translation (MT) has limitations which could be alleviated by the use of syntax-based models. Although the benefits of syntax use in MT are becoming clear with the ongoing improvements in string-to-tree and tree-to-string systems, tree-to-tree systems such as Data Oriented Translation (DOT) have, until recently, suffered from lack of training resources, and as a consequence are currently immature, lacking key features compared to Phrase-Based Statistical MT (PB-SMT) systems.

In this thesis we propose avenues to bridge the gap between our syntax-based DOT model and state-of-the-art PB-SMT systems. Noting that both types of systems score translations using probabilities not necessarily related to the quality of the translations they produce, we introduce a training mechanism which takes translation quality into account by averaging the edit distance between a translation unit and translation units used in oracle translations. This training mechanism could in principle be adapted to a very broad class of MT systems. In particular, we show how when translating Spanish sentences into English, it leads to improvements in the translation quality of both PB-SMT and DOT. In addition, we show how our method leads to a PB-SMT system which uses significantly less resources and translates significantly faster than the original, while maintaining the improvements in translation quality.

We then address the issue of the limited feature set in DOT by defining a new DOT model which is able to exploit features of the complete source sentence. We introduce a feature into this new model which conditions each target word to the source-context it is associated with, and we also make the first attempt at incorporating a language model (LM) to a DOT system. We investigate different estimation methods for our lexical feature (namely Maximum Entropy and improved Kneser-Ney), reporting on their empirical performance. After describing methods which enable us to improve the efficiency of our system, and which allows us to scale to larger training data sizes, we evaluate the performance of our new model on English-to-Spanish translation, obtaining significant translation quality improvements compared to the original DOT system.

Acknowledgements

I would first like to thank my supervisor, Andy Way, for giving me the opportunity to pursue my Ph.D, and for being not only a great advisor but also a great person who made all aspects of my research enjoyable. Without his constant encouragement, guidance and pursuit of perfection this thesis would not exist. Thanks also to my examiners, Josef van Genabith and Jan Hajic for their valuable feedback.

This thesis would not have been possible either without the generous support from Science Foundation Ireland (SFI) through Grant 07/CE/I1142, as part of the Centre for Next Generation Localisation (CNGL) at DCU. I also wish to acknowledge the SFI/HEA Irish Centre for High-End Computing (ICHEC) for the provision of computational facilities and support.

A highlight of my Ph.D studies were the wonderful four weeks that I spent in Amsterdam working with Khalil Sima'an. I am very grateful to Khalil for his advice, patience and insights, and for his sharing of ideas, without which Chapter 4 would not have been possible.

There are too many people at DCU that I would like to thank. I will not attempt to name everyone as I fear forgetting many, so I would like to extend my thanks to all the students and postdocs at CNGL and NCLT, and to the centre administration staff. Thanks for all the fruitful discussions, which pointed me in the right direction at the early stages, but also for all the good times we had with the "lunch group". Thanks also to Daniel Galron for the great work we did together during his stay in Dublin.

Finally, I would like to thank my family for their unconditional support, in particular to my parents Raúl and Silvia for coping with me being away. The most special thanks of all go to my beloved wife Daniela for always being there, for pushing me to move forward, for bravely enduring the last stages of my studies, and for sharing with me the enjoyment of these exciting phases in our lives.

Chapter 1

Introduction

Machine Translation (MT) deals with the problem of designing algorithms which enable the automatic translation of a sentence from one natural language (the *source language*) into a different language (the *target language*). Current research in MT is led by data-driven approaches, meaning that in a *training* stage, patterns are learnt from a large collection of translation examples, and are used later to translate new sentences.

Most of the focus in current MT research is on defining models which can explain how new translations can be built from the information gathered in the training stage. These models also define probability distributions over the translations they generate, and the translation task consists of searching for the most probable translation. The ways in which translation can be modelled range from word-for-word translation to semantic analysis of the input sentence and subsequent generation of a target-language sentence from this semantic representation. Some of the different modelling possibilities can be summarised as in the pyramid in Figure 1.1. As we move up the left side of the pyramid, the models gain information regarding the meaning and the syntactic structure of the source-language sentence. This enables the models to improve *disambiguation.* Let us consider the following example sentence:

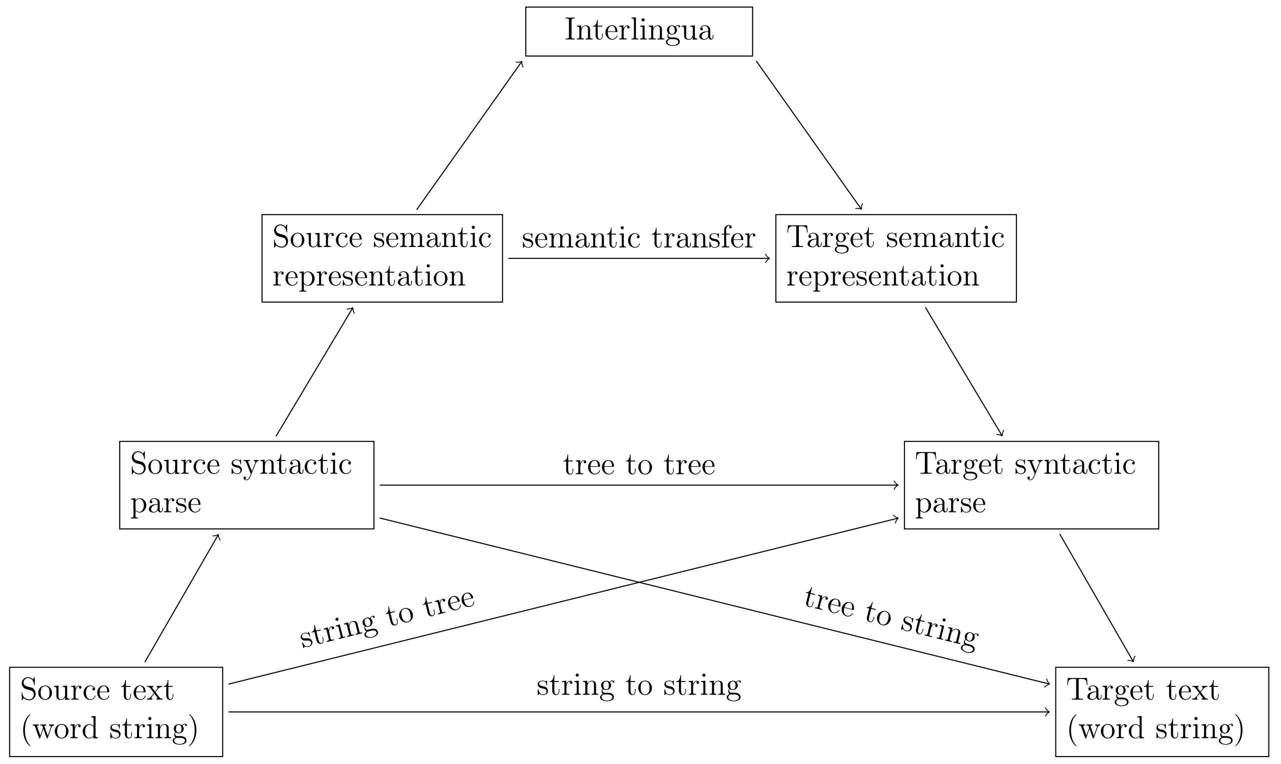


Figure 1.1: Pyramid summarising different approaches to MT. Adapted from (Knight, 1997).

(1) I looked at the kid with the telescope

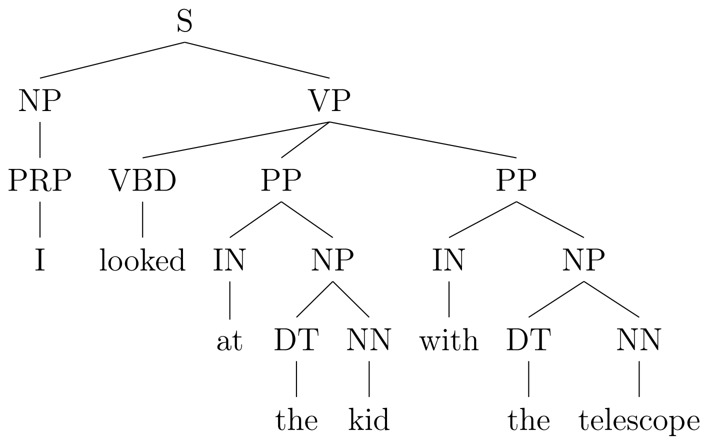
This sentence is ambiguous, as it is uncertain whether what is being said is that *using a telescope* a kid was looked at, or that a kid, who happens to have a telescope, was looked at. When translating this sentence into other languages such as Spanish, there would be two corresponding alternative translations, as in (2):

(2) a. Miré al niño con el telescopio

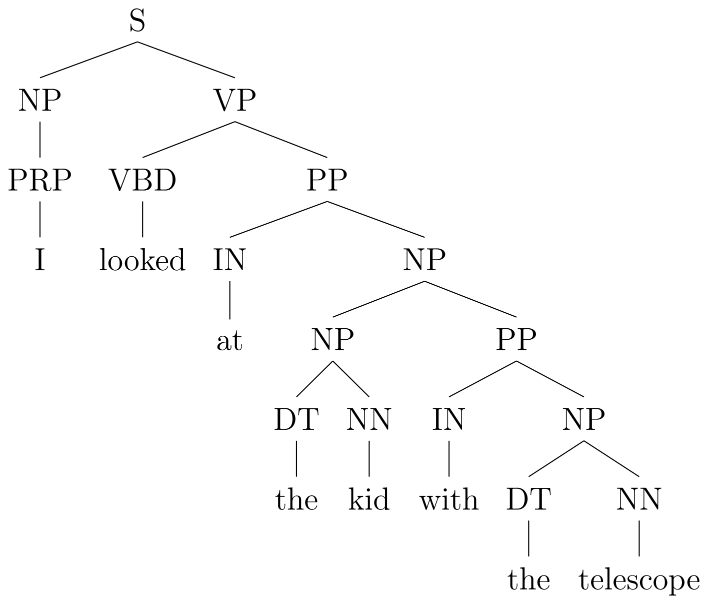
b. Miré al niño que tenía el telescopio

A model which performs word-for-word translation and does not consider the relationship and dependencies between the words on the source sentence might have difficulties distinguishing between these meanings. On the other hand, if a system is allowed to explore more complex structures such as the alternative parse trees in Figure 1.2, the different meanings become evident by considering the attachment of the phrase "with the telescope", and the translation can be chosen according to which of these meanings is more likely.

As regards the opposite side of the pyramid, systems which generate translations based on a representation of the target sentence such as syntax or semantic representations (and which lie higher on the right side of the translation pyramid) will potentially benefit from increased target-language grammaticality. In addition, models which map source syntax representations into their target counterparts are capable of directly modelling differences in word order between the source and target languages. For example, consider the English-French sentence pair in Figure 1.3(a), where the subject in the English sentence is translated as object of the French sentence. A model which performs word-for-word translation would have difficulties capturing such a long-distance movement of target words. In contrast, a model which uses source and target syntax to model translation, and which uses translation rules of the kind in Figure 1.3(b), could directly capture this relation-changing translation.

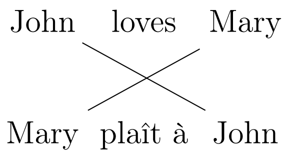


(a) VP-attachment: the PP is modifying the verb phrase

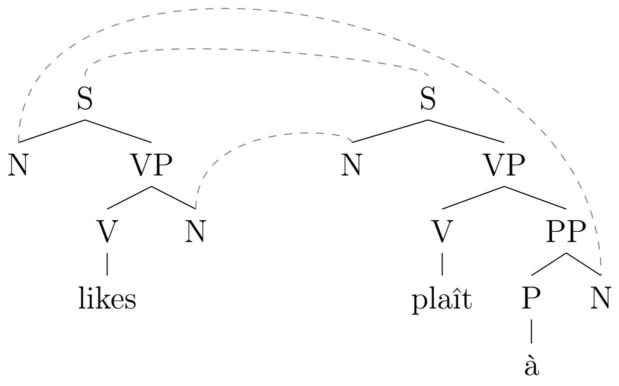


(b) NP-attachment: the PP is modifying the noun phrase

Figure 1.2: Alternative syntactic parses for an English sentence



(a) English and French sentence pair, where links have been drawn to show how the subject of the English sentence becomes the object in the French sentence.



(b) Translation rule which exploits both source and target syntax.

Figure 1.3: English and French sentence pair, and translation rule which captures the relation change by exploiting source and target syntax.

The models which currently achieve state-of-the-art performance (e.g. Phrase-Based Statistical MT (PB-SMT) (Koehn et al., 2003)) remain on the lower part of the translation pyramid, only considering the relationship between source and target words (or phrases), without considering any possible structure over these sequences of words. Given the reasons previously outlined, it is no surprise that the MT research community has been moving towards the goal of increasing source and target syntax-awareness in MT systems.

An underlying assumption in this thesis is that the use of linguistic analysis has the potential to lead to better translations, by helping to disambiguate source sentences and to generate fluent translations. We therefore believe that the way in which the state-of-the-art in MT could be improved is by the use of models which capture both the source and the target syntax in a linguistically motivated framework, using translation rules such as the one in Figure 1.3(b). One such model is Data-Oriented Translation (DOT) (Poutsma, 2000; Hearne and Way, 2003), a tree-to-tree system which translates by composing fragments of source and target parse trees.

Although DOT meets our desired properties of exploiting linguistically motivated source and target syntactic rules, when compared with PB-SMT there are a number of aspects in which the latter stands out, and others in which both could be improved. As we explain in Chapter 3, both DOT and PB-SMT present an inconsistency between the way in which they are trained and the way in which they are evaluated: although the way in which we determine which system is better is by the use of translation quality metrics, these metrics are not directly taken into account when scoring translations to choose between alternative outputs. This inconsistency is present in many other MT systems as well, and gives rise to our first research questions:

RQ1 Can features which relate to expected translation quality be incorporated in MT models?

Criterion	PB-SMT	DOT
Linguistic motivation	✗	✓
Long-distance reorderings	✗	✓
Accuracy-based scoring	$\sim$	✗
Multiple features	✓	✗
Lexical equivalences model	✓	✗

Table 1.1: Summary of the capabilities of PB-SMT and DOT, according to different criteria

RQ2 Can these translation accuracy-based features improve on state-of-the-art MT?

Despite the inconsistencies mentioned above, when it comes to scoring alternative translations, state-of-the-art PB-SMT has a clear advantage over DOT, as the former has the ability to incorporate an arbitrary number of features which increase the sources of information used to assess the potential contribution of a translation rule. In particular, a model explicitly taking into account the relationship between source and target words is used in the scoring. In contrast, DOT is unable to incorporate additional features, having to rely exclusively on the frequency in which its translation rules were observed in the training data. This raises the following further research questions:

RQ3 Can we incorporate new features into the DOT model of translation in such a way that the contribution of each feature can be scaled so as to optimise translation quality?

RQ4 Can we exploit this combination of features by incorporating new ones which lead to increased translation quality?

The comparisons we have made between DOT and PB-SMT are summarized in Table 1.1. In a bid to bridge the gap between DOT and the state-of-the-art approach, in this thesis we address each of the shortcomings in DOT by investigating the research questions we have raised.

1.1 Thesis Structure

The remainder of this thesis is structured as follows.

Chapter 2 provides a review of work which is relevant for the work carried out in this thesis. An overview of different approaches to MT is presented, which is followed by detailed descriptions of the two models most relevant to this thesis: PB-SMT and DOT. The chapter finishes by explaining how the evaluations performed in our experiments were carried out.

Chapter 3 addresses the problem of inconsistency between the objective set for training MT systems and the criteria used to evaluate them. We formulate a general algorithm which is able to take an arbitrary MT system satisfying a few simple assumptions and use it to estimate a feature which relates to the expected translation quality of a particular translation rule. We demonstrate the flexibility of this algorithm by instantiating it both for PB-SMT and for DOT, and assess the impact that it has in the translation quality of the resulting systems by translating from Spanish into English, and comparing against the translations obtained with the baseline systems. We show that for both DOT and PB-SMT, significant gains in translation quality can be obtained using this method.

Having noted that DOT suffers from a limited feature set, we introduce in Chapter 4 a new DOT model which is able to exploit arbitrary features of the source sentence. We take advantage of this new model by defining a novel feature which conditions the choice of each target word to the relevance of the source context to which it is linked. We also explain how efficient decoding for DOT can be achieved, allowing to significantly scale up our training data sizes. We evaluate the performance of our new system by translating English sentences into Spanish, obtaining significant translation quality improvements compared to the original DOT model.

The thesis concludes with Chapter 5, where we summarize the work carried out and how it affects the state of the MT systems we used. We finish by giving our conclusions and avenues for future work.

Much of the contents of this thesis have been published as part of the proceedings of peer-reviewed conferences. The experiments with DOT in Chapter 3 were presented at the *2009 Conference on Empirical Methods in Natural Language Processing (EMNLP 2009)* (Galron et al., 2009). The experiments with PB-SMT in that same chapter were presented at the *Ninth Conference of the Association for Machine Translation in the Americas (AMTA 2010)* (Penkale et al., 2010b). The content in Chapter 4 is currently under review at the *Thirteenth Machine Translation Summit (MT Summit XIII).*

In addition to the aforementioned publications, during the period in which the work for this thesis was carried out the author contributed also in several research activities which have led to publications. In (Srivastava et al., 2009) we examine a number of approaches developed at Dublin City University (DCU) aimed at supplementing the translation model of a PB-SMT system. These approaches have been exploited by the author to contribute to successful DCU participations in international MT campaigns, which led to world-class performance at the *Fourth Workshop on Statistical Machine Translation (WMT09)* (Du et al., 2009) and at the *Fifth Workshop on Statistical Machine Translation (WMT10)* (Penkale et al., 2010a). The author also contributed to the open-source release of some of these approaches under the OpenMaTrEx MT system (Dandapat et al., 2010).

Chapter 2

Related Work

In this chapter we provide an overview of previous work relevant to the work carried out in this thesis, and we detail the metrics which we use to evaluate the impact of our methods.

We begin with a summary of syntax-based approaches to machine translation (MT) in Section 2.1, which is followed in Section 2.2 by a description of PB-SMT, the state-of-the-art approach, and of DOT, our syntax-based system in Section 2.3. As part of our work focuses on integrating translation quality-oriented measures into the scoring of MT systems, we provide an overview of previous approaches to this problem in Section 2.4. Following this, we explain in Section 2.5 the metrics used in our experiments to assess the performance of these systems, as well as the method used to perform statistical significance testing.

2.1 Data-Driven Approaches to MT

Current research in Natural Language Processing, and in Machine Translation (MT) in particular, is dominated by data-driven approaches. Data-driven MT systems exploit large bilingual corpora (*parallel corpora*) created by human translators. These corpora are generally unannotated, meaning that they comprise only a set of source sentences along with their corresponding translations, without any linguistic information associated with the sentences. The first to exploit statistical methods to automatically learn translation equivalences from these parallel corpora were the models of Brown et al. (1988, 1990, 1993). These *word-based* models translate each word in a source-language sentence into a number of target-language words, and then possibly reorder the generated target-language words to obtain a sentence. Although all lexical translation probabilities were based on single words and thus offered limited modelling possibilities, these models were ground-breaking at their time, and still constitute the foundations on which most modern SMT systems are built.

A natural evolution of word-based models was the introduction of Phrase-Based Statistical Machine Translation (PB-SMT) (Och et al., 1999; Marcu and Wong, 2002; Koehn et al., 2003; Och and Ney, 2004), which exploits parallel corpora by extracting a set of bilingual "phrases" (sequences of words not necessarily corresponding to the linguistic notion of a phrase) along with a series of statistics over these bilingual phrases, which are used to disambiguate between candidate translations. The use of phrases as translation units instead of words allows translation equivalences to be induced which use a larger amount of source-side context, and to directly capture local reorderings such as the frequent swapping in adjective and noun placement when translating from Romance languages into English. We present a detailed explanation of PB-SMT in Section 2.2.

2.1.1 Syntax in Statistical Machine Translation

A weakness in pure word- and phrase-based models is that translation equivalences are learnt exclusively by the use of statistical methods over the sentence-aligned parallel corpus, with no linguistic information incorporated at any stage, which can limit the well-formedness of output translations. For example, it has been noted that PB-SMT often produces simple but important errors, such as omitting main verbs in the output translation (Ma and McKeown, 2009). Another weakness in these models is that target-side ordering is strongly driven by the language model score over all possible permutations of target words or phrases within a reordering window (cf. Section 2.2.2). While the local context captured by phrases and the reordering window search might be sufficient to capture local reordering phenomena, it is not possible to model any reordering dependencies which span beyond this reordering window.

A considerable amount of research effort has been made to overcome these limitations, either by explicitly introducing linguistic knowledge, or by exploiting the recursive nature of syntax to be able to model long-distance reorderings. Examples of attempts at enhancing PB-SMT systems by introducing linguistic motivation include the use of parallel treebanks to extract phrase pairs (Tinsley et al., 2007b, 2009), using Word Sense Disambiguation techniques to aid in phrase selection (Carpuat and Wu, 2007), using supertagged language models (Hassan et al., 2007), introducing linguistic annotations as additional factors (Koehn and Hoang, 2007) and exploiting different sources of source-context information (Stroppa et al., 2007; Haque et al., 2009a,b).

Although the aforementioned efforts have in general met with success, an inherent limitation of the framework in which they operate is the incapability of PB-SMT to model long-distance reorderings. To remedy this, different models which allow the time-efficient exploration of discontinuous phrases must be explored. One such alternative model is that of Inversion Transduction Grammar (Wu, 1997), which was later generalized by the Hierarchical Phrase-Based (HPB) model (Chiang, 2005, 2007). The HPB model allows the introduction of translation rules such as (1):

(1) $X \rightarrow \langle X_1 \text{ likes } X_2, X_2 \text{ plaît à } X_1 \rangle$

This kind of rule directly captures the subject and object change when translating the English sentence "John likes Mary" into the French translation "Mary plaît à John". However, rule extraction in HPB is just a generalization of the phrase-extraction method used in PB-SMT, and although it allows the automatic generalization of recursion in language, it is equally linguistically uninformed. As was the case with PB-SMT, attempts have been made at raising linguistic awareness within this model. These include:

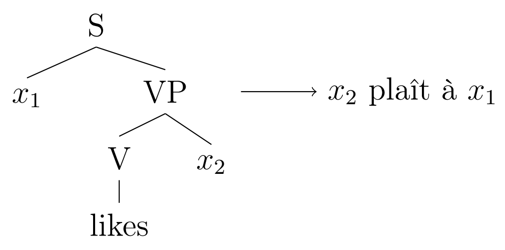
- (Zollmann and Venugopal, 2006), where the labels of the HPB model are augmented using target-side syntactically motivated categories, as well as categories representing incomplete constituents to also permit non-syntactic phrases,
- (Marton and Resnik, 2008), where features are introduced which reward (or penalize) the model score when the span of the source side of a rule matches (or does not match) a constituent with a specific label in the parse tree of the source sentence,
- (Vilar et al., 2008), where the rule extraction phase is modified so that the count used to compute the scores for each rule is dependent on the percentage of words in the phrase that match a constituent,
- (Chiang, 2010), where rules are extracted along with source and target syntax information, using complex categories such as those used in (Zollmann and Venugopal, 2006), and features are introduced which are activated when the left-hand side of a rule used in a derivation matches the label of the rule in which it is being substituted,
- (Haque et al., 2010), where supertag features are introduced which encode complex syntactic information about the source side of the rules.

Although these improvements lead the baseline HPB model towards the desired properties outlined in Chapter 1, a limitation of this framework is its context-free nature. The work in (Chiang, 2010) does bring linguistically motivated source and target syntax to HPB, but the internal structure of rules is ignored, with consideration given only to the roots and leaves of sub-trees, and restrictions placed on the amount of nodes in the rules. In contrast, the STSG models we discuss below allow the modelling of rich rules which encode subtrees of arbitrary size on both source and target languages.

Outside of the framework of the HPB paradigm, models which explicitly exploit the syntactic relationships present in syntactically parsed corpora have also been proposed. These models are usually classified according to whether they exploit syntactic information only on the source-language side (*tree-to-string* models), only on the target side (*string-to-tree* models), or on both sides (*tree-to-tree*).

Yamada and Knight (2001) define a model in which a source-language parse tree is transformed into a target-language sentence, by using stochastic operations at each node. This model, which can be framed as a tree-to-string transducer (Graehl and Knight, 2004), takes a parse tree as input and leads to a tree-to-string alignment. However, this is then used in a noisy-channel-based decoder (Yamada and Knight, 2002), which models translation in the reverse direction: for every input string, the model is used to explore all trees which could have generated the input. Since in practice this leads to syntax being used to model the target side, decoders based on this approach are generally regarded as string-to-tree in the MT literature. A related approach is that of Galley et al. (2004, 2006), who present a tree-to-string rule extraction method which leads to a set of translation rules which explain a tree-to-string training corpus. The rule extraction algorithm takes a word-aligned parallel corpus which has been parsed on its source side, and obtains translation rules such as (2):

(2)



This line of research is also followed by Huang et al. (2006), who extend the model to a log-linear framework which is able to exploit additional features such as an n-gram language model, and which implements a direct tree-to-string decoder, thus using syntax to model the source language in this case. Source-language syntax is also exploited by Quirk et al. (2005), who employ a source-side dependency parser to obtain a treelet translation model, where a treelet is “an arbitrary connected subgraph of the dependency tree”, which is later used to cover the dependency parse of an input sentence.

Although a priori the most promising approach would be that of exploiting both source- and target-language syntax for translation, tree-to-tree models suffer both from increased complexity, as complex structures for both source and target sides have to be maintained when building translations, and increased language-specific prerequisites, as both source- and target-language parsers must be available. This has slowed the rate at which tree-to-tree models have evolved, and perhaps with the exception of (Chiang, 2010), it has also led to little success when compared to simpler models such as PB-SMT (Liu et al., 2009). As noted by Way (2009), the influential articles by Brown et al. (1988, 1990, 1993) did not preclude the use of linguistics in statistical MT:

> it is not our intention to ignore linguistics, neither to replace it. Rather, we hope to enfold it in the embrace of a secure probabilistic framework so that the two together may draw strength from one another (Brown et al., 1993)

If after the introduction of these statistical models, SMT researchers and linguists had worked together more closely, the current interest in syntax-based models would have arisen long ago, and these kinds of system would have matured sooner (Way, 2009).

Other tree-to-tree approaches include (Ding and Palmer, 2005), where a translation model based on dependency structures is defined, and that of Cowan et al. (2006), who extract syntactic structures with alignment information from a parallel corpus that has been bilingually-parsed, and use a perceptron-based model for scoring.

A particularly promising tree-to-tree modelling approach is that of Synchronous Tree Substitution Grammars (STSG) (Hajič et al., 2004; Shieber, 2004) as, unlike the Synchronous Context Free Grammars (SCFG) of the HPB model, they allow an arbitrarily large source and target syntactic context to be captured. Models which use this formalism include Data-Oriented Translation (DOT) (Poutsma, 2000; Hearne and Way, 2003), which assumes a sub sententially-aligned parallel corpus. We will cover DOT in detail in Section 2.3. Models which extend the tree-to-string extraction method of Galley et al. (2004) to the tree-to-tree case also fall within the STSG framework. These approaches include (Zhang et al., 2008), where a tree-sequence model is defined (i.e. rules can consist of more than one tree on either side), (Liu et al., 2009) where the impact of parsing errors is lessened and rule coverage is improved by extracting rules from multiple parse trees (tree forests) instead of single trees, and the previously mentioned work of Chiang (2010). DOT differs from these approaches in that:

- Sub-sentential alignments in DOT are obtained by a sub-tree aligner (Tinsley et al., 2007a; Zhechev and Way, 2008) in which the use of a word alignment over the parallel corpus is avoided, unlike in (Galley et al., 2004). This has the advantage that words in the parallel treebank can be aligned based on information from nodes higher on the tree, where more context is available (Tinsley et al., 2007a), and that errors in the word alignments are not propagated to the sub-tree alignments, which leads to higher quality alignments (Tinsley et al., 2007b).

- In DOT, all possible rules are extracted from the training corpus. Unlike in Chiang (2010), differences in the internal structure of rules are not ignored, and no restrictions are placed on the amount of nodes in a rule or on the depth of derivations unlike in the above-mentioned tree-to-tree approaches.

Other tree-to-tree models based on the STSG formalism include (Eisner, 2003; Hajič et al., 2004; Bojar and Hajič, 2008; Bojar et al., 2009), where in the context of translating between Czech and English by exploiting the Prague Dependency Treebank (Hajič, 2004), deep syntactic transfer is performed by analysing an input sentence up to the tectogrammatical layer and performing source-to-target transfer of the tectogrammatical representations, to finally generate the target sentence from the target tectogrammatical representation. Unlike in DOT, which assumes sub-sentential alignments in the training corpus, Eisner (2003) attempts to automatically learn a translation model from unaligned data by using the EM algorithm to explore all possible derivations.

As regards the problem of reordering in MT, an alternative approach to those previously mentioned is that of exploiting syntax to reorder the source sentence in a preprocessing stage, so as to mimic the word order of the target language, e.g. (Xia and McCord, 2004; Collins et al., 2005; Wang et al., 2007; Khalilov, 2009). In these approaches a set of rules is used to change the order of words in an input sentence so that the translation can be found in a monotonic way. These approaches can be further sub-categorized according to whether reordering rules are manually created, e.g. (Collins et al., 2005; Wang et al., 2007), or automatically learnt from translated data, e.g. (Xia and McCord, 2004; Khalilov, 2009). These approaches have the advantage of leading to relatively faster systems than those which attempt to model reorderings during the translation phase. However, the source-side reordering usually remains fixed or at best constrained, which limits the reordering possibilities compared to those which would be explored by a system which explicitly models long-distance reorderings during translation.

2.2 Phrase-Based Statistical Machine Translation

Since its introduction, PB-SMT has been the predominant paradigm in MT research. PB-SMT is a *string-to-string* translation model, i.e. sequences of lexical tokens from a source-language sentence are directly mapped to target-language lexical tokens, with no underlying structure associated with the sentences. Despite this relative simplicity, the most widely used form of PB-SMT (Koehn et al., 2003, 2007) continues to achieve state-of-the-art performance for many language pairs, and particularly

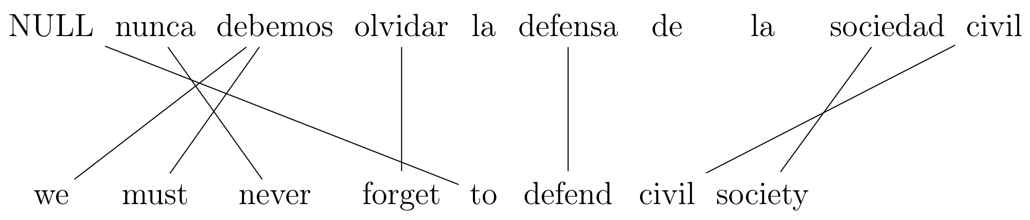


Figure 2.1: Automatic Spanish-to-English word alignment for a sentence pair taken from the Europarl corpus. Since this is a directional alignment, an alignment for the reverse language direction must be obtained before extracting phrase pairs.

amongst European language pairs (Callison-Burch et al., 2010).

2.2.1 Phrase Extraction

PB-SMT uses *phrase pairs* as the basic units of translation. A phrase pair $\langle f, e \rangle$ consists of a sequence of source-language words f along with its corresponding target-language translation e. As a result of the phrase-extraction method used to automatically obtain them, neither source nor target phrases necessarily correspond to the linguistic notion of a phrase.

Phrase pairs are obtained with the aid of a word-aligned parallel corpus, which is in turn obtained using models developed for word-based translation systems (Brown et al., 1990, 1993). These so-called *IBM models* are unsupervised generative models which aim at estimating the probability that a source-language word translates as a particular target-language word. As a by-product they produce a word alignment which relates each target word in a sentence to the source word it was generated from. Figure 2.1 gives an example word alignment between a Spanish and an English sentence which were taken from the Europarl corpus (Koehn, 2005). Links in this figure indicate the source word from which each target word arose. Any target-language word which is not related to a source word is aligned to the special source-language word “NULL”. As the model only explains how *target* words were generated, unused source words are left unaligned. The alignment used in this figure was obtained using the freely available tool GIZA++ (Och and Ney, 2003), using IBM

debemos	we must
nunca debemos	we must never
nunca debemos olvidar	we must never forget
nunca debemos olvidar la	we must never forget
defensa de la sociedad civil	to defend civil society
olvidar	forget
olvidar la	forget
olvidar la defensa de	forget to defend
la sociedad civil	civil society

de	to
defensa de	to defend
la defensa de	to defend
defensa	defend
nunca	never
la defensa	defend
civil	civil
la sociedad	society

Figure 2.2: Phrase pairs extracted from the word-aligned sentences in Figure 2.1, after obtaining word alignments for both language directions and performing the heuristic merge.

model 4. This model generates the target-language sentence by first conditioning each word in the source sentence to the number of words in the target sentence that it will generate (referred to as the *fertility* of the source word), and then the particular target-language words to be generated, and finally the positions in the target-language sentence in which these words will be placed (the model which accounts for target-language reordering of words is referred to as the *distortion* model). The probability of an alignment depends on:

- the source and target positions that it connects,
- the lengths of the source and target sentences,
- the particular source and target words being aligned,
- and the positions of any other target words aligned to the same source word (Brown et al., 1993).

Phrase-Based SMT aims at relating sequences of contiguous source words (source phrases) to their corresponding target-language translation (target phrases). However, using the IBM models it is only possible to relate a target word to an individual source word. To obtain the required many-to-many relationships, it is common practice to obtain word alignments for both the source-to-target and the target-to-source language directions, and to later combine these alignments by obtaining a compromise between the intersection of the alignments (which results in a set of highly accurate phrase alignments) and the union (which results in a larger set of phrases which increase the coverage of the system, although possibly with a lower accuracy). Koehn et al. (2003) investigate multiple heuristics to perform this merging, and find that the heuristic they call "diag-and" gives the best performance. All phrase pairs consistent with this combined word alignment are then extracted. A phrase-pair is considered consistent if all of its source words are aligned only to words in the target phrase, and vice versa (Och et al., 1999). Figure 2.2 lists all phrases that can be extracted from the sentence pair in Figure 2.1, after performing the word alignment in the opposite direction and the heuristic merge. This example shows phrase pairs of up to 5 words in length, although in our experiments we use the standard length of 7 words during phrase extraction.

2.2.2 Decoding

The process of determining the target-language sentence that maximizes the model score given a source-language sentence is known as *decoding*. To perform translation, an input sentence $s = w_1 \ldots w_n$ composed of n words is segmented into I phrases $f_1^I = f_1 \ldots f_I$. Using the set of phrase pairs that were extracted during training, which are stored in a *phrase table*, each source phrase f_i in f_1^I is translated into a corresponding target phrase e_i, resulting in a target sentence $t = e_1^I$.[1]

During the search for the best-scoring translation, all possible segmentations of the input sentence are considered, and target sentences are generated left-to-right. Source phrases might be translated in any order, allowing in this way for reordering to take place in the target sentence. However, most implementations place a limit (typically, 5 to 8 words) on the amount of words that can be skipped between the previously translated phrase and the following phrase. This limit, called the *reordering window*, allows us to avoid an exponential growth in the amount of alternative translations that are considered, and is key to lowering decoding complexity

[1]The notation used is as follows: we use the letters s and t when referring to source and target *sentences*, the letters f and e in the context of *phrases*, and the letter w when considering individual *words*.

to linear on the length of the input string.

Partial candidate translations (or *hypotheses*) are stored in stacks, which maintain a beam of k-best hypotheses by discarding lower-scoring ones. There are n such stacks, where hypotheses are grouped according to the amount of source words covered so far in translating the input sentence.

The decoding algorithm used by standard PB-SMT is able to output not only the best translation according to the model, but also the k-best translations. In addition, we assume that the decoder is able to keep track of the phrases used to build each of the candidate translations. For each output translation in the k-best list, this will allow us to obtain an alignment a. For each target phrase e_i used to generate each of the candidate translations in the k-best list, this alignment specifies a pair of integers $a(e_i) = (l, m)$ which indicate that the target phrase e_i is translated from the source sentence span $w_l \dots w_m$.

2.2.3 Scoring

To select among the many phrase translation options, and among the possible input sentence segmentations into phrases, the target sentence t that maximises $P(t|s)$ is chosen. In state-of-the-art PB-SMT this posterior probability is modelled directly by a log-linear model (Och and Ney, 2002) as in (2.1):

$$P(t|s) = P(e_1 \dots e_I | f_1 \dots f_I) = \exp(\sum_{i=1}^{M} \lambda_i h_i(e_1^I, f_1^I)) \tag{2.1}$$

Here each $h_i(e_1^I, f_1^I)$ is a feature function and each λ_i the corresponding feature weight. These weights are estimated by Minimum Error Rate Training (MERT) (Och, 2003). This is an efficient procedure which maximizes the score of a particular translation quality metric (typically BLEU (Papineni et al., 2002)) on a small held-out set.

The log-linear model of equation (2.1) is a framework in which arbitrary features can be incorporated. The standard features present in most PB-SMT systems are:

- The product of the conditional phrase translation probabilities $p(f_i|e_i)$ and $p(e_i|f_i)$, for all phrase pairs $\langle e_i, f_i \rangle$ involved in the sentence and its translation. These probabilities are estimated using relative frequency over the multiset of phrases extracted from the parallel corpus, as in (2.2):

$$p(f_i|e_i) = \frac{\text{count}(f_i, e_i)}{\sum_{f'} \text{count}(f', e_i)} \tag{2.2}$$

- An n-gram language model over the target translations, which is usually estimated using improved Kneser-Ney smoothing (Chen and Goodman, 1998).

- A "lexical weighting" feature for each language direction. This is used to smooth phrase translation probabilities by considering how often words within a phrase pair were aligned in the parallel corpus. For each phrase pair extracted from the word-aligned corpus, an alignment a is also extracted, which contains a tuple (l, j) for every target word t_l aligned to a source word s_j within the phrase pair (f_i, e_i). The lexical weight is computed as in (2.3):

$$\text{lex}(f_i|e_i, a) = \prod_{l=1}^{\text{len}(e_i)} \frac{1}{|\{j|(l,j) \in a\}|} \sum_{\forall (l,j) \in a} w(t_l|s_j) \tag{2.3}$$

  where $w(t_l|s_j)$ is estimated by the frequency whereby t_l was aligned to s_j in the word-aligned parallel corpus, relative to the frequency of any other target word aligned to s_j (Koehn et al., 2003).

- Penalty features which count the amount of words and phrases in the candidate translation.

2.2.4 Reordering Model

As previously mentioned, source phrases can be translated in any order (although the reordering window limits the amount of source words that can be skipped between a phrase and the next). State-of-the-art PB-SMT incorporates lexicalised

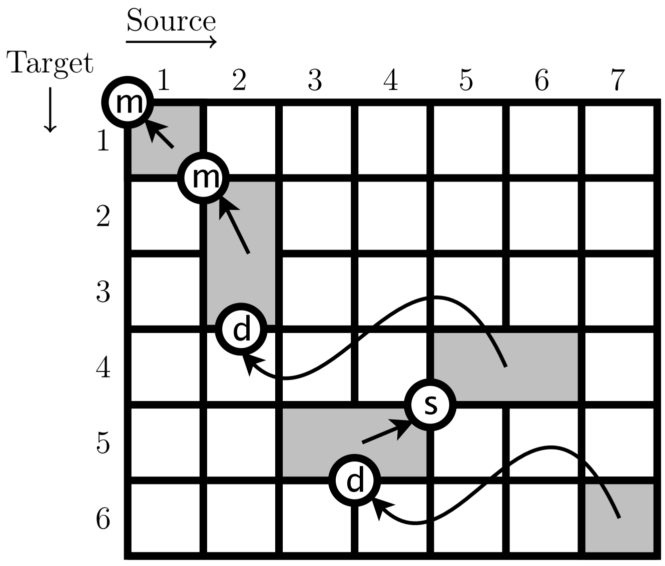


Figure 2.3: Lexicalized reordering orientations: monotone (m), swap (s) and discontinuous (d). Adapted from Koehn (2009) (original Figure 5.8, page 143).

reordering features (Koehn et al., 2005). Once a source phrase has been translated, the location in the source sentence from which the next phrase to be translated is taken is influenced by the score assigned by these features. To avoid data sparseness problems, the features model only three kinds of *orientations* between a phrase and the previously translated phrase. The modelled orientations (illustrated in Figure 2.3) are: monotone (a phrase directly follows the previous phrase), swap (a phrase is swapped with the previous phrase) and discontinuous (neither monotone nor swap). In the example in Figure 2.3, the second target phrase (which spans the second and third target words) translates the source word directly adjacent to the word translated by the first target phrase, and therefore receives a monotone orientation. Relative to the second target phrase, the third target phrase translates source words which are neither adjacent nor swapped, and is therefore considered a discontinuous translation. Finally, the fourth target phrase is swapped relative to the previous phrase. When extracting phrases from the parallel corpus, the orientation used by each phrase is also extracted. Counts of these events are obtained, which are used to estimate the probability of an orientation given a phrase pair. Analogous features which consider the following phrase (as opposed to the previously translated phrase) are also introduced.

2.3 Data-Oriented Translation

DOT (Poutsma, 2000; Hearne and Way, 2003, 2006), which was inspired by the the Data-Oriented Parsing (DOP) model (Bod, 1992), potentially overcomes the shortcomings in PB-SMT by explicitly modelling syntax, both in the hierarchical structure and the linguistic sense.

2.3.1 Parallel Treebanks

As a training corpus, DOT assumes a *parallel treebank*, which consists of a parallel corpus (a set of sentences with their corresponding translation) where each sentence has been augmented with its parse tree, and each pair of nodes in the source and target trees whose yields convey the same meaning have been aligned (Volk and Samuelsson, 2004). Figure 2.4(a) gives an example English and French sentence pair which has been syntactically parsed and sub-sententially aligned.

Initial experiments with DOT were carried out using manually-aligned corpora (Poutsma, 2000; Hearne and Way, 2003). Although this ensures that training examples are obtained from a high-quality source, requiring the use of manually-aligned corpora either prohibitively raises the costs of training new systems, or limits training data-set sizes and domains to the few existing parallel treebanks, e.g. (Han et al., 2002; Čmejrek et al., 2004; Hansen-Schirra et al., 2006; Ahrenberg, 2007; Megyesi et al., 2008; Volk et al., 2010). Fortunately, over the past few years major improvements have been made in the area of automatic parallel treebank generation. In particular, in this thesis we automatically build parallel corpora for DOT by exploiting the automatic tree aligner introduced in (Tinsley et al., 2007a) and extended in (Zhechev and Way, 2008). This is an efficient greedy-search algorithm which allows all possible alignments to be considered. Using initial word-alignment probabilities (obtained using GIZA++, cf. Section 2.2.1), it iteratively selects the highest-scoring alignment, discarding all other alignments that conflict with it. As explained by Zhechev (2009) and Tinsley (2010), the advantages of this approach include:

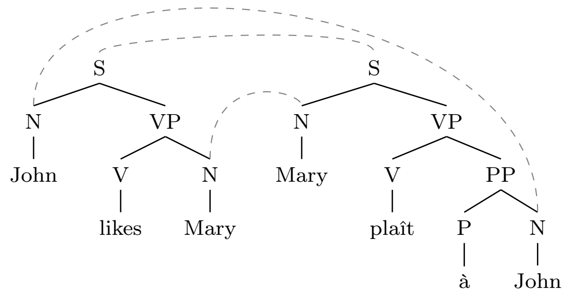
- independence of language pair,
- independence of linguistic representation,
- computational efficiency,
- preservation of the tree structures,
- minimal requirement for external resources.

Although such an automatic generation process will inevitably lead to alignment errors in the parallel treebank, Tinsley et al. (2007a) and Zhechev and Way (2008) report good precision and recall figures when directly evaluating the alignments they obtain, and improvements in Machine Translation evaluation metrics when using this algorithm to train a DOT system, compared to the results obtained when training with a manually generated parallel treebank.

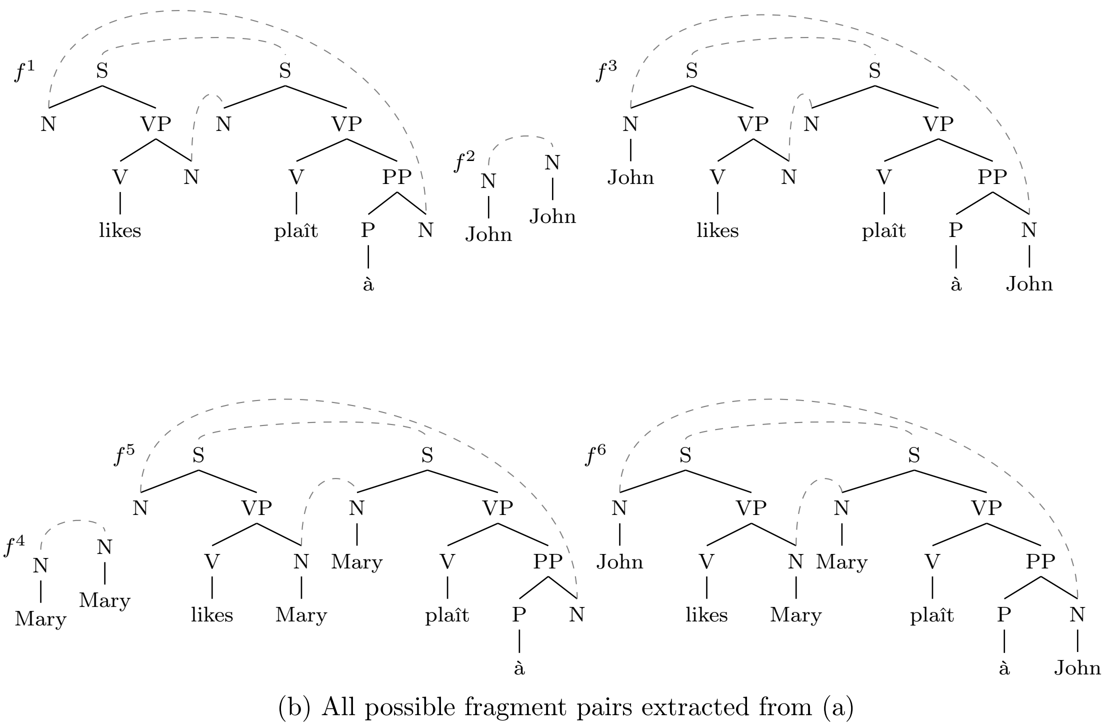
2.3.2 Fragment Extraction

The basic units of translation in DOT are subtree pairs (or *fragment pairs*). The process by which each fragment pair can be extracted from a tree pair in the parallel treebank is described in terms of the *root* and *frontier* operations. Given a tree pair in the parallel treebank, we can extract a new fragment pair by the following process:

1. Choosing a *linked* node of the original tree pair to be the root of the new fragment pair. We then create a new fragment pair which initially consists of this node pair along with the source and target subtrees they dominate, and with links between source and target nodes as in the original tree pair. For example, if choosing as new root pairs the nodes labelled "S" from the tree pair in Figure 2.4(a), our new fragment, which we will modify in the next step, initially consists of an exact copy of the original tree pair.



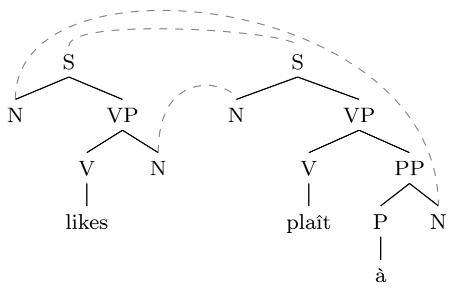
(a) Sub-sententially aligned tree pair



(b) All possible fragment pairs extracted from (a)

Figure 2.4: Example tree and fragment pairs

2. Choosing a (possibly empty) set of *linked* nodes on the new fragment pair and removing from our newly created fragment pair source and target descendants of nodes in this set. For example, if from our newly created fragment pair we choose the two source nodes labelled "N" (along with their target-side counterparts), after removing descendants of these nodes we obtain the following fragment pair:



Nodes with no descendants are called *frontier nodes*, and linked nodes which are frontier nodes are called *substitution sites*, e.g. the nodes labelled "N" in the above fragment pair. Note that we assume that only non-terminals nodes can be linked in the original tree pair.

Figure 2.4 further illustrates this fragment-pair extraction process, by listing in (b) all the fragment pairs that can be extracted from the tree pair in (a).

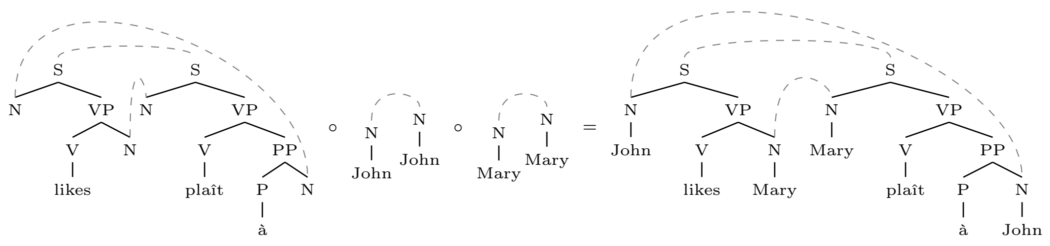
2.3.3 Composition

Fragment pairs can be combined by using the DOT *composition* operator, denoted $\circ$. To ensure that each derivation is unique, we define composition in terms of the *leftmost* substitution site. A fragment pair $\langle f_1, e_1 \rangle$ can be composed with a fragment pair $\langle f_2, e_2 \rangle$ if three conditions are met:

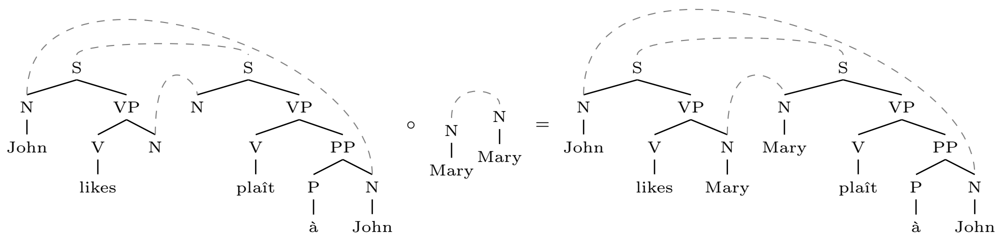
a) $\langle f_1, e_1 \rangle$ has at least one substitution site,

b) the label A on the *leftmost* substitution site in f_1 (e.g. the leftmost N in fragment pair f^1 in Figure 2.4) equals the root label in f_2,

c) the label of the target-side node linked to A (e.g. the rightmost N in fragment pair f^1) equals the root label in e_2.



(a) Derivation involving three fragment pairs



(b) Different derivation for the same string pair, using only two fragment pairs

Figure 2.5: Two different DOT derivations for the same string pair

The resulting fragment pair consists of a copy of $\langle f_1, e_1 \rangle$ where, on the source side A has been replaced by f_2, and on the target side the node linked to A has been replaced by e_2 (e.g. fragment pair f^3 in Figure 2.4).

A sequence of fragment compositions is called a *derivation*. When we have such a sequence of composition operations, we will follow the convention that composition is left-associative. Figure 2.5(a) shows fragment pairs which are involved in a derivation of the English “John likes Mary” which results in the French translation “Mary plaît à John”. It is important to note that there might be more than one derivation yielding the same source and target sentences. For example, if we were to begin a new derivation by composing the fragment pair f^3 in Figure 2.4 with fragment pair f^4, we would obtain a different (shorter) derivation for the same string pair, as shown in Figure 2.5(b).

2.3.4 Scoring

During training, we extract *all* possible fragment pairs from the parallel treebank. By considering $|\langle d_f, d_e\rangle|$, the count of times that the fragment pair $\langle d_f, d_e\rangle$ was extracted from the parallel treebank, we assign to each fragment pair a probability equal to the frequency in which it was extracted, relative to the frequency in which fragment pairs with the same root pairs were extracted, as in (2.4):

$$P(\langle d_f, d_e\rangle) = \frac{|\langle d_f, d_e\rangle|}{\sum\limits_{\{\langle u_f, u_e\rangle | \mathrm{root}(u_f)=\mathrm{root}(d_f) \wedge \mathrm{root}(u_e)=\mathrm{root}(d_e)\}} |\langle u_f, u_e\rangle|} \tag{2.4}$$

Then, with the assumption that fragment pairs are composed conditionally independently of each other, the probability of a derivation is the product of the probabilities of the fragments it is composed of, as in (2.5):

$$P(\mathbf{d}_{\langle s,t\rangle}) = P(\langle d_f, d_e\rangle_1 \circ \ldots \circ \langle d_f, d_e\rangle_N) = \prod_{i=1}^{N} P(\langle d_f, d_e\rangle_i) \tag{2.5}$$

Given an input source-language sentence s,[2] DOT searches for all possible derivations resulting in a tree pair whose source-side yield is equal to s. The yields of the target-side trees in these derivations constitute the set of candidate translations, from which the output translation is chosen so as to maximize the (string) *translation probability* in (2.6):

$$P(s,t) = \sum_{\mathbf{d}_{\langle s,t\rangle} \in \mathbf{D}} P(\mathbf{d}_{\langle s,t\rangle}) \tag{2.6}$$

Hearne and Way (2006) explored the effects of using criteria different to the translation probability in order to choose the final translation, such as the most probable derivation or the shortest derivation, and found that in some cases translation quality remains the same or improves when using the shortest derivation or the most probable derivation, compared to using the most probable translation. In this

[2]To avoid confusion, we use a notation similar to the one used in Section 2.2: we use the letters s and t when discussing source and target *sentences*, and the letters f and e in the context of *fragments*.

work we experiment with using the most probable translation and the most probable (Goodman) derivation (cf. next section) methods to select the output translations.

2.3.5 Efficient Fragment Representation

As previously mentioned, DOT extracts all subtrees in the parallel treebank. Since the amount of subtrees grows exponentially with the size of the training corpus trees, this presents computational challenges because enumerating all subtrees in a reasonably-sized corpus is intractable. Fortunately, there are efficient representations for such collections of trees. Goodman (1996) introduced a method to reduce a DOP model to a Probabilistic Context Free Grammar (PCFG) yielding the same trees with the same probabilities, and Hearne (2005) extended this reduction to the bilingual case of DOT. Although in (Hearne, 2005) the complete details needed to compute bilingual translation probabilities using the Goodman reduction for DOT are not given, Hearne (p.c.) did manage to obtain an efficient mapping from source-language trees to their corresponding target-language counterparts, which we now explain.

The process of obtaining a Goodman PCFG grammar for DOT involves preprocessing the parallel treebank in many ways. Firstly, the reduction assumes the parallel treebank to be in Chomsky Normal Form (CNF), i.e. all nodes except those directly dominating a terminal leaf must have exactly two children, while nodes directly dominating terminal leaves (*pre-terminal* nodes) must have exactly one child. Since this is not necessarily true for a general parallel treebank, we binarize both source and target trees in the training corpus. Throughout this thesis, we use *right binarization*, as explained in Figure 2.6, although any standard binarization method could be used.

Once the parallel treebank has been binarized, we modify it in two ways:

a) For every pair (X, Y) of *linked* nodes in the treebank, we replace both labels X and Y with the new label "X=Y".

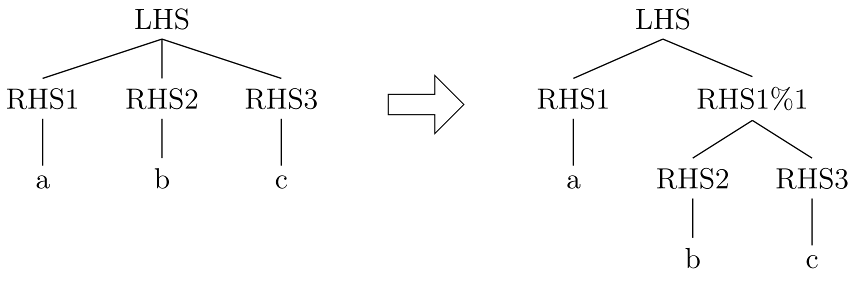


Figure 2.6: Right binarization of a non-CNF parse tree. The non-CNF tree in the left is converted to the CNF tree in the right by introducing a new binarization node which contains the special symbol % and a binarization index. Binarization nodes are left unaligned.

b) We then annotate the label of each node in the parallel treebank with a unique index (its *Goodman index*).

We note that when performing fragment extraction, each binary-branching node and its two children can either be: an internal (i.e. neither root nor frontier) node in a fragment, a fragment root, or a fragment frontier. Accordingly, we add rules to the grammar reflecting the role that each node can take, keeping unaligned nodes and nodes introduced during binarization as fragment-internal nodes. Keeping these considerations in mind we generate two PCFG grammars, a source and a target one. Rules on the grammars and their associated probabilities are generated as in the standard Goodman reduction for monolingual DOP (Goodman, 1996). For the case where a node and both of its children are aligned, as in Figure 2.7:

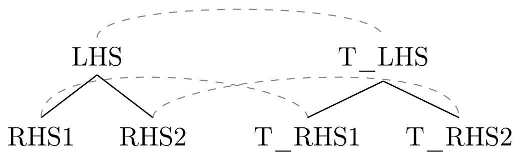


Figure 2.7: Fully-external fragment pair

we add 8 rules with their corresponding probabilities to the source grammar, as follows:

$$\begin{array}{llll} \text{LHS} \rightarrow \text{RHS1 RHS2} & 1/a & \text{LHS+j} \rightarrow \text{RHS1 RHS2} & 1/a_j \\ \text{LHS} \rightarrow \text{RHS1+k RHS2} & b_k/a & \text{LHS+j} \rightarrow \text{RHS1+ RHS2} & b_k/a_j \\ \text{LHS} \rightarrow \text{RHS1 RHS2+l} & c_l/a & \text{LHS+j} \rightarrow \text{RHS1 RHS2+l} & c_l/a_j \\ \text{LHS} \rightarrow \text{RHS1+k RHS+l} & b_k c_l/a & \text{LHS+j} \rightarrow \text{RHS1+k RHS2+l} & b_k c_l/a_j \end{array} \tag{2.7}$$

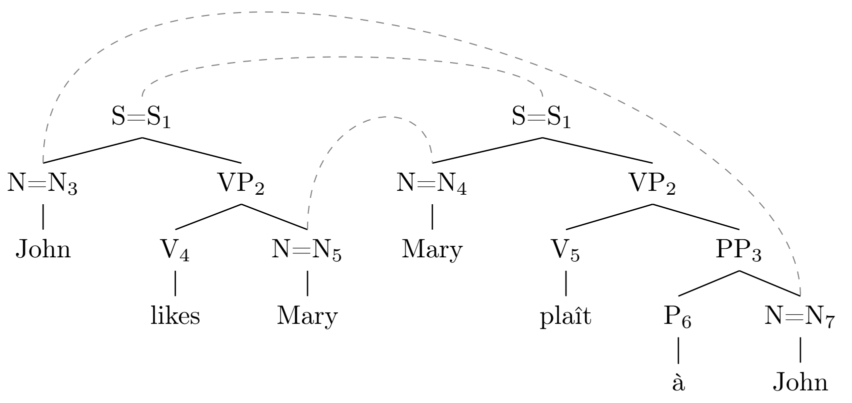
where a_j represents the number of subtrees headed by the node LHS with Goodman index j (respectively b_k for RHS1 with index k and c_l for RHS2 with index l),[3] and a is the number of subtrees headed by nodes with non-terminal LHS (i.e. $a = \sum_j a_j$, with analogous definitions for b and c).

A category label which ends in a '+' symbol followed by a Goodman index is fragment-internal and all other nodes are either fragment roots or frontier nodes. In this representation, a fragment pair is then a pair of subtrees in which the root does not have an index, all internal nodes have indices, and all the leaves are either terminals or unindexed nodes. We give an example Goodman PCFG reduction in Figure 2.8, where the tree pair is shown after the appropriate modifications have been carried out.

We store the source and target grammars separately, and keep track of the alignment correspondences between source and target Goodman indexes. This means that when parsing a source sentence using the source Goodman grammar, for each source rule which contains a Goodman index we can always uniquely identify the corresponding target rule and thus build the translations using the target PCFG grammar.[4] To handle the case where a source derivation contains an *external rule* (a binary rule in which the LHS, the RHS1 and the RHS2 are aligned, i.e. as in Figure 2.7), we also maintain a list of source external rules along with their alternative corresponding target-side rules. The method of annotating linked nodes in the

[3]As noted by Goodman (1996), a_j can be efficiently computed by the recursive formula $a_j = (b_k + 1)(c_l + 1)$.

[4]In practice we only need to identify the root and substitution sites of each source fragment, along with one of the Goodman indexes in its highest level rule. Using this information and the alignments between source and target Goodman indexes, we can deduce the corresponding target fragment.



(a) A tree pair where linked nodes have been relabelled to include both source and target labels, and all nodes have been indexed.

Source PCFG		Target PCFG	
S=S → N=N VP+2	0.5	S=S → N=N VP+2	0.5
S=S → N=N+3 VP+2	0.5	S=S → N=N+4 VP+2	0.5
S=S+1 → N=N VP+2	0.5	S=S+1 → N=N VP+2	0.5
S=S+1 → N=N+3 VP+2	0.5	S=S+1 → N=N+4 VP+2	0.5
N=N → John	0.5	N=N → Mary	0.5
N=N+3 → John	1	N=N+4 → Mary	1
VP+2 → V+4 N=N	0.5	VP+2 → V+5 PP+3	1
VP+2 → V+4 N=N+5	0.5	V+5 → plaît	1
V+4 → likes	1	PP+3 → P+6 N=N	0.5
N=N → Mary	0.5	PP+3 → P+6 N=N+7	0.5
N=N+5 → Mary	1	P+6 → à	1
		N=N → John	0.5
		N=N+7 → John	1

(b) Source and target Goodman grammars corresponding to the tree pair in (a).

Figure 2.8: A parallel tree and its corresponding Goodman reduction.

source grammar with their target-side counterparts ensures that (with the exception of derivations which contain source external rules), we can compute bilingual derivation scores using just the score determined by the source grammar rules. To obtain bilingual derivation probabilities for the general case of derivations which might contain source external rules, we first multiply, for every source external rule in a derivation, the relative frequency probability of the target-side rule given the source external rule, and finally multiply this score by the source-grammar score.

2.4 Translation Quality-driven Scoring in MT

As can be observed from our descriptions in Section 2.2 and 2.3, both the state-of-the-art PB-SMT approach and the DOT model disambiguate between alternative translations by using scores which are obtained from different sources, such as the log-linear features of PB-SMT or the fragment-pair probabilities in DOT. Although the ultimate goal when determining the best translation from a set of alternative candidates is to select the one that will maximize the perceived translation quality of the system, these scores are computed using methods that maximize the likelihood of the training data. While maximizing the likelihood of the training corpus can be usually done with efficient methods, such as the use of relative-frequency probabilities, this objective does not *directly* takes into account the expected translation quality of the system.

There has been a range of research on the subject of translation quality-based scoring in MT. As mentioned in Section 2.2, MERT (Och, 2003) is the standard way to assign the weights λ_i in equation (2.1). While this scales the components of the model in an optimum way so as to maximise translation quality on a held-out corpus, the features themselves rely heavily on relative frequencies of phrase pairs on the training corpus, a statistic that might improve the likelihood of the training corpus, but that does not necessarily maximise *translation* quality. A well-known limitation of MERT is its difficulty to scale to a larger amount of features than the ones present in a standard PB-SMT system (Och et al., 2003). There has been research carried out on methods that seek to overcome this problem, e.g. (Watanabe et al., 2007) and (Chiang et al., 2008) improve on MERT by using the Margin Infused Relaxed Algorithm (MIRA) (Crammer et al., 2006) to estimate a large amount of syntactic and distortion features from a small held-out corpus, which led to significant translation quality gains in the Hierarchical Phrase-Based SMT framework (Chiang, 2005).

In another line of research, (Liang et al., 2006), (Tillmann and Zhang, 2006) and (Arun and Koehn, 2007) use perceptron-like algorithms to introduce a large number of binary features globally trained to increase Bleu score. These approaches have the drawback that training procedures —including in some cases decoding— have to be redeveloped. In addition, these approaches do not report significant improvements over state-of-the-art PB-SMT systems trained with standard features.

2.5 Evaluation Metrics

To assess the quality of translations output by our systems, and the difference in performance between the various methods we investigate, a metric must be defined.

Ideally one or more evaluators would examine input sentences and output translations and assess how *fluent* the translations are, i.e. how grammatically correct they are and how naturally they read in the target language, and how *adequate* they are, i.e. how well the meaning in the input sentence is preserved in the output translation. Alternatively, given the output from two systems the relative ranking could be determined, i.e. whether one of the system's output is better or whether they are of the same quality. In addition, we would desire that having established the criteria used to evaluate, two different evaluators would rate the same translations in the same way, i.e. we want a high *inter-annotator agreement.* We would also desire the *intra*-annotator agreement to be high, i.e. an evaluator should be consistent in their ratings.

Candidate: the the the the the the the
Reference: The cat is on the mat

Figure 2.9: Example candidate translation and reference. From (Papineni et al., 2002).

Unfortunately, such manual evaluations are both costly and time-consuming. As an alternative we follow standard practice in most MT research, and employ automatic evaluation metrics which have shown high correlation levels with human judgements. However, we do manually inspect sample sentences and provide example output translations, so as to gain insights into the actual effects of our modifications.

The following sections provide a description of the automatic evaluation metrics we employ, as well as the method used to perform statistical significance testing once these scores have been obtained.

2.5.1 Bleu

Bleu (Papineni et al., 2002) is the most widely used metric in MT research. It uses the geometric average of the (modified) precision of n-grams between a candidate translation and the references, together with a brevity penalty to account for recall.

Precision is defined as the percentage of words in the candidate sentence which match a word in a reference translation. Typically, precision is not used alone, but is instead accompanied by a *recall* figure, which amounts to the percentage of words in the reference translation which are present in the candidate sentence.

For example, the (implausible) candidate sentence in Figure 2.9 has a high precision of 7/7. Usually, attempts to game the precision metric in this way are punished with a low recall score, which would amount to 2/6 in this case. However, computing a recall score in the context of MT is troublesome, as it is often desirable to compare translations against multiple references instead of only one. For this reason, Bleu introduces the concept of *modified precision*, which first counts the maximum number of times a word occurs in any single reference translation, and then truncates the total count of each candidate word by its maximum reference count (Papineni et al., 2002). The modified precision of the candidate in Figure 2.9 is, therefore, just 2/7. This concept of modified precision is then generalized to cover the case of n-grams. As described so far, we have only taken into account unigram precision. We can compute analogous scores for bigrams, i.e. sequences of two contiguous words in the candidate sentence which match sequences of two contiguous words in the references, and similar for any n-gram size. When computing the modified precision, statistics are gathered not on a sentence-level basis, but over the entire test corpus. Truncated n-gram matches are obtained and added for each sentence, and divided by the number of candidate n-grams in the test corpus.

Since, as we mentioned, computing recall is troublesome, BLEU introduces a *brevity penalty* which penalizes sentences which are shorter than the best-matching reference. Like modified precision, this penalty is computed over the entire test corpus by first computing r, the sum of the lengths of the references that are closest to each candidate translation, and then c, the sum of the lengths of all candidate sentences in the test corpus.

Let p_n be the precision of n-grams between the candidates in the test corpus and the reference set. Then the BLEU score can be computed as in (2.8):

$$\textsc{Bleu}_N = \mathsf{BP}\cdot \exp(\frac{1}{N}\sum_{n=1}^{N}\log p_n) \tag{2.8}$$

where the brevity penalty BP is defined as in (2.9):

$$\mathsf{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{1-r/c} & \text{if } c \leq r \end{cases} \tag{2.9}$$

Typically, up to 4-grams are considered, i.e. $\textsc{Bleu}_4$ is used.

2.5.2 NIST

Doddington (2002) noted that some aspects of Bleu could be improved, and decided to formulate a modified version of Bleu. The changes introduced include:

- use of the arithmetic average of n-gram counts instead of the geometric average, in order to minimize harsh behaviour when there are low counts for larger values of N,
- introduction of "information" weights which favour those n-grams which occur less frequently,
- modification of the brevity penalty to minimize the effect of small variations in length.

The modified formula is given in (2.10), where β is chosen so as to make the brevity penalty factor equal to 0.5 when the translation length is $2/3^{\text{rds}}$ of the average reference length, $\bar{L}_{ref}$ is the average over all reference translations of the number of words in a reference translation, L_{sys} is the number of words in the translation being scored, and $\mathsf{Info}(w_1 \ldots w_n)$ is computed in the reference document as in (2.11):

$$\mathsf{NIST} = \sum_{n=1}^{N} \left\{ \frac{\sum\limits_{\substack{\text{all } w_1 \ldots w_n \\ \text{that co-occur}}} \mathsf{Info}(w_1 \ldots w_n)}{\sum\limits_{\text{all } w_1 \ldots w_n \text{ in sys output}} 1} \right\} \cdot \exp\left\{ \beta \log^2 \left[\min\left(\frac{L_{sys}}{\bar{L}_{ref}}, 1 \right) \right] \right\} \tag{2.10}$$

$$\mathsf{Info}(w_1 \ldots w_n) = \log_2 \left(\frac{\text{number of occurrences of } w_1 \ldots w_{n-1}}{\text{number of occurrences of } w_1 \ldots w_n} \right) \tag{2.11}$$

2.5.3 Meteor

In contrast with NIST, Meteor (Banerjee and Lavie, 2005; Lavie and Agarwal, 2007; Lavie and Denkowski, 2009) is not a modification of the original Bleu formula, but instead attempts to improve the potential weaknesses of BLEU by a complete new formulation.

Meteor computes a score by first obtaining an *alignment* between the candidate sentence and a reference translation. This alignment, which maps individual words in any of the strings to at most one word in the other string, is incrementally obtained by a series of stages. In the first stage, exact matching of words is used to determine mappings. The second stage uses the Porter stemmer (Porter, 1980) to map two unigrams if they are the same *after* they are stemmed. Finally, the third stage maps two unigrams together if at least one sense of each word belongs to the same synset in WordNet (Miller and Fellbaum, 2007).

Once an alignment has been obtained, unigram precision and recall are computed, and the final score is computed as in (2.12):

$$\mathsf{Meteor} = (1 - \mathsf{Pen}) \cdot F_{\mathsf{mean}} \tag{2.12}$$

where F_{mean} is computed as in (2.13), and Pen as in (2.14):

$$F_{\mathsf{mean}} = \frac{\mathsf{Precision} \cdot \mathsf{Recall}}{\alpha \cdot \mathsf{Precision} + (1 - \alpha) \cdot \mathsf{Recall}} \tag{2.13}$$

$$\mathsf{Pen} = \gamma \cdot \mathsf{frag}^{\beta} \tag{2.14}$$

α, β and γ are parameter constants, and frag is the number of contiguous matching "chunks" divided by the number of matching words.

2.5.4 F-Measure

The last metric we consider uses the notions of precision and recall and combines them into the F-Measure, to allow for a more intuitive interpretation of the scores than BLEU or NIST. This metric is implemented by the General Text Matcher (GTM) (Turian et al., 2003).

Turian et al. (2003) define precision and recall as in (2.15) and (2.16), respectively:

$$\mathsf{precision}(C|R) = \frac{\mathsf{MMS}(C, R)}{|C|} \tag{2.15}$$

$$\mathsf{recall}(C|R) = \frac{\mathsf{MMS}(C, R)}{|R|} \tag{2.16}$$

Here C is the candidate sentence, R the corresponding reference, and MMS stands for the *maximum match size*, a concept from graph theory which is computed by considering the maximum amount of candidate words that match a word in the reference, avoiding the double-count of words which match more than one word.

Having defined precision and recall, the final metric is obtained by the use of the standard f-measure, as in (2.17):

$$\mathsf{f\text{-}measure}(C|R) = \frac{2 \cdot \mathsf{precision}(C|R) \cdot \mathsf{recall}(C|R)}{\mathsf{precision}(C|R) + \mathsf{recall}(C|R)} \tag{2.17}$$

2.5.5 Statistical Significance

When performing experiments using two different systems (or when introducing modifications to a system) and evaluating on a test corpus, we wish to be able to determine with some confidence whether the difference shown by the automatic evaluation metrics is due to a significant improvement in a system's behaviour, or whether the difference in the scores is simply due to random variations in our particular test corpus.

Standard statistical methods to perform statistical significance testing are troublesome to implement in an MT context, given that the ability to assess the quality of an individual sentence would be required, while metrics such as Bleu or NIST were designed to be used at document level. To solve this we use a widely used method known as *paired bootstrap resampling* (Koehn, 2004).

Paired bootstrap resampling makes the assumption that estimating a confidence interval from a large number of test sets, each consisting of n sentences drawn with replacement from an original set of n test sentences, is as good as estimating the interval with the sentences being drawn from an infinite source of test sentences. Under this assumption, which is made in order to avoid having to evaluate our systems on an implausibly large amounts of sentences, the method is as follows: we translate a set of sentences, consisting of (say) 300 sentences, using the two systems that we wish to compare. We then proceed to create a large number (e.g. 1000) of new test sets by taking sentences from the original test set, with replacement. For each of the newly created test sets, we compute the automatic evaluation scores using the output from both systems, and determine which system has a higher score. If one system is better on at least an $x\%$ of cases (e.g. 950 samples if creating 1000 new test sets), then it is deemed to be statistical significantly better at the $p \leq 1 - x$ p-level, e.g. 0.05.

In our experiments we perform statistical significance testing by using paired bootstrap resampling to create 10,000 new test sets.

2.6 Summary

In this chapter we gave an overview of different approaches to machine translation. In particular we provided an overview of PB-SMT which, while being an arguably simple model which does not exploit any structural or linguistic information, still achieves state-of-the-art performance. We also provided an overview of different approaches which attempt to define models in terms of richer structures. In particular, we gave a detailed description of DOT, a tree-to-tree system on which our efforts to overcome the limitations we perceive in the state-of-the-art approach are focused. We finished the chapter with an overview of work carried out to bring translation-quality awareness into MT scoring, and with an introduction to the evaluation metrics which we use to determine the differences in translation quality between our systems.

In the next chapter, we address our research questions **RQ1** and **RQ2** by formulating an algorithm which can relate the translation units used by an MT system to their expected impact on translation quality. We experiment with adapting this algorithm for PB-SMT and for DOT, reporting significant gains in translation quality for a Spanish-to-English translation task.

Chapter 3

Translation Quality-Oriented Parameter Estimation

In this chapter we present a scoring algorithm which allows an MT system to take decisions based on the notion of the expected impact on translation quality.

We motivate our approach in Section 3.1, and then introduce our scoring algorithm in Section 3.2. This algorithm takes an arbitrary MT system satisfying certain properties, and uses it to obtain the list of candidate translations for a corpus. From this list, the sentences which maximize translation quality (the *oracles*) are determined, and a score is obtained for each translation unit according to how similar they are to units used in the oracle translations. We provide two instantiations of this scoring method, one for the DOT translation system (in Section 3.3) and one for PB-SMT in Section 3.4. We empirically evaluate the performance of this method on Spanish-to-English translation, and provide insights into the benefits of this approach.

3.1 Motivation

In Chapter 2, we presented an overview of PB-SMT, the state-of-the-art approach to Machine Translation, and of DOT, a syntax-based tree-to-tree system. We can consider the workflow used to build and test a PB-SMT system as consisting of three phases:

1. Phrase extraction and feature estimation.
2. Tuning of feature weights.
3. System testing and evaluation.

The ultimate goal is to obtain a system that performs as well as it can in the third phase, where system performance is determined by a translation quality metric such as Bleu (Papineni et al., 2002). For this reason, in the second phase the weight that each feature receives is set via MERT (Och, 2003) in such a way that the score assigned by such a metric when translating a development corpus is maximized. However, in the first phase there is no direct relation between the way in which features are estimated and the score assigned by translation quality metrics. In particular, there is no notion of the expected impact that using a particular phrase pair has on translation quality. The main features in PB-SMT are the phrase translation probabilities which, in an attempt to maximize the likelihood of the training corpus,[1] are estimated by counting the frequency of occurrence of a target phrase as a translation of a source phrase, relative to the alternative translations for that source phrase. The remaining features either aim at smoothing the phrase translation probabilities (as is the case with the lexical weighting features), or at controlling the length of the output sentence (phrase and word penalty features). Maximizing the likelihood of the training corpus is a reasonable objective which can lead to good results, as evidenced by the relatively good performance of the MT systems which incorporate these features. However, this objective is only *indirectly*

[1]It should be noted that, since many phrases can overlap and nest in the parallel corpus, leading to many possible derivations for a single sentence pair, the phrase extraction method in PB-SMT does not necessarily maximise the likelihood of the training data, and is instead a heuristic which has been shown empirically to perform well (DeNero et al., 2006). Similarly, given that DOT extracts the complete tree pairs in the parallel treebank as fragment pairs, the trivial maximum likelihood estimator for DOT would assign all of the probability mass to the complete tree pairs in the treebank and none of it to smaller fragments, with the corollary that the model would not be able to generalize over unseen data (Johnson, 2002).

Phase	**Objective to Maximize**
Training/Feature Estimation	Likelihood
Feature Weights Tuning	Translation Quality
Evaluation	Translation Quality

Table 3.1: Phases in PB-SMT workflow, along with the objective that each phase aims at maximizing.

related to translation quality.

This lack of a translation quality-oriented feature in the model presents an inconsistency with the goal used to evaluate the system, which is clearly shown when we list the criterion that is used in each phase to determine the quality of the models, as in Table 3.1.

A similar situation arises in DOT, where derivations are scored according to the relative frequency of the fragment pairs they are composed of. In this case the situation is worse, as this is the *only* source of information considered in the scoring (cf. Chapter 4, where we address this issue). This is also the case with a wide range of MT systems, e.g. (Yamada and Knight, 2001; Galley et al., 2004, 2006), where probability distributions are estimated via maximum likelihood over each decision type, or (Chiang, 2005, 2007), which inherits the feature set used in standard PB-SMT (Koehn et al., 2003).

Given such a wide range of systems lacking in this regard, it would be desirable to come up with a translation quality-oriented estimation method that would be general enough to be adopted by a variety of MT systems. We propose such a method in the next section. We instantiate this approach in Section 3.3 by obtaining an accuracy-based DOT system, and we demonstrate the flexibility of the formulation presented in the next section by adapting it to a PB-SMT setting in Section 3.4.

3.2 Accuracy-Based Scoring

Previous approaches aimed at incorporating translation-quality awareness into MT scoring (Liang et al., 2006; Tillmann and Zhang, 2006; Watanabe et al., 2007; Blunsom et al., 2008) attempt to globally optimize translation quality by estimating the weights of *millions* of features, thus replacing the original feature set. However, relative frequency-based features have the property of being efficient to estimate: only the counting of occurrences of events is required. In addition, in some cases these features have been shown to outperform purely translation quality-oriented approaches (Arun and Koehn, 2007). This is perhaps a consequence of the complexity of these approaches: their use requires the redevelopment of the training procedures, and the decoders need to be reengineered so that this kind of features can be used, which leads to prototypes being used as opposed to fully fledged systems. For example, (Liang et al., 2006) provide results using a system which only implements a limited distortion model, which underperforms compared to a standard PB-SMT system.

For these reasons, in our bid to incorporate translation quality-oriented features into MT, we do not aim at replacing current scoring models; rather, we aim to complement them by incorporating (a few) new features that provide a notion of the expected translation quality of the decisions the system takes.

3.2.1 Preliminaries

We assume that an MT system contains a set of *translation units* which it uses to build a translation given an input sentence, e.g. phrase pairs in PB-SMT and fragment pairs in DOT. From a scoring point of view, these translation units are atomic, meaning that a score can be assigned to them, but they cannot be further decomposed into other smaller units which can be scored.

We assume that translation quality is measured by a function $\mathbf{E}(\mathbf{s}, \tilde{\mathbf{t}}, \vec{\mathbf{t}})$ which assigns a score between 0 and 1 to a translation $\tilde{\mathbf{t}}$, measuring how “good” it is as a translation of the sentence $\mathbf{s}$, taking the sentences in the vector $\vec{\mathbf{t}}$ as references. Examples of these functions are the translation quality metrics defined in Section 2.5, namely Bleu, NIST, the F-Measure and Meteor.

Given an input sentence $\mathbf{s}$, let $T_{\mathbf{s}}$ be the set of target translations that the system is able to produce by composing its translation units. In the case of PB-

$$\begin{tikzcd} \text{He} & \textbf{runs} \arrow[d] & \text{the company} \\ \text{El} & \textbf{dirige} & \text{la empresa} \end{tikzcd}$$

(a)

$$\begin{tikzcd} \text{He} & \textbf{runs} \arrow[d] & \text{the marathon} \\ \text{El} & \textbf{corre} & \text{la maratón} \end{tikzcd}$$

(b)

Figure 3.1: Different translations for the word “runs”, depending on different source contexts.

SMT, $T_{\mathbf{s}}$ contains all combinations of the target sides of phrase pairs whose source sides appear in $\mathbf{s}$. This set is, therefore, really large, and as a result of the pruning techniques employed by the decoders, in practice MT systems do not consider all of the candidate translations contained therein. However, theoretically our ultimate goal is to score phrases in such a way that the output of the system is the sentence $\hat{\mathbf{t}}$ that maximizes $\mathbf{E}$ over $T_{\mathbf{s}}$, as in (3.1):

$$\hat{\mathbf{t}} = \operatorname*{argmax}_{\tilde{\mathbf{t}} \in T_{\mathbf{s}}} \mathbf{E}(\mathbf{s}, \tilde{\mathbf{t}}, \vec{\mathbf{t}}) \tag{3.1}$$

Accomplishing this is extremely difficult, since the quality $\mathbf{E}$ of a translation depends not only on the quality of each individual translation unit involved in it, but on the entire series of translation units that constitute the translation. This means that while using a translation unit can be completely adequate when translating some source words in one sentence, the same translation unit translating the same source segment can be completely inadequate when taking a different source (or target) context into account. For example, when translating the English word “runs” into Spanish, if we consider the context of a sentence such as the one in Figure 3.1(a), we would obtain a different translation than if considering the source context in Figure 3.1(b). Since in most formulations (with the exception of global features such as language model scores), translation units are scored independently of each other, the approach that we take is to attempt to differentiate between units that *on average* lead to good translations from units that typically do not. We encode this information as a function $\mathsf{Acc}(f, e)$ that quantifies how similar a translation unit

$\langle f, e \rangle$ is on average to a translation unit involved in a candidate translation that was shown to maximize the translation quality metric $\mathbf{E}$.

3.2.2 Estimation Corpus

We estimate the function Acc by determining how similar (or dissimilar) a translation unit is compared to translation units which were deemed to be of high quality when translating a particular input sentence. These input sentences are taken from a special corpus which we call the *estimation corpus*. The estimation corpus behaves as a large test set. Sentences in this corpus will be translated, and evidence (both positive and negative) for the quality of translation units used during translation will be collected. Note that the amount of preprocessing required for these sentences is only that which would also apply to test sentences, e.g. for the case of DOT there is no need for these sentences to be parsed and sub-sententially aligned.

In order to be able to observe translation units in as many contexts as possible, and to be able to score as many units as possible, this corpus should be large (i.e. in the order of the sizes used for translation unit extraction). To alleviate the need for large amounts of additional parallel data, parallel corpora used to extract translation units (i.e. training data) can be included in the estimation corpus.[2] However, a significant amount of unseen data (i.e. held-out data) should also be included in this corpus. In our experiments, the percentage of held-out data in estimation corpora is at least 50%.

3.2.3 Source Spans

We assume that an MT system can generate a list (which we denote T_N) of the N best-scoring translation candidates for an input sentence. Since it is infeasible

[2]Translating a sentence in the training data will likely result in the translation observed during training to be output as the most probable translation, and in few (but large) translation units to be used. However, our use of the training data as estimation corpus is possible because for every input sentence we will consider all possible translations explored by the system (and not just the most probable one), which will cause many different translation units to be observed.

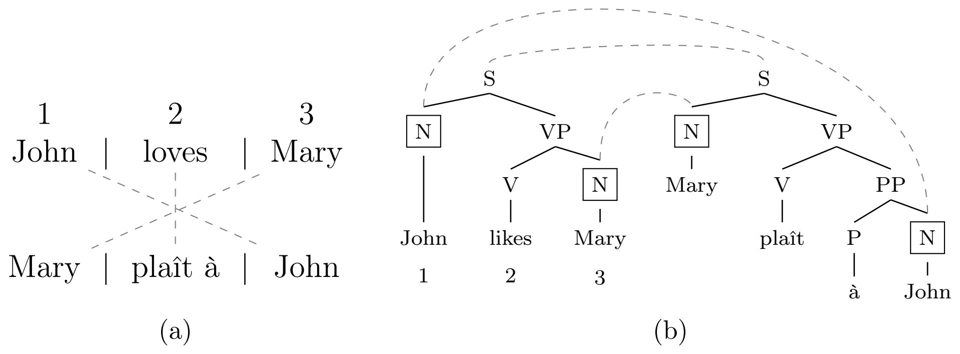
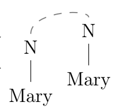
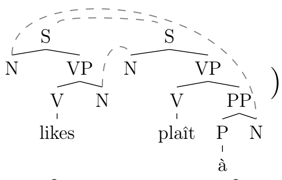


Figure 3.2: Determining the translation units used to translate a sentence, and the source span of each unit.

to explore the entire set $T_{\mathbf{s}}$, we approximate this by only considering the target-language sentences present in this N-best list. Furthermore, we assume that we are able to determine the translation units used to build each of the translations in the N-best list, a feature most decoders incorporate. Finally, we assume that we can deduce the source-sentence span associated to each translation unit involved in the generation of a translation. For example, in PB-SMT we can obtain the segmentation used and the corresponding source phrase for each target phrase, as in Figure 3.2(a). From this information we can obtain the list of phrase pairs used, and we can determine the source sentence span associated with each phrase pair by considering the position in the source sentence of the source side of each phrase pair. For the case of DOT, Figure 3.2(b) gives the derivation used by indicating substitution sites using boxed nodes. From this we can determine the fragment pairs used, and we can deduce the source span for each fragment by considering the span covered by the root of each source-side fragment. We denote the source-sentence span associated to each translation unit as $\mathsf{source\_span}(u) = (l, m)$, meaning that the translation unit u has a source span starting at the source-sentence position l and ending at source position m (e.g. in Figure 3.2, $\mathsf{source\_span}($ 

$) = (3, 3)$).
Note that for the case of DOT, the source span of a fragment pair in a derivation is determined by the *root* of the source side of the fragment pair, regardless of whether

the fragment pair contains substitution sites or not. For example, in Figure 3.2(b)

source_span(

) $= (1, 3)$).

Using this information, a function Ω indicating the mapping between a source sentence span (l, m) and the associated set of translation units in the oracle translations $\mathcal{O}$ can be defined, as in (3.2):

$$\Omega_{\mathcal{O}}(l, m) = \{\tilde{e}_o | \exists\, \mathbf{t} \in \mathcal{O} : \tilde{e}_o \in \mathbf{t} \wedge \mathsf{source\_span}(\tilde{e}_o) = (l, m)\} \tag{3.2}$$

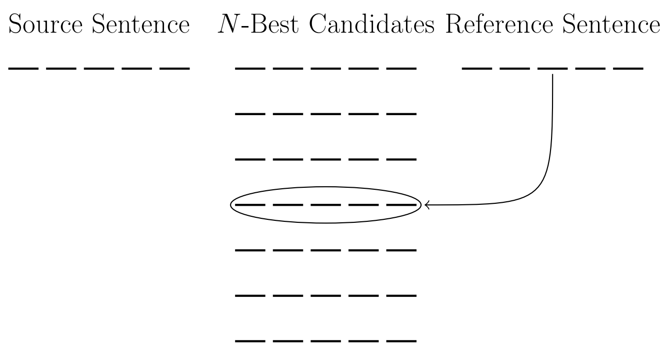
3.2.4 The Algorithm

Our accuracy-based scoring procedure takes an MT system which satisfies our assumptions and associates its translation units with a new score. We refer to this original MT system as the *baseline* system.

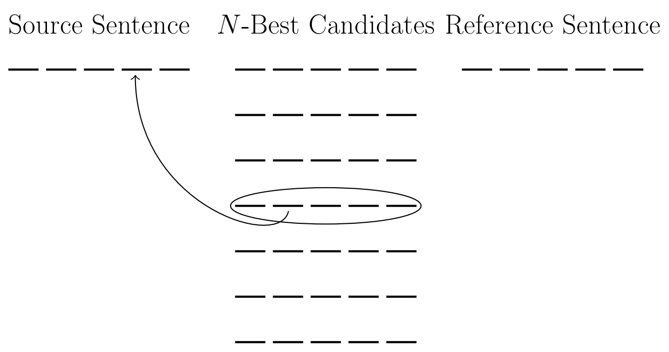
The estimation procedure is as follows. For each source-language sentence $\mathbf{s}$ in the estimation corpus, we obtain an N-best list of translation hypotheses T_N using the baseline system, and we use the metric $\mathbf{E}$ to determine the translation in T_N that maximises translation quality. Noting that many target translations may receive the same highest score under $\mathbf{E}$, we define the set $\mathcal{O}$ as in (3.3):

$$\mathcal{O} = \operatorname*{argmax}_{\tilde{\mathbf{t}} \in T_N} \mathbf{E}(\mathbf{s}, \tilde{\mathbf{t}}, \vec{\mathbf{t}}\,) \tag{3.3}$$

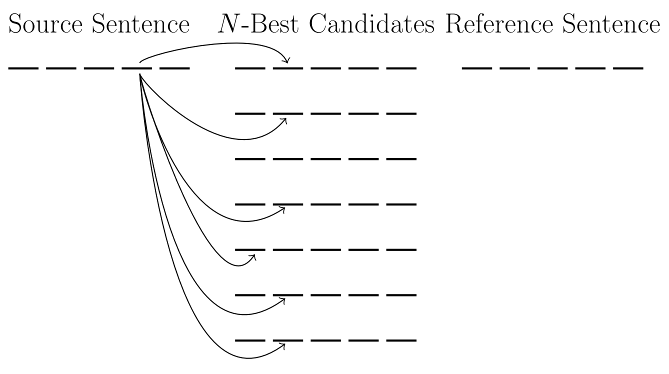
We refer to the candidate translations in $\mathcal{O}$ as *oracle* translations. Obtaining oracles from an N-best list (as illustrated in Figure 3.3 (a)) is referred to as *local updating* by Liang et al. (2006) (as opposed to what they call *bold updating*, where the decoder is forced to produce the reference translation). In the context of perceptron-based training for MT, Liang et al. (2006) find that local updating significantly outperforms bold updating (possibly due to the forcing of the alignments between the source sentence and the reference, which might produce unreasonable alignments).



(a) Using a translation quality metric, an oracle translation is chosen from the N-best list.



(b) For each unit used in the oracle, its corresponding source sentence span is determined.



(c) All translation units in the N-best list associated with the source span in (b) are determined. These translation units will later be compared with the oracle unit associated with this source-sentence span.

Figure 3.3: Sketch of our proposed accuracy-based scoring algorithm

However, in the future it would be interesting to measure the effects of directly using the reference sentence as an oracle. This would require obtaining an alignment between the reference and the words in the input sentence, which could be performed using word-alignment methods, e.g. (Brown et al., 1993), or methods similar to forced alignments in speech recognition, e.g. (Bahl et al., 1983). A further possibility for improving oracle selection is to search for the sequence of target units in the N-best list which produce a target sentence which maximizes translation quality, or alternatively the sequence of target words in the N-best list which produce the best target sentence. As in the case of directly using the reference as an oracle, this would require an alignment phase which finds the correspondence between this best sentence and the source segmentation needed to generate it.

Once oracles have been obtained, we consider the translation units which were used to build them (the *oracle units*). As the use of these units led to the highest translation quality according to some automatic evaluation metric, we assume them to be of high quality, and assign them a high score. To allow for a degree of flexibility, we compare the remaining units in the N-best list with the oracle units, which allows a larger amount of units to receive a score rather than only oracle units. For this comparison to be meaningful, we only compare candidate translation units with oracle translation units that translate the same source span, as illustrated in Figure 3.3(b). Then, for each translation unit u in the oracle sentences, we consider all translation units in the N-best list which are associated with the same source-sentence span (l, m) (as indicated by the function $\mathsf{source\_span}(u)$ (cf. equation (3.2)) and as illustrated in Figure 3.3(c)), and we compare them with the oracle units associated with this same source-sentence span (l, m) (i.e. the units in $\Omega_{\mathcal{O}}(l, m)$). This comparison between translation units is performed by a unit-similarity metric δ, which assigns a score to a translation unit representing the similarity between the candidate unit and the oracle unit. There might be more than one oracle unit associated with the same source span in the case where more than one candidate sentence receives the same best score under $\mathbf{E}$ when selecting oracles. When this happens,

Algorithm 1 Accuracy-Based Scoring

```
Input: Estimation corpus: $\{(s_1, ref_1) \ldots (s_H, ref_H)\}$
 1: count ← assign 0 to all units
 2: score ← assign 0 to all units
 3: for $i = 1 \ldots H$ do
 4:    $T_N \leftarrow \mathsf{decode}(s_i)$
 5:    $\mathcal{O} \leftarrow \underset{c \in T_N}{\text{argmax}}\, \mathbf{E}(s_i, c, ref_i)$
 6:    for all $c \in T_N$ do
 7:       for all $u \in c$ do
 8:          $\mathcal{U} \leftarrow \Omega_{\mathcal{O}}(\mathsf{source\_span}(u))$
 9:          if $\mathcal{U} \neq \emptyset$ then
10:             $\mathsf{score}(u) \leftarrow \mathsf{score}(u) + \underset{u_o \in \mathcal{U}}{\text{argmin}}\, \delta(u, u_o)$
11:             $\mathsf{count}(u) \leftarrow \mathsf{count}(u) + 1$
12:          end if
13:       end for
14:    end for
15: end for
16: for all $u$ such that $\mathsf{count}(u) \neq 0$ do
17:    $\mathsf{ABS(u)} \leftarrow \frac{\mathsf{score}(u)}{\mathsf{count}(u)}$
18: end for
19: return ABS
```

we choose to compare a candidate unit with all of the oracle units associated with its same source span, and keep the score obtained from the unit it is most similar to.

By repeating this process over all sentences in the estimation corpus, and averaging the similarity scores obtained by each unit, we obtain a notion of how different on average a particular unit is to oracle units. We can then exploit this score by augmenting our baseline system so that these scores are taken into account when choosing between alternative translations. The scoring method we have just outlined is summarized in Algorithm 1.

Note that a translation unit needs to appear in an N-best list translating the same source span as an oracle unit in order for it to receive an accuracy-based score using Algorithm 1. As discarding units which do not receive a score could lead to a system with a reduced coverage, or might be technically challenging (e.g. the Goodman reduction for DOT implicitly encodes all possible fragments), the question of how to deal with unscored units arises. A possible solution is to assign unscored units a default score equal to the median of all of the scores observed. This is one of the solutions that we implement here, although other methods could be used depending on the particular MT system we are using. In particular in the future more sophisticated methods for assigning this default weight could be developed, for example the average score obtained by units of the same size (or some other property such as labels used in fragments) can be found, and units can receive a default score according to these properties.

Note also that in Algorithm 1, except possibly for line 5,[3] only one sentence in the estimation corpus is taken into account at a time, with the other sentences in the corpus playing no role in the estimation. This means that in the case where **E** is a sentence-level metric, the estimation of our accuracy-based feature can be performed in parallel at least up to the sentence level, enabling significant increases in estimation speed to take place.

As our scoring method takes a baseline system and produces a new score distribution for its translation units which might replace or modify the original scoring model, in the following we refer to this process as *rescoring*.

The particular unit-similarity metric δ we use depends on the nature of the translation units. Details of these metrics and of the precise way in which these accuracy-based scores can be incorporated into a particular MT framework are given in the following sections.

3.3 Accuracy-Based Scoring for DOT

As explained in Section 2.3, DOT scores derivations using exclusively the relative frequencies of the constituent fragments involved therein. Our goal is to bring translation-quality awareness into the DOT model. However, unlike in PB-SMT,

[3]Some sentence-level approximations of BLEU might lead to dependencies between sentences in the estimation corpus. For example in (Watanabe et al., 2006, 2007) the BLEU score for a particular sentence depends on the entire estimation corpus.

DOT is not currently able to exploit features other than the relative-frequency of fragment pairs. We address this issue in Chapter 4, where we define a log-linear version of DOT which is able to exploit not only additional features of the fragment pairs, but also features which include information not available at training time, such as the particular input sentence we are translating. This new model will allow features of the kind present in PB-SMT to be incorporated into DOT, allowing for the gap between the two systems to be bridged.

For the moment, for the purpose of demonstrating the adaptability of the scoring method outlined in the previous section to the case of DOT, we reformulate DOT scoring as log-linear but at the fragment level, making it possible to directly incorporate our new scores into the grammar. For all tree fragment pairs $\langle d_f, d_e \rangle$, let $l(\langle d_f, d_e \rangle)$ be the logarithm of the probability of the fragment pair (equation (2.4) in Section 2.3), as in (3.4):

$$l(\langle d_f, d_e \rangle) = \log(P(\langle d_f, d_e \rangle)) \tag{3.4}$$

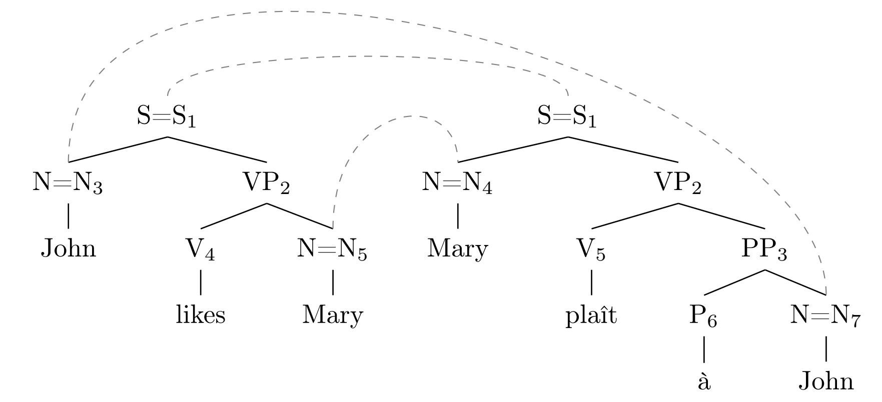
The general form of a fragment pair will now be as in (3.5):

$$S(\langle d_f, d_e \rangle) = \alpha_0 l(\langle d_f, d_e \rangle) + \alpha_1 \mathsf{Acc}(\langle d_f, d_e \rangle) \tag{3.5}$$

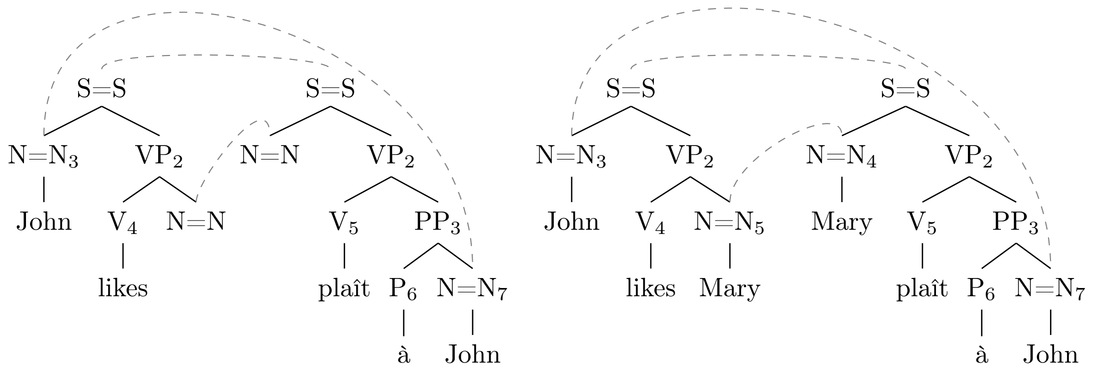
That is, the score S of a fragment will be a weighted linear combination of the relative-frequency score and the accuracy-based score. The score of a derivation is now given by (3.6):

$$S(d) = S(\langle d_e, d_f \rangle_1 \circ \ldots \circ \langle d_e, d_f \rangle_N) = \sum_i S(\langle d_e, d_f \rangle_i) \tag{3.6}$$

Although our approach is formulated in terms of DOT fragments, in practice it is infeasible to score only those fragments seen during the scoring process. The Goodman reduction for DOT (Goodman, 1996; Hearne, 2005) allows the efficient representation of fragments and their probabilities by assuming that probabilities



(a) A training tree pair, where the annotations required by the Goodman reduction have been made.



(b) Two of the possibly extracted fragment pairs, as they would be represented by the indexed Goodman PCFG

Source PCFG	
S=S → N=N VP+2	0.5
S=S → N=N+3 VP+2	0.5
S=S+1 → N=N VP+2	0.5
S=S+1 → N=N+3 VP+2	0.5
N=N → John	0.5
N=N+3 → John	1
VP+2 → V+4 N=N	0.5
VP+2 → V+4 N=N+5	0.5
V+4 → likes	1
N=N → Mary	0.5
N=N+5 → Mary	1

(c) Source side of the PCFG generated by the Goodman reduction.

Figure 3.4: A parallel tree and its corresponding Goodman reduction.

are conditioned only on fragment root pairs. This allows it to reduce the amount of grammar rules needed to represent fragments by using the same rules to represent the portion of two fragments which share the same root pair. For example, in Figure 3.4(a), a tree pair is given. Two of the fragment pairs which can be extracted from this tree pair are shown in Figure 3.4(b). These two fragments differ only in the inclusion the word "Mary" in the second one. Although in the DOT formulation these are two completely independent fragments, since they share the same root pairs the Goodman reduction (given in Figure 3.4(c)) allows us to reduce the number of rules needed to represent them, by representing shared portions of the fragments using a single grammar rule. Consider for example the rule in (3.7):

$$\text{S=S} \rightarrow \text{N=N+3} \quad \text{VP+2} \tag{3.7}$$

As both fragments differ only in portions not relevant to this rule, the rule is stored only once in the grammar. The probabilities in the remaining rules accommodate for any differences in probabilities between the two fragments.

If we were to assign a new score to each individual fragment, we could face the situation where the two fragments in Figure 3.4(b) have different scores. In this case, new rules would need to be added to the grammar to differentiate the fragments and to allow for a different score to be assigned to each. Since the amount of fragments extracted from the parallel treebank is exponential in the amount of nodes in the trees, this would lead to a substantial increase in grammar size. Instead, we score the individual PCFG rules resulting from the Goodman reduction which the fragments are composed of. We evenly divide the total amount of scoring mass among the PCFG rules used to represent a particular fragment. After running our scoring method over the whole estimation corpus, an individual PCFG rule would have received several different scores as a result of it being used in different fragments translating different input sentences. We assign to each of these rules the average of the rule score obtained over all fragments in which it appears. That is, if $\Delta(\langle d_f, d_e \rangle)$

is a list of all the accuracy-based scores received by a fragment pair $\langle d_f, d_e \rangle$, and $\mathsf{size}(d_e)$ represents the amount of PCFG rules in the target side d_e of the fragment pair, then the score of a rule r is as in (3.8):

$$S(r) = \frac{\sum\limits_{\langle d_f,d_e\rangle : r \in d_e} \; \sum\limits_{\delta \in \Delta(\langle d_f,d_e\rangle)} \delta / \mathsf{size}(d_e)}{\sum\limits_{\langle d_f,d_e\rangle : r \in d_e} \mathrm{len}(\Delta(\langle d_f, d_e\rangle))} \tag{3.8}$$

As a side-effect, fragments which were unseen during the scoring process will receive a score according to fragments which share common PCFG rules. For example, if during scoring we assign an accuracy-based score to the first fragment in Figure 3.4(b), but the second fragment never appears in an N-best list translating the same source span as an oracle fragment and so a score is not assigned to it, the fact that these fragments share common PCFG rules means that the second fragment is indirectly scored according to the score received by the first. This raises the question of whether this phenomenon is beneficial or not. As the assignment of an individual score to each fragment would require a complete reimplementation of our system, and as we show in our experiments in Section 3.3.6 that this solution is not an impediment for significant improvements to be obtained with our scoring method, in this thesis we do not aim at assigning an individual fragment score.

3.3.1 Oracle Selection Metric

After translating a sentence in the estimation corpus using the baseline system, we obtain a target-language chart which efficiently stores all of the candidate translations, along with information about the bilingual fragments which were used in the derivations that generated these candidate translations. From this chart we select oracles by using the metric **E**. Although BLEU is the most widely used metric in MT research, which would make it the natural choice for **E**, this metric was designed to evaluate the output of an MT system when translating a *document* consisting of a collection of sentences. Evaluation at the sentence level as required in equation

(3.3) is troublesome for BLEU, as it will assign a score equal to 0 to most sentences. We investigate the performance of a sentence-level approximation of BLEU (which we call sBLEU, and which was proposed by (Liang et al., 2006)) in Section 3.4 when performing experiments with PB-SMT. Here we choose to use the F-Measure as our metric **E**. This metric has the property of being efficient to compute, which renders it suitable for large-scale estimation. In addition, it has been found to have good correlation with human judgements at the sentence level (Sun, 2010).

3.3.2 Structured Fragment Rescoring

We now define the similarity measure δ used in line 10 of Algorithm 1 to compare a candidate fragment pair with an oracle. Our comparison will only take into account the *target side* of fragments. For a given target-side tree e of a fragment pair, let $\mathsf{root}(e)$ be the root of the target tree, let $\mathsf{rhs1}(e)$ be the left subtree of $\mathsf{root}(e)$, and let $\mathsf{rhs2}(e)$ be the right subtree, and let $\mathsf{yield}(e)$ be the list of frontiers in e. The difference between a candidate fragment e_c and an oracle fragment e_o is given by the recursive equations in (3.9) and (3.10). The base case, where both e_c and e_o are unary subtrees or substitution sites, is given in (3.9):

$$\delta(e_c, e_o) = \begin{cases} 0 & \text{if } e_c = e_o \\ 1 & \text{if } e_c \neq e_o \end{cases} \tag{3.9}$$

The inductive case where at least one of e_c and e_o are not unary trees is given in (3.10). In the case where one of the fragments is unary, only those terms which are defined are considered, i.e. only those terms which do not attempt to use the rhs1 or rhs2 operators on the unary tree:

$$\delta(e_c, e_o) = \min \begin{cases} \delta(\mathsf{rhs1}(e_c), \mathsf{rhs1}(e_o)) + \delta(\mathsf{rhs2}(e_c), \mathsf{rhs2}(e_o)), \\ \delta(\mathsf{rhs2}(e_c), \mathsf{rhs1}(e_o)) + \delta(\mathsf{rhs1}(e_c), \mathsf{rhs1}(e_o) + 1, \\ \delta(e_c, \mathsf{rhs1}(e_o)) + |\mathsf{yield}(\mathsf{rhs2}(e_o))|, \\ \delta(e_c, \mathsf{rhs2}(e_o)) + |\mathsf{yield}(\mathsf{rhs1}(e_o))|, \\ \delta(\mathsf{rhs1}(e_c), e_o) + |\mathsf{yield}(\mathsf{rhs2}(e_c))|, \\ \delta(\mathsf{rhs2}(e_c), e_o) + |\mathsf{yield}(\mathsf{rhs1}(e_c)| \end{cases} \tag{3.10}$$

These equations define a minimum edit distance between two fragment trees, allowing *sub-fragment* order inversion (second argument of the min function in (3.10)), deletion (arguments 3 and 4) and insertion (arguments 5 and 6) as edit operations. This contrasts with other tree edit-distances in the literature, e.g. (Tai, 1979; Zhang and Shasha, 1989; Emms, 2006), in which edit operations are defined over *nodes* (as opposed to our use of sub-fragments). This means that our metric is able to capture the swapping between the children of a node with a single operation, as is the case with DOT translation rules. If we consider the target fragments in Figure 3.5:

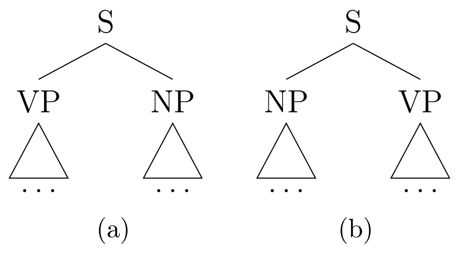


Figure 3.5: Target sides of two fragment pairs

our metric can assign an edit operation of 1 to the reordering needed to transform tree (a) into (b). In comparison, the metric in (Tai, 1979) would need insertion and substitution operators to be applied to each node in each of the subtrees, leading to the swapping operation being dependant on sub-tree size.

Figure 3.6 further illustrates our edit distance metric. The only difference be-

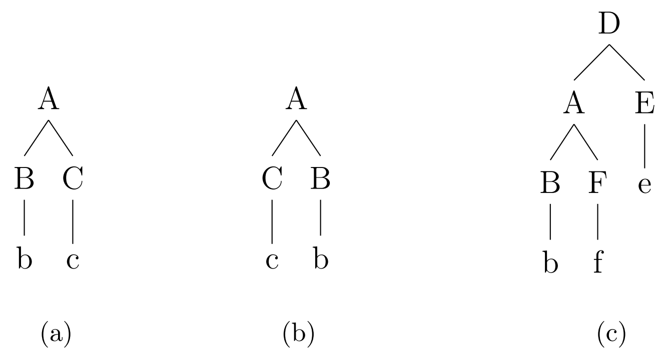


Figure 3.6: Comparing trees (a) and (b) with our distance metric yields a value of 1. The difference between trees (a) and (c) is 2, and for trees (b) and (c) the distance is 3.

tween trees (a) and (b) in this figure is that their children have been inverted. To compare these trees using our distance metric, we first compute the first argument of the min function in equation (3.10), directly comparing the structure of each immediate subtree. We then compute the second argument, obtaining the cost of performing an inversion, and finally compute the remaining arguments, assessing the cost of allowing each tree to be a direct subtree of the other. The result of this computation is 1, representing the inversion operation required to transform tree (a) into tree (b). If we compare trees (a) and (c) in Figure 3.6, we obtain a value of 2, given that the minimum operations required to transform tree (a) into tree (c) are inserting an additional subtree at the top level and then substituting the subtree rooted by C for the subtree rooted by F. If we compare tree (b) with tree (c) then the distance is 3, since we are now required to also replace the subtree rooted by C by the one rooted by B. To efficiently compute these differences, we implement the recursion using dynamic programming, which allows us to evaluate each sub-structure only once.

We directly use our tree edit distance of equation (3.10) as accuracy-based scores for fragments. To define the function Acc of equation (3.5), we compare each fragment against the set of oracle fragments associated with the same source span and select the lowest edit cost, assigning to the candidate the negative difference between it and the oracle fragment it is most similar to, as in (3.11):

$$\mathsf{Acc}_{\mathsf{SFR}}(\langle d_f, d_e\rangle) = \max_{\langle d_f^o, d_e^o\rangle \in \mathbf{D}^o : d_e^o \in \Omega_{\mathcal{O}}(\mathsf{source\_span}(\langle d_f, d_e\rangle))} -\delta(d_e, d_e^o) \tag{3.11}$$

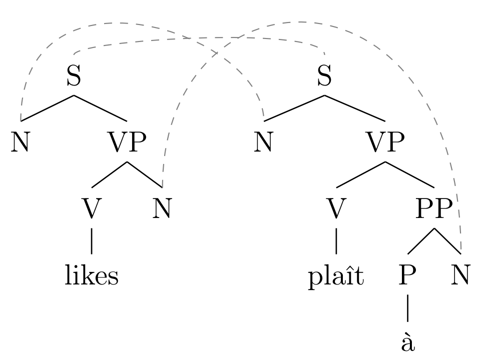
As previously explained, given the Goodman reduction for DOT, in practice we divide the fragment score by the number of rules in the fragment, and assign to each rule the average rule score obtained across all rescored fragments in which it appears.

We refer to this as *Structured Fragment Rescoring* (SFR) as, in contrast with the metric defined in Section 3.3.4, the internal structure of the fragment pairs is taken into account to compute edit distances.

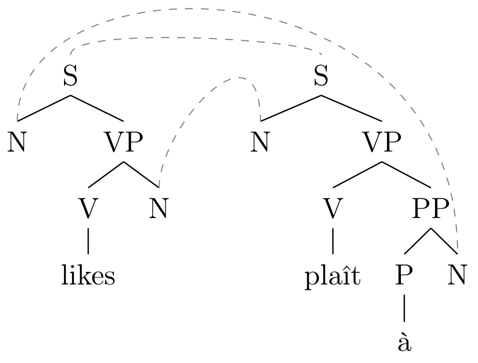
3.3.3 Normalised Structured Fragment Rescoring

When comparing fragment pairs using the edit distance δ, on average we would expect a larger amount of edit operations to be needed to transform trees composed of a large amount of rules, compared to the case of computing the difference between "small" fragments. When assigning the edit distance as a direct score as in the $\mathsf{Acc}_{\mathsf{SFR}}$ feature of equation (3.11), this might lead to an unwanted bias against larger fragments, as on average those will have a larger edit distance score. To avoid this, we experiment with a feature that normalises the absolute edit distance by the maximum possible edit distance, i.e. the amount of frontier nodes in the larger fragment. The normalised score is given in (3.12):

$$\mathsf{Acc}_{\mathsf{NSFR}}(\langle d_f, d_e\rangle) = \max_{\langle d_f^o, d_e^o\rangle \in \mathbf{D}^o : d_e^o \in \Omega_{\mathcal{O}}(\mathsf{source\_span}(\langle d_f, d_e\rangle))} \log(1 - \frac{\delta(d_e, d_e^o)}{\max \begin{cases} |\mathsf{yield}(d_e)|, \\ |\mathsf{yield}(d_e^o)| \end{cases}}) \tag{3.12}$$



(a) Candidate fragment pair



(b) Oracle fragment pair

Figure 3.7: Candidate fragment pair which differs with the oracle only in its alignments.

3.3.4 Fragment Surface Rescoring

The above similarity metrics compare the target side of two fragments by computing the tree edit distance between them. If the candidate and oracle fragments have the same (or similar) list of frontiers, but differ in their internal structure, this metric would penalize the candidate fragment by assigning it a low score. However, our ultimate goal is to produce sentences which are as close as possible to the oracle translations. It would, therefore, be appealing to consider a metric which assigns a high score to a candidate fragment if it has lexical nodes and substitution sites similar to the oracle, regardless of its internal structure.

We experiment therefore with a metric which computes the standard Damerau-Levenshtein string edit distance $\delta_{dl}(d_e, d_e^o)$ (Damerau, 1964) between the yields of the target fragment pairs. The case of fragments having more than one substitution site with the same label represents a problem because we wish to penalize fragments whose links differ from those in the oracle fragments. For example, the fragment pairs in Figure 3.7 differ only in their alignments, and share the same frontier yield. However, using the candidate fragment will result in an incorrect ordering of the target sentence. We account for this situation by representing substitution sites by the source span which they cover in the derivation that generated the oracle sentence. We refer to this metric as *fragment surface rescoring* (FSR), as opposed to the *structured fragment rescoring* (SFR) of equation (3.11). The corresponding accuracy-based estimator is given in (3.13), where the string edit distance is represented by $\delta_{dl}(d_e, d_e^o)$:

$$\mathsf{Acc}_{\mathsf{FSR}}(\langle d_f, d_e\rangle) = \max_{\langle d_f^o, d_e^o\rangle \in \mathbf{D}^o : d_e^o \in \Omega_{\mathcal{O}}(\mathsf{source\_span}(\langle d_f, d_e\rangle))} -\delta_{dl}(d_e, d_e^o) \tag{3.13}$$

As with the previous estimators, the target side of a fragment pair is compared to all oracle fragments associated with its same source span, and the chosen score is the one obtained when comparing to the fragment it is most similar to.

It should be noted that to compare the yield of the target side of a candidate fragment to that of an oracle fragment, a standard string-based evaluation metric could be used, such as Bleu (Papineni et al., 2002) the F-Measure (Turian et al., 2003) or TER (Snover et al., 2006). We chose to use the string edit distance for efficiency reasons: estimating the Bleu or F-Measure score for all of the fragments involved in the rescoring process would be too computationally expensive. In addition, the yields which we are comparing are significantly shorter than an entire sentence, which means that these evaluation metrics (which in the case of Bleu were designed to work at the *document* level) might not be suitable for this task. In particular, TER consists of a metric which computes the edit distance of the string, allowing an additional edit operation which moves large contiguous sequences of words to another location in the string ("phrasal shifts"). We feel that the added benefit of this edit operation would not be fully exploited when comparing short strings such as fragment yields, and that the use of the standard string edit distance is a suitable substitute.

3.3.5 Experimental Setup

We evaluated the impact on translation quality of our new scoring method by performing Spanish-to-English translation, and comparing the performance of the accuracy-informed system with the baseline DOT system which only uses fragment relative frequencies for scoring.

To create a parallel treebank to train our systems, we randomly selected 10,000 sentences from the Europarl corpus (Koehn, 2005). We should note that this data set size is relatively small. However, this is an order of magnitude increase compared to previously published experiments with DOT (Hearne and Way, 2006). As we explain in Chapter 4, DOT has difficulties scaling up to larger training data sizes because of the computational challenge of managing the large amount of fragments which are extracted from the parallel treebannk. We develop in Chapter 4 (Section 4.4) methods to further improve DOT's scalability, enabling one additional order of magnitude increase in training data size.

We parsed the target side of the parallel corpus using the Berkeley parser (Petrov and Klein, 2007), and the source side using a Spanish version (Chrupała and van Genabith, 2006) of Bikel's parser (2002), trained on the Cast3LB Spanish treebank (Civit and Martí, 2004).

The Cast3LB treebank contains a rich set of part-of-speech tags which provide detailed morphological information, such as number, gender, person, etc. We found this tag set to be too fine-grained for our purposes: with the training data sizes we are using, the large amount of different tags leads to poor grammar coverage. Therefore, after obtaining the Spanish parse trees (using the unmodified original parser), we modified the part-of-speech labels in the trees that we obtained so that only the first sub-categorization of the tag was included, and the rest was discarded. For example, for the case of nouns, the sub-classification as proper or common was kept, but we discarded gender and number information. The list of POS tags we obtained is given in Appendix A. The remaining non-POS tags are as defined by Chrupała and van Genabith (2006). For both Spanish and English, we removed unary chains by keeping the uppermost label in the chain and discarding the rest.[4]

After obtaining constituency parse trees for both sides of our parallel corpus,

[4]As our training corpora are not of a very large size, the more traditional solution of creating a new label by concatenating the labels of all nodes involved in a unary chain resulted in labels that were too fine-grained.

we sub-sententially aligned source and target parse trees using a tree-to-tree aligner (Tinsley et al., 2007a).[5] From this parallel treebank we obtained the source and target PCFG used by the Goodman reduction of DOT, as described in (Hearne, 2005) and explained in Section 2.3.5, although on an order of magnitude larger amount of training data than on previous experiments with (supervised) DOT (Hearne and Way, 2003, 2006).

To perform the rescoring, an estimation corpus is required, as explained in Section 3.2.2. To create this corpus, we use the 10K sentence pairs from which the grammars were extracted (i.e. the training data[6]), with an additional 10K randomly selected and previously unseen sentence pairs. Our estimation corpus is therefore composed of 20K sentence pairs, 50% of which were seen during training, with the remainder being composed of previously unseen sentences. To investigate the impact that the amount of additional unseen data has on the estimation method, we also obtained 20K additional previously unseen sentence pairs, and perform experiments where our feature is estimated using the combined 40K sentence pairs as the estimation corpus. As explained in Section 3.2.2, the sentence pairs used as an estimation corpus do not require parsing or sub-sentential alignment. However, this is a substantial amount of parallel sentences which need to be obtained in addition to those used for training. We explain in Section 3.4.4 (when performing experiments with PB-SMT) a technique which can be exploited to lessen the need for additional unseen data, and leave the adaptation of this technique for DOT as future work.

While developing our system, we repeatedly evaluated its performance on a small test set comprising 200 sentences, which we call our development test set. We evaluated the performance of our final systems on a larger test set, consisting of 2000 randomly chosen sentences. To allow for a reasonably quick experimental turnaround time, all sentences (including the 10K sentence pairs from which we

[5]http://www.ventsislavzhechev.eu/Home/Software/Software.html

[6]As previously explained in Section 3.2.2, the use of training data as estimation corpus is possible due to our use of N-best lists during estimation, rather than just considering the most probable translation.

extract our grammars, the additional 30K sentence pairs used as estimation corpus, and the test and development test sets) were restricted to a length of up to 20 words.

Our system translates an input sentence by first obtaining a chart representation of the N-best parses of the input sentence using the source Goodman grammar,[7] and then using the target Goodman grammar to obtain translations for each substitution site in the source chart (cf. Section 4.4 in Chapter 4 for a more detailed explanation of the architecture of our system). A beam is maintained in which at most k translations are considered for each substitution site. In our experiments, we use $N = 10,000$ and $k = 5$, and obtain the final translations from the target-side yield of the most probable derivation.

Statistical significance was tested by paired bootstrap resampling (Koehn, 2004). In our discussion, absolute scores for Bleu and the F-Measure are reported as percentages.

3.3.6 Experimental Results

In this section, we evaluate the impact that our accuracy-based features have on translation quality when incorporated into a baseline relative frequency system. We used the baseline system to translate our estimation corpus, scoring fragments which occurred in the N-best lists using the metrics defined in Sections 3.3.2–3.3.4. Once accuracy-based scores were obtained, we integrated them into the baseline system by using equation (3.5). As this requires every fragment to receive an accuracy-based score, and as fragments which were not observed during scoring cannot obtain one, we compute a default score equal to the median of all accuracy-based scores observed. Unscored fragments were then assigned this default score.

To integrate our new scores in the model, values for α_0 and α_1 in equation (3.5) must be determined. Although in future work an automatic method (such as Minimum Error Rate Training (Och, 2003)) could be used to determine these

[7]To obtain the N-best parses, our system implements the algorithm described by Jiménez and Marzal (2000)

		0.2-0.8	0.4-0.6	0.5-0.5	0.6-0.4	0.8-0.2	Baseline
Bleu	SFR	10.30	10.31	**10.32**	10.27	10.08	
	NSFR	8.31	9.37	9.53	9.66	9.90	8.78
	FSR	10.19	10.25	10.18	10.19	9.93	
Nist	SFR	3.792	3.805	**3.808**	3.800	3.781	
	NSFR	3.431	3.638	3.661	3.693	3.722	3.582
	FSR	3.784	3.799	3.792	3.795	3.764	
F-Measure	SFR	**40.92**	40.82	40.86	40.84	40.78	
	NSFR	37.53	39.50	39.93	40.38	40.78	38.21
	FSR	40.83	40.85	40.87	40.91	40.67	

Table 3.2: Results on test set. Estimation corpus of 20K sentences. *SFR* stands for Structured Fragment Rescoring, *NSFR* for Normalized SFR and *FSR* for Fragment Surface Rescoring. A column labelled *i-j* indicates the corresponding system was trained with $\alpha_0 = i$ and $\alpha_1 = j$ in (3.5). Underlined results are statistically significantly better than the baseline at $p = 0.01$.

weights, in the present experiments we set them manually, investigating the impact of different weightings for each feature.

In order to understand the strength of our baseline DOT system, a comparison to a standard system such as the the Moses PB-SMT system (Koehn et al., 2007) would be beneficial. However, a direct comparison between these systems would be unfair, since unlike PB-SMT, DOT is not able to exploit multiple features whose weights are optimized with MERT. In particular, DOT lacks a language model, one of the key features in PB-SMT. To provide a fairer comparison, we train the Moses system using the training corpus from which our grammars were obtained, using no language model and using uniform feature weights. We used this system to decode our development test set, and as a result we obtained a Bleu score of 10.72, which is comparable to the 10.82 Bleu score obtained by our baseline on the same set (cf. Table 3.4 for the complete set of results in the development test set). The score obtained by the PB-SMT system when allowing it to exploit a language model and when tuning feature weights using MERT is 23.59 Bleu points.[8]

Table 3.2 gives translation quality results on our test set for the case of an esti-

[8]As we consider these results more a guide to the reader than part of our main results, these scores were not included in our tables.

mation corpus of 20K sentences. In this table, columns labelled *i-j* indicate that the corresponding system was trained using parameters $\alpha_0 = i$ and $\alpha_1 = j$ in equation (3.5). The baseline DOT system achieves a score of 8.78 BLEU points. As can be observed in Table 3.2, all of the metrics we investigated are able to bring statistically significant improvements over the baseline. The best result for BLEU and NIST is obtained by the absolute tree edit distance (SFR), at $\alpha_0 = 0.5$ and $\alpha_1 = 0.5$. This system achieves an improvement over the baseline of 1.54 BLEU points, a 17.53% relative improvement. The best result for the F-Measure is also obtained by SFR, although using the weights $\alpha_0 = 0.2$ and $\alpha_1 = 0.8$. Although the normalised tree edit distance (NSFR) obtains statistically significant improvements over the baseline, it underperforms compared to the unnormalised metric (the difference between SFR and NSFR is statistically significant at $p = 0.01$ for most weight assignments). In addition, the scores obtained using NSFR decrease as the weight α_1 assigned to it increases. We believe this to be caused by the difference in magnitude between the NSFR scores and the likelihood scores. Most of the likelihood scores we observed are quite small numbers, while the normalized edit distance scores have a wide range of values which go from zero[9] to one, and which makes the differences in magnitude too large. The weights assigned to each score must be set accordingly to compensate for the difference in magnitudes, but it appears that the manual setting of these weights fails to do so (cf. Section 3.4, where we automatically assign these weights when adapting our scoring method to PB-SMT).

For most configurations the difference between SFR and FSR was not statistically significant at $p = 0.05$. Our analysis indicated that surface differences tended to co-occur with structural differences. We hypothesize that as we scale up to larger and more ambiguous grammars, the system will infer more derivations with the same yields, rendering a larger difference between the quality of the two scoring mechanisms.

When we increase the estimation corpus size from 20K to 40K sentence pairs

[9] As directly assigning a score of zero is too harsh, in practice we assign a small minimum score.

		0.2-0.8	0.4-0.6	0.5-0.5	0.6-0.4	0.8-0.2	Baseline
Bleu	SFR	**10.59**	10.58	10.41	10.38	10.08	
	NSFR	8.61	9.71	9.90	9.96	9.93	8.78
	FSR	10.49	10.48	10.35	10.38	10.06	
Nist	SFR	**3.841**	3.835	3.810	3.807	3.785	
	NSFR	3.515	3.694	3.713	3.734	3.727	3.582
	FSR	3.834	3.833	3.820	3.816	3.784	
F-Measure	SFR	**41.12**	40.99	40.86	40.88	40.75	
	NSFR	38.16	40.39	40.69	40.90	40.75	38.21
	FSR	41.03	41.02	41.01	40.98	40.72	

Table 3.3: Results on test set. Rescoring on 40K sentences. Underlined are statistically significantly better than the baseline at $p = 0.01$.

(Table 3.3), we obtain results which follow a similar pattern than those observed in Table 3.2, with all our evaluation metrics following a similar trend. The best results are again obtained using SFR, which brings a 1.81 absolute Bleu points compared to the baseline (a 20.71% relative improvement). The best Bleu result in Table 3.3 (SFR with $\alpha_0 = 0.2$ and $\alpha_1 = 0.8$) is statistically significantly better than the best result in Table 3.2 (SFR with $\alpha_0 = 0.5$ and $\alpha_1 = 0.5$) at $p = 0.02$.

We note that out of the 655,000 PCFG rules in the grammar, 275,000 of them receive an accuracy-based score when estimating our feature over 20K sentence pairs, while the remainder are assigned the default score. This number goes up to 280,000 rescored rules when using 40K sentence pairs as an estimation corpus. With such a small difference in the percentage of rescored rules, and the 20K estimation corpus being included in the 40K one, we are inclined to believe that the differences in performance seen between Table 3.2 and Table 3.3 are due to more accurate estimates rather than to an increased number of fragments obtaining an accuracy-based score. In addition, it is interesting to note that, although the differences between the scores obtained by different weights assignments for α_0 and α_1 are low, when using the more reliable estimates obtained by estimating over 40K sentence pairs, translation quality scores seem to improve as our new feature is given a higher weight than the relative frequency feature.

		0.2-0.8	0.4-0.6	0.5-0.5	0.6-0.4	0.8-0.2	Baseline
Bleu	SFR	11.34	**12.12**	11.94	11.97	11.78	
	NSFR	9.68	10.99	11.38	11.63	11.30	10.82
	FSR	11.40	11.49	11.72	11.91	11.72	
Nist	SFR	3.653	**3.727**	3.723	3.708	3.694	
	NSFR	3.376	3.530	3.554	3.616	3.572	3.493
	FSR	3.655	3.675	3.698	3.701	3.675	
F-Measure	SFR	44.84	**45.47**	45.36	45.33	45.08	
	NSFR	41.44	43.38	44.18	44.79	44.26	42.31
	FSR	44.68	44.91	45.15	45.19	44.82	

Table 3.4: Results on the development test set used to obtain examples for discussion. Rescoring on 40K sentences.

3.3.7 Discussion

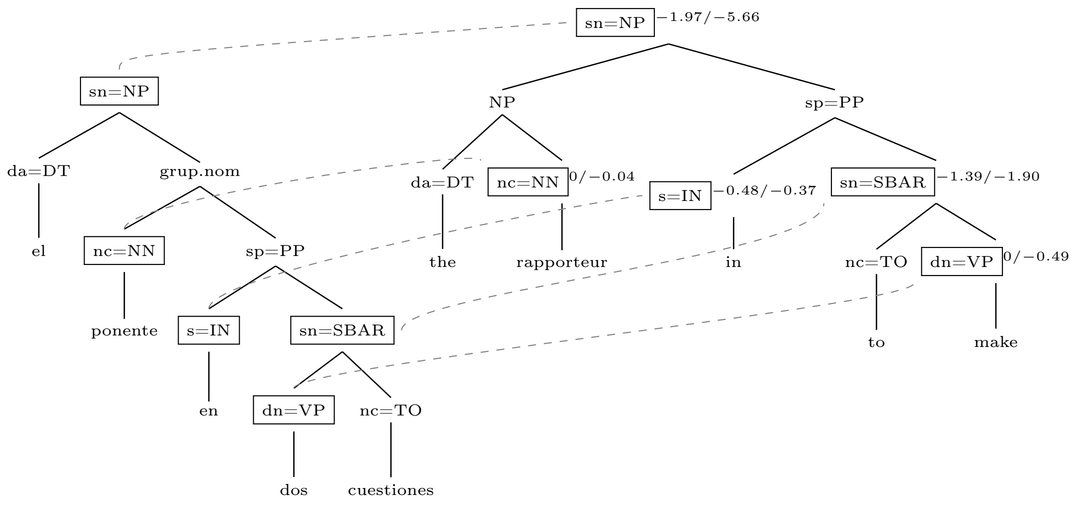
To understand the impacts that the introduction of our accuracy-based feature has on output translations, we investigate the derivations used to translate our 200-sentence development test set with the systems estimated over 40K sentences. The results obtained in this set are provided in Table 3.4. These results follow a pattern roughly equal to the ones in Tables 3.3 and 3.2.

Figure 3.8(a) gives a sentence from our development test set, along with its reference and the translations produced by the baseline system and our best-scoring system. As can be observed, the translation produced by the rescored system is of better quality than the one generated by the baseline. To understand the way in which each translation was generated, we give in Figure 3.8(b) and (c) the highest-scoring derivation which generates both translations (we omit the first part of the derivation, which generates the first words shared by the translations output by both systems). Boxed nodes denote substitution sites, and scores in superscripts denote the score of the sub-derivation according to the baseline (score on the left) and to the SFR system (score on the right). We see that the rescoring procedure brings not only changes in lexical choice, but the structure of the derivation presents differences as well, with the rescored system preferring a longer derivation than the one used by the baseline system. Of special interest is the score assigned to the translation of

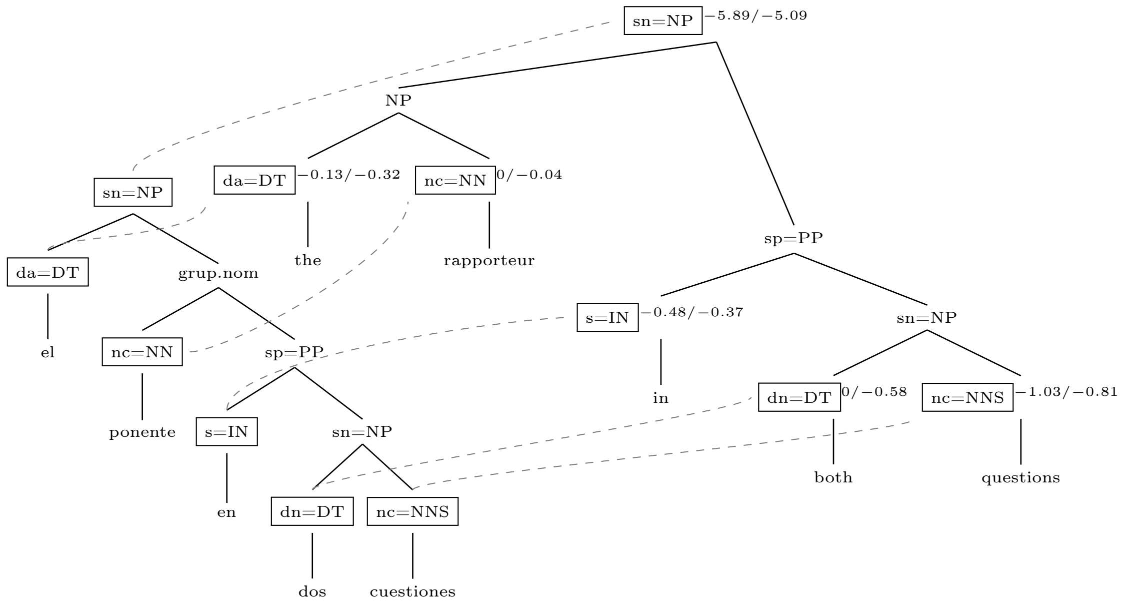
Source: Estoy de acuerdo con el ponente en dos cuestiones
Reference: I agree with the rapporteur on two issues

Baseline: I agree with the rapporteur in to make
SFR: I agree with the rapporteur in both questions

(a) A sentence from our development test set, along with its reference translation and the translations produced by the baseline system and the system with SFR rescoring.



(b) Best-scoring derivations for the translation produced by the baseline system. Superscripts give the score of a subderivation according to the baseline system (left score) and to the SFR system (right).



(c) Best-scoring derivations for the translation produced by the system rescored by SFR .

Figure 3.8: Best-scoring derivations for the translations of a sentence, according to the baseline system (b) and the SFR system (c). Boxed nodes are substitution sites.

"dos" (Spanish for "two") to the word "make". While the relative frequency baseline assigns a very high log-score of 0 to this translation, the rescored system assigns it a lower score of -0.49, which causes the more appropriate translation "both questions" to be preferred in the rescored system.

3.3.8 Conclusions

In this section we have adapted the scoring method outlined in Section 3.2 to the particular case of DOT. Our results show that significant improvements in translation quality can be achieved by enhancing a DOT system so that translation quality is taken into account in the scoring. Our experiments indicate that taking the internal structure of fragments into account is beneficial, although the difference with the metric that only takes fragment frontiers into account is small, and extracting grammars from larger parallel treebanks might reduce this difference. Obtaining accuracy-based scores from a larger estimation corpus leads to better scores. The improvement seems to be caused by more reliable estimates, rather than by a larger amount of fragments receiving a score.

3.4 Accuracy-Based Scoring for PB-SMT

As was apparent from our exposition in Section 2.2, and as we noted in Section 3.1, PB-SMT scores alternative translations by combining the score assigned to phrases by multiple model components, such as phrase translation and language model probabilities, which are induced in the training stage by the use of relative frequencies. Although the contribution of each component to the final score is weighted so as to optimise translation quality on held-out data via Minimum Error-Rate training (MERT) (Och, 2003), the individual components themselves only attempt to increase the likelihood of the training corpus, and only indirectly impact in translation quality. Since our ultimate goal in training a PB-SMT system is to maximize the quality of its translations when confronted with unseen data, we now adapt the scoring method presented in Section 3.2 to the case of PB-SMT, which enables us to introduce a new feature which indicates how likely a phrase pair is to contribute to good translations.

In this section we investigate the effects of accuracy-based scoring specifically for PB-SMT systems. We use a baseline PB-SMT system (Koehn et al., 2007) to obtain N-best lists, and then choose oracle translations according to a range of evaluation metrics. We then compare each phrase pair in the N-best list against phrases present in the oracle translations, and assign a score to each phrase pair according to how similar they are to those oracle phrase pairs.

Unlike most previous work related to translation quality-driven scoring in the context of (hierarchical) phrase-based MT, e.g. (Liang et al., 2006; Watanabe et al., 2007), our approach has, as mentioned in Section 3.2, the benefit of simplicity. This means that it can be easily performed using off-the-shelf decoders and tuning algorithms like MERT, and is therefore readily available to PB-SMT practitioners. In addition, the estimation of our feature is easily parallelizable, as sentences are processed independently of each other. Our experiments show that our approach leads not only to translation quality improvements, but also to improvements in translation speed and memory consumption.

Unlike in the experiments in Section 3.3, we do not require large amounts of additional held-out data here, as we estimate the new features using only the parallel data used to train the baseline system. Furthermore, unlike in the experiments with DOT, we evaluate the impact of different evaluation metrics when selecting oracles (namely BLEU and the F-Measure).

The scoring model in PB-SMT is —unlike the default one in DOT— a log-linear framework explicitly formulated to allow the inclusion of additional features. Once we have estimated the accuracy-based function $\mathsf{Acc}(f_i, e_i)$, this will allow us to directly incorporate a new feature h_{Acc} into the log-linear model of equation (2.1), as in (3.14):

$$h_{\mathsf{Acc}}(e_1^I, f_1^I) = \prod_{i=1}^{I} \mathsf{Acc}(f_i, e_i) \tag{3.14}$$

3.4.1 Oracle selection

As previously explained (Sections 2.2.2 and 3.2), we assume that we can recover the phrase alignment a used to generate each candidate translation. We repeat the function Ω defined in (3.2) here for convenience. This function uses the alignment a to indicate the mapping between a source sentence span (l, m) and the corresponding set of target phrases in the oracle translations $\mathcal{O}$ (*oracle phrases*), as in (3.15):

$$\Omega_{\mathcal{O}}(l, m) = \{\tilde{e}_o | \exists\, \mathbf{t} \in \mathcal{O} : \tilde{e}_o \in \mathbf{t} \wedge a(\tilde{e}_o) = (l, m)\} \tag{3.15}$$

We experiment with two translation-quality metrics $\mathbf{E}$, namely BLEU and the F-Measure, which the following two subsections describe. In the future we will consider evaluating the effects of additional evaluation metrics, such as TER (Snover et al., 2006).

BLEU

As we explained in Section 2.5.1, the BLEU score (Papineni et al., 2002) computes a geometric mean of the unigram to N-gram precisions between a candidate sentence and a set of references (typically $N = 4$). If there is not at least one N-gram match between the candidate sentences and the reference set, BLEU is undefined. In those cases, we define BLEU to be 0. Since our aim is to use BLEU not at the document level where this phenomenon would be rare, but at the sentence level in equation (3.3), this is problematic because in practice BLEU will be 0 for most sentences. We thus follow (Liang et al., 2006) and approximate BLEU by a smoothed version that combines the scores of BLEU for various N, as in (3.16):

$$\text{sBLEU} = \sum_{i=1}^{N} \frac{\text{BLEU}_i}{2^{4-i+1}} \tag{3.16}$$

Note that the direct use of document-level approximations of Bleu such as those used in (Watanabe et al., 2007) would be impractical in our approach, as it would introduce dependencies across sentences which would limit parallelisation.

F-Measure

The General Text Matcher (GTM) (Turian et al., 2003) computes the F-Measure between a candidate translation and a reference using the notions of precision and recall. This computation is parameterised by an exponent, which adjusts the weights of longer n-grams in the score. For the purpose of obtaining oracle translations, in these experiments we use the F-Measure with an exponent of 1.5, which was estimated by evaluating the quality of the oracles obtained on held-out data.

3.4.2 Similarity Metrics

To estimate the function Acc in (3.14), we need a notion of similarity between the target phrases present in a candidate translation $\tilde{e}_c$ and the ones present in an oracle translation $\tilde{e}_o$. We relate target phrases in candidate translations to phrases in oracle translations by considering the source-sentence span they translate. To achieve this, the mapping a between the source-sentence span and the target phrases as determined by the decoder is required. The estimation of Acc will be limited to those phrases in a candidate translation for which oracle phrases exist which translate the same source span, i.e. we only score target phrases for which $\Omega(a(\tilde{e}_c)) \neq \emptyset$.

Edit distance scoring

To compare two phrase pairs with the same source side and different target translations, we use the (word-level) Levenshtein distance $\delta_{dl}(\tilde{e}_c, \tilde{e}_o)$ (Damerau, 1964) between the target side of the phrase pairs. This measures the amount of insertions, deletions, or substitutions of words needed to transform the candidate phrase into the oracle phrase. For a phrase $\tilde{e}_c$ (in the candidate translation) which is translated from a source span $a(\tilde{e}_c)$, we assign as a score the exponential of the negative edit distance between $\tilde{e}_c$ and the oracle phrase $\tilde{e}_o$ it is most similar to, as in (3.17):

$$\mathsf{Acc}_{ed}(f_i, \tilde{e}_c) = \max_{\tilde{e}_o \in \Omega(a(\tilde{e}_c))} \exp(-\delta_{dl}(\tilde{e}_c, \tilde{e}_o)) \tag{3.17}$$

Note that after repeating this for all sentences in the held-out set, the score assigned to a phrase pair is the average of the scores it obtained.

Normalised Edit Distance

As was the case with the metric in Section 3.3.2, a potential problem with the metric in (3.17) is that we would expect that on average the edit-distance between a candidate phrase $\tilde{e}_c$ and an oracle phrase $\tilde{e}_o$ would grow with phrase length. Since this could introduce an unwanted bias against long phrases, we also experiment with a score that normalises the edit distance by the amount of words in the target phrase, as in (3.18):

$$\mathsf{Acc}_{norm}(f_i, \tilde{e}_c) = \max_{\tilde{e}_o \in \Omega(a(\tilde{e}_c))} 1 - \frac{\delta_{dl}(\tilde{e}_c, \tilde{e}_o)}{\max(|\tilde{e}_c|, |\tilde{e}_o|)} \tag{3.18}$$

3.4.3 Reordering Model

We also re-estimate the lexicalised reordering model by considering the order between phrases involved in oracle translations. For each phrase pair involved in an oracle translation, we obtain the orientation by considering both the previous and next phrases as in (Koehn et al., 2005). We thus obtain a list of triples (f_i, e_i, o), where (f_i, e_i) is a phrase pair and $o \in \{\text{monotone}, \text{swap}, \text{discontinuous}\}$, which we use to estimate $p_{\mathsf{Acc}}(o|f_i, e_i)$.

To incorporate this information into the model, phrases for which we did not extract orientation information are assigned a default score equal to the median score for a particular orientation of the scored phrases. Then, for some constant q, we interpolate this new reordering score with the original score $p(o|f_i, e_i)$, as in

(3.19):

$$p_r(o|f_i, e_i) = q \cdot p(o|f_i, e_i) + (1 - q) \cdot p_{\mathsf{Acc}}(o|f_i, e_i) \tag{3.19}$$

3.4.4 Estimation Corpus

In our experiments with DOT in Section 3.3.5, we created an estimation corpus (as explained in Section 3.2.2) by combining the training data with previously unseen data. Since this increases the amount of parallel resources required by the system, a way to avoid the need of additional parallel data would be desirable.

We propose to avoid the need of additional unseen data to estimate the accuracy-based feature by using Deleted Estimation (Jelinek and Mercer, 1985), a technique that has successfully been used in Data-Oriented Parsing (Zollmann and Sima'an, 2005) and a wide range of machine learning approaches such as decision tree induction (Breiman et al., 1984).

In a similar way to 10-fold cross validation, we create a new training corpus T by keeping 90% of the sentences in the original training corpus, and a new estimation corpus H by using the remaining 10% of the sentences. Using this scheme we make 10 different pairs of corpora (T_i, H_i) in such a way that each sentence from the original training corpus occurs in exactly one H_i for some $1 \leq i \leq 10$, which ensures that each sentence is observed during estimation. We train 10 different systems using each T_i, and use each system to estimate Acc on its corresponding held-out set H_i. We then consider all of the scores obtained by each phrase pair in any H_i, and assign the average of those scores (Jelinek and Mercer, 1985) as a final estimate to each phrase pair. The new feature is then added to the baseline system, trained on the whole original training set.

Note that if we were to adapt this technique to the case of DOT, care should be taken to avoid overlaps between the Goodman indexes assigned to nodes in each of the smaller systems. When creating each of the systems using 90% of the training data, Goodman indexes should be assigned in a way such that when finally combining all of the scored fragments, they match those assigned to the original baseline system.

3.4.5 Experimental Setup

We empirically evaluate the impact of our new feature by performing Spanish-to-English translation and comparing against a baseline system trained using standard parameters.[10] In all of our experiments we use the Moses toolkit (Koehn et al., 2007). We train on the training section of the Spanish–English Europarl corpus as provided for the Fourth Workshop on Statistical Machine Translation (WMT09).[11] We discarded sentences with more than 40 words,[12] which left us with 1,083,773 sentences for training. We use the first 500 sentences of dev2006 as a tuning set for MERT. We use test2006 as a development test set, and test2008 as the final test set (each containing 2,000 sentence pairs). We use the 5,000-best translations returned by our decoder to select oracles and perform the scoring (note that on our experiments with DOT in Section 3.3.6, we obtained oracles from the 10,000-best parse trees of the input sentence).

As in our previous experiments, statistical significance was tested by paired bootstrap resampling (Koehn, 2004), and absolute scores for Bleu, Meteor and the F-Measure are reported as percentages.

3.4.6 Dealing with Unestimated Phrase Pairs

As mentioned in Section 3.4.4, each sentence will appear in one held-out set, and will be decoded by a system which was trained on a reduced section of the training corpus which does not include this sentence. Even though this ensures that all of the training sentences will be considered in the estimation process, this does not

[10]The features we use are those introduced in Section 2.2 (i.e. phrase translation probabilities in both language directions, lexical weighting in both language directions, language model, word and phrase penalties, and lexicalized orientation-based reordering model). The maximum phrase length used was 7, and the reordering window was set to 6.

[11]http://statmt.org/wmt09/

[12]Note that our experiments with DOT used sentences of up to 20 words. The efficiency of the PB-SMT model allows us to increase the length of the sentences while still achieving reasonable decoding times.

guarantee that every phrase pair will receive a score according to Acc, as a phrase pair needs to occur in an N-best list translating the same source span as an oracle phrase pair in order to receive a score. In fact, out of the 46,994,471 phrase pairs in the baseline phrase table, only 6,056,274 of them can obtain an accuracy-based score when using the F-Measure to select oracles, and just 5,994,142 when using sBLEU.

We experiment with two ways of dealing with unscored phrases. Firstly, we use the same technique used in our experiments with DOT, and calculate a default score equal to the median score among phrase pairs that receive a score, and assign this score as the Acc estimation for phrase pairs for which no accuracy-based score was obtained. Secondly, we build a system which uses only the phrase pairs that receive some score, namely just 13% of the phrase table in the baseline system.

3.4.7 Experimental Results

In this section, we evaluate the effect that our phrase-distance metrics have, the impact of rescoring the reordering-model, and the effects of using different oracle selection metrics. While developing our system, we repeatedly tested our incremental improvements on a development test set (reported as scores between squared brackets), and then performed our evaluation on the test set to obtain our final results. Results with single underlines are statistically significantly better than the baseline at $p = 0.05$ and those with double underlines are significantly better at $p = 0.01$. Unless specifically mentioned, the oracle selection metric in our experiments is the F-Measure.

Accuracy-Based Feature and Reordering Model Rescoring

We used the methods described in Section 3.4.2 to assign new scores to phrase pairs and to rescore the reordering model. Our Accuracy-Based (AB) feature encoding the average similarity between a phrase in a candidate translation and a phrase in an oracle translation was calculated using two metrics, namely the edit-distance ("ed")

System	Bleu	Nist	Meteor	F-Measure
Baseline	32.72 [32.29]	7.7941	56.55	64.88
AB feature + default reordering (F-Measure Oracles)				
ed	32.64 [32.41]	7.7962	56.55	65.04
norm	$\underline{\underline{33.16}}$ [$\underline{\underline{32.76}}$]	$\underline{\underline{7.8449}}$	$\underline{\underline{56.97}}$	$\underline{\underline{65.30}}$
AB feature + rescored reordering (F-Measure Oracles)				
ed	$\underline{\underline{33.03}}$ [$\underline{\underline{32.71}}$]	$\underline{\underline{7.8582}}$	$\underline{\underline{56.77}}$	$\underline{\underline{65.18}}$
norm	$\underline{\underline{\mathbf{33.41}}}$ [$\underline{\underline{\mathbf{32.83}}}$]	$\underline{\underline{\mathbf{7.8879}}}$	$\underline{\underline{\mathbf{57.14}}}$	$\underline{\underline{\mathbf{65.43}}}$
AB feature + rescored reordering (sBLEU Oracles)				
ed	$\underline{\underline{33.11}}$ [$\underline{\underline{32.56}}$]	$\underline{\underline{7.8379}}$	$\underline{\underline{56.97}}$	$\underline{\underline{65.30}}$
norm	$\underline{\underline{33.11}}$ [$\underline{\underline{32.65}}$]	$\underline{\underline{7.8513}}$	$\underline{\underline{56.91}}$	$\underline{\underline{65.28}}$

Table 3.5: System performance with accuracy-based features and default score for unscored phrases. "ed" and "norm" represent the metrics of equations (3.17) and (3.18), respectively.

in equation (3.17), and the normalised edit-distance ("norm") of equation (3.18).

To single out the contribution of our accuracy-based feature, we first conducted experiments with this feature and a default reordering model, assigning a default score to unscored phrase pairs. The first five rows in Table 3.5 show the performance of the baseline system (without our AB feature) and the system with the baseline features and the addition of our new feature. The effect of using two different similarity metrics (ed and norm) is also presented. As expected, the normalised edit-distance metric (5th row in Table 3.5) yields higher translation quality compared to the (absolute) edit-distance. While the "ed" metric is not able to produce significantly better translations, using "norm" leads to statistically significant gains over the baseline across all evaluation metrics we used. Using this setup, there is a 0.44 absolute improvement in Bleu, corresponding to a 1.34% relative improvement. This contrasts with the results from our experiments with DOT in Section 3.3.6, where the normalised edit-distance underperforms when compared to the absolute edit-distance. We believe this might be a result of our use of MERT, which can determine a weight for our feature that properly scales it to the magnitudes of the other model components, while with DOT we resorted to arbitrarily assigning

Feature	**Baseline**	**Default Reordering**	**Rescored Reordering**
Acc_{norm}	-	0.4513	0.4707
Language Model	0.2512	0.1756	0.1895
$p(f \mid e)$	0.1600	0.0818	0.0952
$lex(f \mid e)$	0.1498	0.0939	0.1252
$p(e \mid f)$	0.1166	0.0274	0.0822
$lex(e \mid f)$	0.0064	0.0309	0.0118
Phrase Penalty	0.3687	0.2732	0.1405
Word Penalty	-0.0530	-0.1344	-0.1152

Table 3.6: Weight assigned by MERT to each (non-reordering) feature in the models of the baseline system, the system with our AB feature + default reordering, and the system with the AB feature + rescored reordering

a weight for our feature on a manual basis.

To investigate the combined effect of using both the Accuracy-Based feature and a rescored reordering model, we estimated a new reordering model using $q = 0.5$ in equation (3.19). Clearly we see the added value of this, as gains are observed across all evaluation metrics. The best system, i.e. using both an AB feature and a rescored reordering model with "norm" as the similarity metric, outperforms the baseline by 0.69 Bleu points, corresponding to a 2.11% relative improvement. We note also that the 0.25 absolute Bleu points improvement between the system with rescored reordering (8th row in Table 3.5) and the system with default reordering (5th row) is statistically significant at $p = 0.01$. The remaining evaluation metrics present a similar pattern compared to Bleu.

We give in Table 3.6 the normalised weights assigned by MERT to each of the (non-reordering) features, for the baseline system and for the systems with an accuracy-based feature estimated using the normalised edit distance. We see that after adding the AB feature, most of the original features have ceded their contribution to the overall scoring, which is now dominated by this feature. This shows that the translation quality improvements observed in Table 3.5 are due to the introduction of our feature. Interestingly, while 30% of the contribution of the language model feature to the overall score has been given away to our new feature, the phrase translation probabilities on each direction cede 48% and 76% of their weight. The greater loss in weight for the source-to-target direction is expected, as this is the direction our technique assumes. An experiment using a system with our accuracy-based feature and no phrase translation or lexical weighting probabilities (i.e. using also phrase and word penalties, features which are not estimated during training) results in a Bleu score of 31.26 when decoding our development test set. While this is a significant drop of 1.03 Bleu points, we nonetheless find it remarkable that the system is able to perform at a level not so distant from the baseline while ignoring relevant features such as phrase translation probabilities, especially considering that these features have been considered integral components in both word- and phrase-based SMT since their inception.

Oracle Selection Metric

The last two rows in Table 3.5 show the effect of using sBLEU (equation (3.16)) instead of the F-Measure to select oracles. As can be seen, the systems with F-Measure oracle selection consistently outperform those using sBLEU across all evaluation metrics, where the best system using the F-Measure has a gain of 0.3 absolute Bleu points over using sBLEU, corresponding to a 0.9% relative improvement. While this gain appears to be modest, it is statistically significant at $p = 0.01$, demonstrating the advantage of using the F-Measure over sBLEU for oracle selection. This is not surprising given that, as noted in Section 3.4.1, Bleu is specifically designed for document-level evaluation while the F-Measure is more suitable for evaluation at sentence-level.

We collected oracle selection statistics in order to further investigate this process. It turned out that 92.24% of the oracles are not the top hypothesis in the N-best list. In fact, if we evaluate the score obtained by using the 1-best hypotheses when decoding the training set to obtain the N-best lists, we obtain a Bleu score of 40.46, while using the oracle translations obtained by the F-Measure yields a score of 52.86 in Bleu. The corresponding score for sBLEU oracle selection is a Bleu score of 53.39. Given that the top hypothesis in the N-best list is the most likely translation

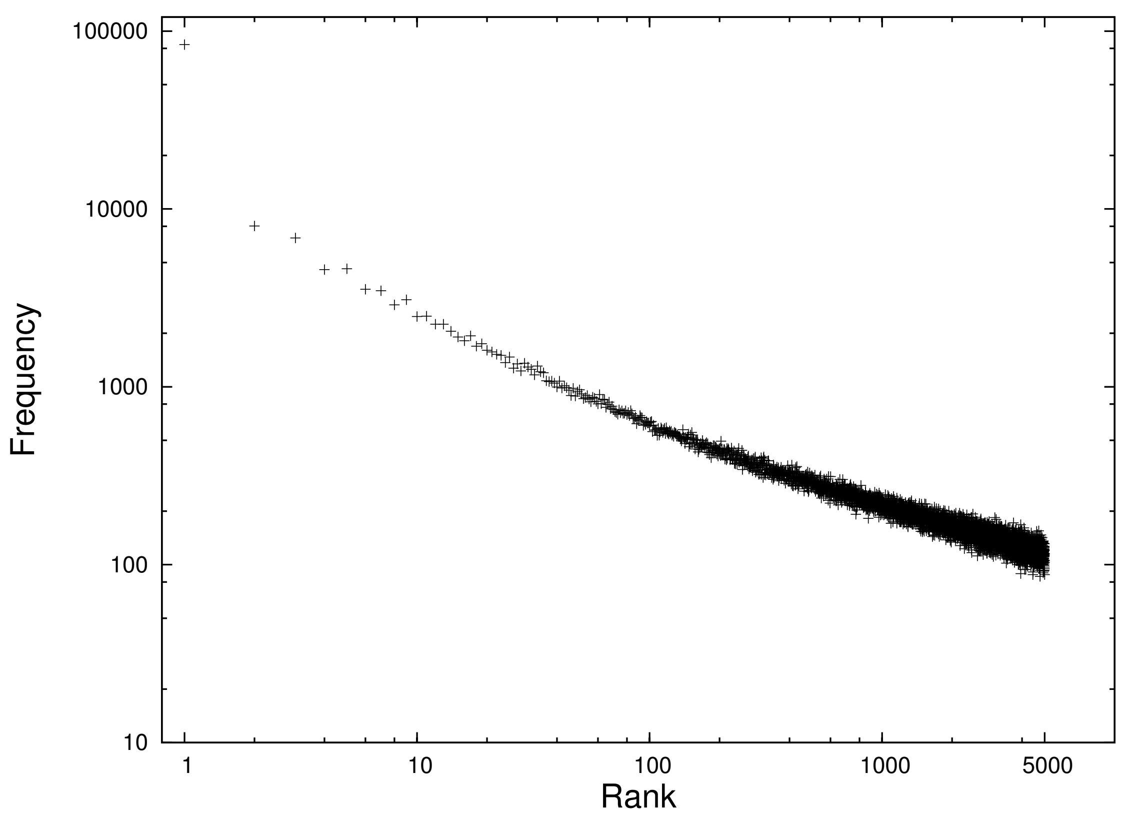


Figure 3.9: Oracle rank frequencies (logarithmic scale)

according to the current model parameters, it is clear that there is plenty of space for improving the model to allow for the best translation in the N-best list (the oracle) to be scored the highest. This confirms the rationale of our methods which can improve the model parameterisation and subsequently the translation results.

In order to show the rank of the oracle in the N-best lists, we plotted the frequency distribution of the ranks as shown in Figure 3.9. We can see that oracles are very frequently selected from the top 100 hypotheses of the N-best list. Hypotheses with a rank above 1000 may still be selected as the oracle, but with a much lower frequency (corresponding to the dense tail on the right of the graph). As a matter of fact, 7.76% of the oracle translations are the 1-best hypothesis (corresponding to the point at the top left corner of the graph), 11.41% are selected from the top-10 hypotheses, 19.88% from the top-100 hypotheses, 45.34% from the top-1000 hypotheses, and the remaining 54.66% are selected from hypotheses ranking from 1000 to 5000. It is clear that a large N-best list is crucial in order to select a better

System	Bleu	Nist	Meteor	F-Measure
Baseline	32.72 [32.29]	7.7941	56.55	64.88
AB feature + rescored reordering (F-Measure Oracles)				
ed	33.04 [32.66]	7.8665	56.93	65.29
norm	33.23 [32.60]	7.8835	**57.21**	**65.58**
AB feature + rescored reordering (sBLEU Oracles)				
ed	**33.35** [32.56]	7.8590	57.20	65.42
norm	33.25 [**33.19**]	**7.9236**	57.06	65.54

Table 3.7: System performance using only scored phrases

oracle translation. Given the behaviour observed in Figure 3.9, we would expect a large number of oracles to be found in ranks below 5000, which would indicate that if time constrains are not an issue, using an n-best list even larger than 5000 could be beneficial.

Discarding Unscored Phrases

The previous results were obtained by assigning a default score to phrases which were not able to receive an AB score during the scoring process. Unlike in DOT, where the Goodman reduction encodes all of the possible extracted fragments, in PB-SMT it is straightforward to discard those phrases which did not receive a score. From Table 3.7, we can see that using only those phrases that received a score yields improvements over the baseline across all evaluation metrics. There is an improvement of 0.63 absolute Bleu points over the baseline using sBLEU for oracle selection and "ed" as the similarity metric, corresponding to a 1.93% relative improvement over the baseline. We also observe a modest gain over using all phrases (Table 3.5) across most of the metrics (except for the Bleu score of the system using F-Measure for oracle selection and "norm" as the similarity measure). This is remarkable given that the system with the AB feature uses a phrase table 87% smaller than the one in the baseline, which leads to speed increases and memory consumption reductions.

Unlike in the experiments using the complete phrase table (Table 3.5), in this case there is a disagreement between the different evaluation metrics as to which

System	Bleu	Nist	Meteor	F-Measure
Baseline	32.72	7.7941	56.55	64.88
Random Pruning	18.17	5.6319	47.59	55.16
F-Measure Pruning	32.61	7.7675	56.57	64.86

Table 3.8: System performance with different pruning criteria. Results on test set.

oracle selection metric is better. Bleu and NIST (two closely related metrics) deem sBLEU to be the best method, while the best results for the F-Measure and Meteor are obtained when using the F-Measure to select oracles. We believe that the cause of this concordance between the method used to select oracles and the evaluation metric which achieves best results is the lack of a default score. When using the full phrase table, the majority (87%) of phrase pairs receive the same default score, which dilutes the impact of our feature. In this case, the accuracy-based score assigned to each phrase pair is more closely related to the scores that maximized a particular oracle selection metric, which leads to this concordance.

In order to confirm that the improvements observed in Table 3.7 are indeed due to our rescoring method and are not for some unknown reason a consequence of the reduced phrase table size, we performed an additional experiment in which the phrase table in the baseline system was pruned so that only 13% of its phrases were kept. We randomly selected 6,056,274 phrases from the baseline system (i.e. the same amount of phrases used by the system which used the F-Measure to select oracles) and allowed only these phrases to be used during decoding. The resulting system, as shown in Table 3.8, achieved a score of only 18.17 Bleu points, a 44.46% decrease compared to the baseline. Table 3.8 also gives, in its last row, results for a system which uses only those phrases which received a score when obtaining oracles using the F-Measure. However, our accuracy-based feature was not incorporated in this system, which instead uses the baseline feature set for scoring. This results in a system that performs at a level statistically insignificantly worse than the baseline, but which uses only 13% of the phrases present in the baseline phrase table. The remaining evaluation metrics follow a similar pattern, with the exception of an

Systems	**System 1 Better**	**Equal**	**System 2 Better**
B vs. AB	558	739	703
B vs. AB+O	515	793	692
B vs. R	633	555	812
AB vs. AB+O	482	1006	512
AB vs. R	616	712	672
AB+O vs. R	660	667	663

Table 3.9: Pairwise comparison of different systems via sentence-level evaluation

insignificant improvement in Meteor when using the F-Measure to filter the phrase table. These results confirm that our scoring method is able to determine a subset of phrase pairs from which the same performance can be achieved as when using the complete set of phrase pairs in the baseline system, and that incorporating our accuracy-based scores into this reduced system can lead to significant gains in translation quality. In the future it will be interesting to compare this phrase-table filtering method with techniques specifically designed for this purpose, e.g. (Johnson et al., 2007; Sánchez-Martínez and Way, 2009).

Sentence-level evaluation

In addition to the document-level automatic evaluation, we conducted a sentence-level evaluation using Meteor. Pairwise comparison was performed for four systems including the Baseline system (B), the system with the AB Feature and default reordering (AB), the AB Feature and rescored reordering (AB+O), and the system using only the rescored phrases (R). In the pairwise comparison, we count the number of sentences in the test set where one of the systems is better or both systems are equal.

The results in Table 3.9 are consistent with the document-level evaluation in Tables 3.5 and 3.7. We see that with AB-Scoring (B vs. AB), sentences with improved translations are far more numerous than those whose translations become worse (703 vs. 558). Adding both the AB feature and a rescored reordering model (B vs. AB+O and AB vs. AB+O) further improves system performance, with more

Source: la comunidad internacional no puede contentarse por más tiempo con esconder la cabeza como el avestruz (. . .)

Reference: the international community can no longer content itself with burying its head in the sand (. . .)

Baseline: $\overset{523092}{\text{the}}$ | $\overset{1385}{\text{international community}}$ | $\overset{2}{\text{cannot be satisfied}}$ | $\overset{31382}{\text{by}}$ | $\overset{233}{\text{more time}}$ | $\overset{9965}{\text{to}}$ | $\overset{8}{\text{bury our}}$ | $\overset{3}{\text{heads in the sand}}$ | (. . .)

Rescored: $\overset{1}{\text{the international community can no}}$ | $\overset{7}{\text{longer}}$ | $\overset{8}{\text{be content}}$ | $\overset{72416}{\text{with}}$ | $\overset{1}{\text{burying one 's head in the sand}}$ | (. . .)

Figure 3.10: Example output translation from the baseline and our best-scoring system

sentences receiving better translations. Using only rescored phrases yields substantial improvements over the baseline (B vs. R), and encouragingly, using only the rescored phrases does not result in any decrease in translation quality; there is instead a marginal gain of 3 sentences (AB+O vs. R).

In order to qualitatively assess some of the improvements to which our method leads, we give in Figure 3.10 some example output from the baseline system and from our best-scoring system, when translating a sentence in our development test set. In this figure, vertical bars represent the target-side phrase-segmentation used to build the sentence, and the numbers above the phrases indicate the amount of times that the corresponding phrase pair was extracted from the training corpus. We see that the baseline system uses short phrases which are very frequent in the training corpus. In contrast, the rescored system uses fewer (but longer) phrases, which do not occur as frequently in the training corpus, but which yield a better translation. This might be explained by the weights given in Table 3.6: the rescored system does not have to heavily rely on the amount of times a phrase pair occurs in the corpus, allowing it to use longer (and more infrequent) phrases when evidence has been observed that such a phrase typically leads to a good translation. We note that while the baseline system uses 32,728 phrase pairs to translate the whole development set, the rescored system can translate it using only 26,272 phrase pairs, an indication that this phenomenon might not be unique to this particular sentence.

3.4.8 Conclusions

In this section, we introduced additional features for PB-SMT which bring translation quality-related knowledge into the scoring of phrase pairs. On the WMT09 Spanish-to-English translation task, significant gains over the baseline are obtained across many evaluation metrics. Encouragingly, our method can also lead to a substantial reduction (87%) in phrase-table size without significant loss in translation quality. Given the size of the weight associated with our AB feature (Table 3.6), it is not an exaggeration to conclude that gains in MT quality and efficiencies in speed and memory usage are due almost entirely to our new feature. Although discarding key features such as phrase-translation probabilities and using only our feature leads to a loss in translation quality, doing so results in a system that performs nearly as well as the baseline, which suggests that a purely accuracy-based system could eventually be developed.

3.5 Summary

In this chapter we introduced a scoring method which is able to relate the translation units used by an MT system to a score representing the average edit distance between this unit and a unit which has been shown to maximize translation quality.

We adapted this general scoring method to the DOT translation system, which led to translation quality improvements compared to a baseline DOT model. We also showed that the usefulness of this scoring method is not limited to the case of DOT, by adapting it for use with the PB-SMT model. Results with this model are also positive, obtaining translation quality improvements compared to a standard PB-SMT system and providing more evidence for the usefulness of our method. For the case of PB-SMT our method not only leads to translation quality improvements, but we can also achieve such improvements with a system which is significantly faster and uses significantly fewer resources than the baseline.

In the following chapter we tackle our remaining research questions (**RQ3** and **RQ4** in Chapter 1). We address the issue of the limited feature set in DOT, by defining a log-linear model which can exploit arbitrary features of the source sentence. We take advantage of this log-linear model by introducing lexical features and a language model.

Chapter 4

Log-Linear Scoring and Lexical Selection for DOT

In this chapter we turn to the difficulty of incorporating additional features into DOT. We identify weaknesses in DOT's method of scoring and define features which we believe would be beneficial for DOT, and investigate their effect. We then incorporate these features into a new DOT model which is able to exploit both the original structural and reordering model and a new feature-based log-linear model.

We begin by motivating our approach in Section 4.1. We then introduce our new log-linear model in Section 4.2, which we exploit by defining features which condition a target word to the source lexical items it is aligned to, and by incorporating an n-gram language model. In Section 4.3 we explore the effects of two different estimation methods for our feature: Maximum Entropy and improved Kneser-Ney. Section 4.4 explains how our new model is integrated into the original DOT model, and how fast decoding speeds can be achieved. We evaluate the performance of our new system in Section 4.6 and give example translations in Section 4.7, and finally conclude in Section 4.8.

4.1 Motivation

As explained in Section 2.3, finding the most probable translation in DOT involves summing over all derivations with the same target yield. This is akin to finding the most probable sentence from an input lattice under DOP and other models, e.g. (Bod, 1992; Sima'an, 2004), which has been shown to be intractable (Sima'an, 1996). Given this, one has to resort to approximations such as sampling (Hearne and Way, 2006) or constraining the computation to the n-best derivations.

The Goodman reduction for DOT assigns to a particular fragment extracted from the parallel treebank a probability as in (4.1):

$$\frac{1}{\mathsf{subtrees}(A, B)} \tag{4.1}$$

where (A, B) is the pair of roots of the fragment pair, and $\mathsf{subtrees}$ is a function which counts the number of fragment pairs rooted by (A, B) which were extracted from the treebank. If we compare (4.1) with the fragment pair probability in (4.2):

$$P(\langle d_f, d_e\rangle) = \frac{|\langle d_f, d_e\rangle|}{\mathsf{subtrees}(\mathrm{root}(d_f), \mathrm{root}(d_e))} \tag{4.2}$$

this is not necessarily equal to the probability assigned by the model to the fragment pair, as the number of times that this particular fragment pair was extracted from the parallel treebank is not taken into account, i.e. $|\langle d_f, d_e\rangle|$ is missing. In theory this is not a problem, since when summing over derivations to compute the (string) translation probability, the probabilities assigned by the Goodman reduction end up being equal to the ones defined by the model, as each of the fragments rooted by (A, B) will add $1/\mathsf{subtrees}(A, B)$ to the overall probability, and the amount of times that a fragment pair was extracted will be implicitly incorporated into the score. However, this means that when using the n-best DOT derivations to approximate the most probable translation (as in the experiments in this thesis), if n is not large enough, fragment probabilities will tend to be approximately $1/\mathsf{subtrees}(A, B)$ rather than the correct probability. This means that a large fragment which is observed only a few times during fragment extraction might compete using a probability roughly equal to the probability of a smaller (and more frequent) fragment pair, just because they happen to share the same root pair.

In addition to this, DOT's relative frequency estimator is inherited from the probability estimator used in DOP, which has been shown to be inconsistent (Johnson, 2002) and biased toward the use of larger subtrees (Bonnema et al., 1999). We argue that DOT's bias and our inability to explore the complete derivation space, having instead to resort to approximations to obtain the most probable translation probabilities, mean that in practice DOT derivation probabilities are driven mainly by properties of the derivation structure (such as the size of the fragment pairs used), rather than by lexical equivalences: derivations which use a few fragments of large size will tend to be preferred, regardless of the words associated with these fragments. Some evidence for this can be seen in the findings of Hearne and Way (2006) who, despite using sampling instead of the n-best derivations, observe that in many cases the translation quality remains the same or improves when using the shortest derivation compared to the most probable translation to select translations.

The scoring method introduced in the previous chapter indirectly addresses this issue by assigning a score to fragments which is independent of the frequency in which fragments with the same root pair were extracted. Although this is desirable, we feel that a model that explicitly considers the relations between lexical entries is beneficial and if possible should be exploited alongside an accuracy-based model such as that introduced in Section 3.3 (page 54). We find empirical motivation for this in our experiments in Section 3.4.7 (page 80), where the translation quality of a PB-SMT system decreases when phrase translation probabilities are completely replaced by accuracy-based scores.

When attempting to introduce new scoring sources into DOT, we face the problem that this model does not easily allow the introduction of additional scores. In the previous chapter we introduced accuracy-based scores into DOT by reformulating the model as log-linear, but we did so at the fragment level. Therefore, we remain unable to exploit non-local features to the fragments —such as n-gram language models– or features which are not possible to be encoded in the grammar, such as features of the particular sentence we are translating.

In this chapter we address these issues by proposing a new DOT model whose scoring is driven by a log-linear combination of features. Given a translation search space constrained by the original DOT model, our new model is able to exploit features of the complete source sentence while maintaining the advantage of exploiting DOT's strong structural and reordering models. This new model represents a framework in which different features can be explored, of which we take advantage by introducing a bilingual lexical feature and an n-gram-based language model. Our lexical feature conditions the choice of a target word to the source words it is linked to, as well as the source context in which these source words occur. We investigate different estimation methods for this feature (namely MaxEnt and interpolated smoothing) and report on their empirical performance, as well as on the impact of source-context information when incorporated into the LM-augmented system.

4.2 Log-Linear Data-Oriented Translation

As previously mentioned, DOT's scoring model is heavily driven by derivation structure rather than by lexical choice. To alleviate this, we take a novel log-linear formulation of DOT, in which we introduce features of each target word in a fragment pair, given the source sentence and the fragment pair. In our new model we assign a score to a target-language sentence given the source as in (4.3):

$$P(t|s) = \exp(\sum_{i=1}^{N} \lambda_i h_i(t,s)) \tag{4.3}$$

where each h_i is a feature function and each λ_i its corresponding weight. The feature $h_{\mathsf{rf}}(s,t) = \log(\sum_{\mathbf{d}_{\langle s,t\rangle}} P(\mathbf{d}_{\langle s,t\rangle}))$ accounts for the original relative-frequency probability in DOT (Section 2.3 in Chapter 2).

To define our lexical features, we need to associate a fragment in a derivation with the source sentence span that it covers. Given a derivation $\mathbf{d} = \langle d_f, d_e \rangle_1 \circ \ldots \circ \langle d_f, d_e \rangle_M$, let $\mathsf{span}(\langle d_f, d_e \rangle_i, \mathbf{d})$ be a function which maps a fragment pair $\langle d_f, d_e \rangle_i$ belonging to a derivation to a pair of numbers indicating the starting position in the source sentence of the span covered by the source fragment d_f, and the length (i.e. amount of words) of this span.

Our source lexical features take the form of (4.4):

$$h_{\mathsf{sc}}(s,t) = \sum_{\mathbf{d}_{\langle s,t \rangle}} \prod_{\langle d_f, d_e \rangle \in \mathbf{d}_{\langle s,t \rangle}} \text{score}(\langle d_f, d_t \rangle, s, \mathsf{span}(\langle d_f, d_e \rangle_i, \mathbf{d}_{\langle s,t \rangle})) \tag{4.4}$$

where

$$\text{score}(\langle d_f, d_e \rangle, s, i, j) = \prod_{t_k \in lex(d_e)} p(t_k | \mathsf{sc}(d_f, s, i, j)) \tag{4.5}$$

and where $lex(d_e)$ denotes the list of words in the target fragment d_e, and sc denotes a source-context function which associates individual target words with the source-side context of the source fragment they are linked to (cf. Section 4.2.1 for a precise definition of this function).

For each target fragment in each derivation with source side s and target side t, equation (4.4) takes the product of the source context from each target word in the fragment. For example, when considering the fragment pair in Figure 4.1, we evaluate the source contexts of both t_i and t_{i+1}. This ensures that every target word is supported by source-side evidence. Although from the example in Figure 4.1 it might appear redundant to condition on each target word, in large fragments which include substitution sites this prevents the introduction of words not related to the source side of the fragment pair.

In principle any information from the source-language sentence words and its parse tree can become part of the context. In this work we focus on the lexical window features we define in the following section, although in the future many

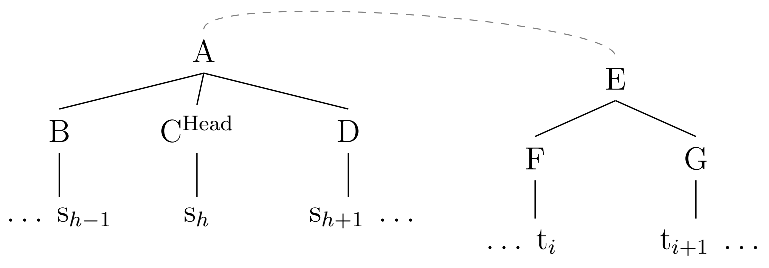


Figure 4.1: Lexical window feature

other sources of context information could be considered, such as head word of the sentence, recursiveness of the parse tree, POS tag windows (Stroppa et al., 2007) or even dependency information (Haque et al., 2009b).

4.2.1 Lexical-Window Feature

Given the source side of a fragment pair and the source-sentence span that this source fragment covers, we obtain a source context for a target word t_i by first determining s_h, the first source word dominated by the head[1] child of the source fragment. We then take a window of a predetermined size consisting of an ordered tuple of source-sentence words, introducing end-of-sentence or beginning-of-sentence symbols in cases where the window spans the sentence boundaries. If the window size is odd, windows are centred at s_h, otherwise we centre the window so that it includes an additional word to the right of the head.[2]

As these windows consist of a fixed amount of purely lexical items, we anticipate data-sparseness problems for large window sizes. We avoid this problem by also considering all possible sub-windows within the original window. The source contexts we consider thus consist of a list of tuples of words. The first tuple is the original window, of size n, followed by 2 tuples of size $n-1$, then 3 of size $n-2$, and so on. In total, if the window size is n, there are $\frac{n*(n+1)}{2}$ tuples in a source context.

For example, when translating the target word t_i using the fragments in Figure

[1]In our experiments translating from English, we use the head-finding rules defined by Collins (1999).

[2]In right-branching languages, on average we would expect words to the right of the head to be more related to it.

4.1, if the head child of A is C and we are using a window size of 3, the source context sc is composed of the following source word tuples: (s_{h-1}, s_h, s_{h+1}), (s_{h-1}, s_h), (s_h, s_{h+1}), (s_{h-1}), (s_h), (s_{h+1}).

Note that if we used a window of size 4 or more in this particular case, the context information would include words which lie outside of the span covered by A, the node linked to t_i. As we discuss in Section 4.6.1, this situation is frequent and beneficial.

4.2.2 Language Model

In addition to the previous feature, which acts locally on the target side, we also incorporate an n-gram language model (LM) which takes the complete target sentence into account. As standard DOT exclusively uses its grammar for scoring, this is a novel feature. Although it could be argued that no LM should be necessary given that the DOT system is syntax-based, in practice obtaining the effects of a LM using a DOT grammar would require this grammar to be highly lexicalised (by percolating words to the labels of fragment roots), increasing even more the complexity of the model. In addition, the beams used for pruning do not allow all of the possible combinations to be explored. We therefore believe that a LM will be beneficial to DOT even when large training data sizes are used, as it allows fluency to be maintained across the boundaries of fragments without the need for a highly lexicalized grammar.

Our current implementation computes this feature only when a complete target sentence has been generated, i.e. when translating the grammar's TOP symbol. During search, however, we multiply (as a heuristic) both the derivation score and the language model score of a particular fragment's target-side yield to prune derivations. For example, when incorporating the fragment pair in Figure 4.1 into a derivation, we consider the score assigned to the fragment pair by the model an the language model score of the target span of the fragment pair, i.e. the sequence of words t_i t_{i+1}. During decoding, a beam is maintained for each source substitution site (cf. Section 4.4), in which at most k translations are allowed. When more than k translations of a source substitution site are found, we consider this heuristic score and keep the highest-scoring translations, pruning the remaining ones.

In the future we expect to see further improvements by properly computing this feature's score in an incremental manner while building the translations. To this end a technique (such as cube pruning (Chiang, 2007)) which allows the efficient computation of language model scores when combining entries stored in a chart will have to be implemented.

4.3 Context Score Estimation

The scores in equation (4.5) rely on computing the conditional probability of a target word given a source context, which consists of a list of tuples of words of various sizes. Such a parameter space is prone to data sparseness, and a sound estimation method is therefore crucial for these features to prove useful.

We propose two alternative methods for estimating this conditional probability, namely MaxEnt (Berger et al., 1996) and improved Kneser-Ney smoothing (Chen and Goodman, 1998). We evaluate the performance of both and report on their strengths and weaknesses.

4.3.1 Maximum Entropy

MaxEnt is a framework which is particularly well-suited to tasks of this kind, and it has successfully been used in similar situations (Bangalore et al., 2007; He et al., 2008).

In this framework we directly estimate the distribution of $p(t|\mathsf{sc}(d_f, s, i, j))$ by using an exponential model as in (4.6):

$$p(t|\mathsf{sc}(d_f, s, i, j)) = \frac{1}{Z(\mathsf{sc}(d_f, s, i, j))} \exp\left[\sum_{k=1}^{n} \lambda_k f_k(\mathsf{sc}(d_f, s, i, j), t)\right] \tag{4.6}$$

where Z is a normalization constant to ensure that $\sum_t p(t|\mathsf{sc}(d_f, s, i, j)) = 1$, and each f_k is a binary feature indicating the presence or absence of a particular lexical tuple in the source context list sc, as in (4.7):

$$f_{lex\_window,t'}(\mathsf{sc}, t) = \begin{cases} 1 & \text{if } t = t' \text{ and } lex\_window \in \mathsf{sc} \\ 0 & \text{otherwise} \end{cases} \tag{4.7}$$

The feature weights λ_k are estimated using the *limited memory variable metric* (or L-BFGS) algorithm (Malouf, 2002).[3]

4.3.2 Interpolated Smoothing

Given that the windows that we use are composed purely of lexical items, there will be a huge number of features (in the order of millions) whose weights will need to be estimated under the MaxEnt approach. Even when using efficient algorithms like L-BFGS, the use of MaxEnt on such a highly lexicalised model could prove to be prohibitively slow (cf. Section 4.6.1 for more discussion on the efficiency of MaxEnt).

As an alternative, we propose to maximize the likelihood of the contexts occurring in the training data by using relative frequencies. In a similar setting, Gimpel and Smith (2008) avoid the data-sparseness problem of this method by introducing a different feature for each of the relative-frequency estimates of contexts of various sizes. Here we choose to directly interpolate the likelihood of increasingly larger context sizes. For every context-window list $\mathsf{sc}(d_f, s, i, j) = c_1, \ldots, c_n$ where every c_k is a tuple of source-language words of decreasing size, we estimate $p(t|c_1, \ldots, c_n)$ by considering the count of occurrences of the event $c_1, \ldots, c_n, t$ in the training data, and backing-off to $p(t|c_2, \ldots, c_n)$ when this count is below a certain threshold. We use improved Kneser-Ney (IKN) smoothing to perform this estimation, as implemented by the IRSTLM language modelling toolkit (Federico and Cettolo, 2007).

The features in the maximum entropy framework only test for membership of

[3]In our experiments we use the MaxEnt toolkit implemented in C++ by Le Zhang, available at http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.html

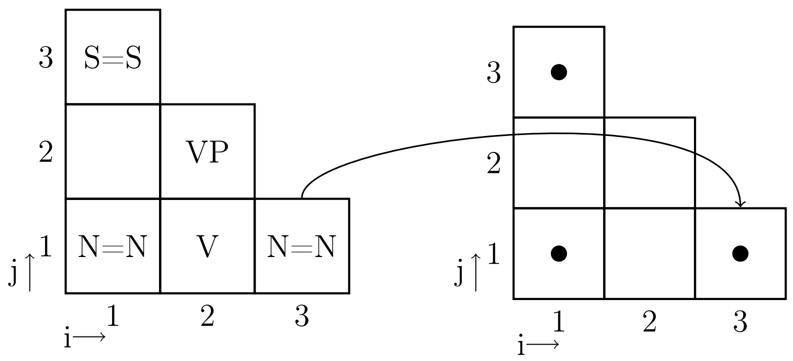


Figure 4.2: Obtaining a target-language chart from the substitution sites in a source-language chart.

a particular lexical tuple in the source context list, and the order in which these tuples appear in the list is, therefore, irrelevant. However, when using IKN as an estimation method, it is crucial to order contexts in such a way that when backing off, contexts which are both more frequent and more related to the target word t_i are kept, and more infrequent and less relevant contexts are discarded.

For example, if we were to use the order listed in Section 4.2.1 when extracting contexts for Figure 4.1, we would be left with the situation in which for larger window sizes, the smaller back-off model $p(t_i|c_n)$ would be conditioned on a word that is of dubious relation to t_i, as it might have appeared outside the span covered by the node A linked to t_i. Instead, we take an ordering that places first (and therefore discards first when backing-off) sub-windows whose centre is as far from the head word s_h as possible. When the centres of two sub-windows are the same distance from s_h, we choose to place windows which contain more words to the left of s_h first.[4] For the case discussed in Section 4.2.1, this would give the following ordering: (s_{h-1}, s_h, s_{h+1}), (s_{h-1}, s_h), (s_h, s_{h+1}), (s_{h-1}), (s_{h+1}), (s_h).

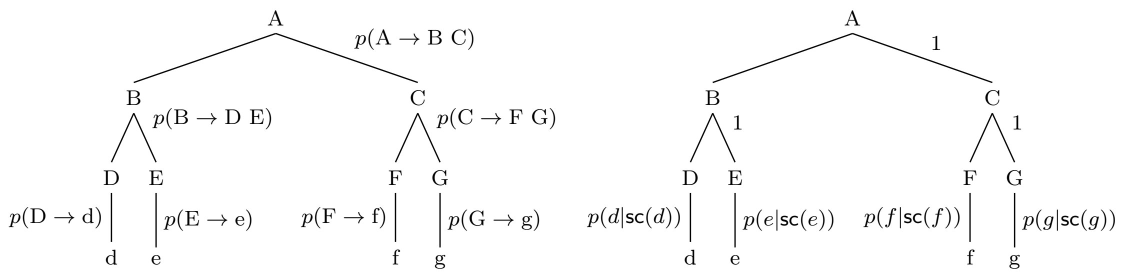
4.4 System Architecture

Since the scores used by our features depend on the particular source sentence we are translating, it is infeasible to encode them in a Goodman PCFG grammar. This poses a problem, since on the one hand we wish to introduce our new features, which is not allowed by the grammars, and on the other hand we do wish to continue exploiting the rich structural and reordering model that using these grammars provides. We solve this problem by taking the approach of using the original DOT model to bootstrap the new one.

Using the original DOT grammar, we parse the source sentence and compute the n-best source derivations, obtaining a forest of source parses which we efficiently store in a chart. Since we keep track of the alignment between source and target Goodman indexes, using this source chart we can create a target chart which, for each source substitution site, stores translations obtained with the target DOT grammar, as illustrated in Figure 4.2. Using this method we obtain a large collection of bilingual derivations. If we were using the original DOT model, we would assign to each of these the score indicated by the grammar, and determine the best translation by summing over derivations which yield the same target sentence. Instead, we take the source chart given by the source grammar and then proceed to build the target chart for each source substitution site bottom up, but using the model of equation (4.3) to perform the scoring instead of the one defined by equation (2.6) and encoded in the original grammars. In practice, in our implementation, whenever the grammar is queried for probabilities we replace the score of unary (and therefore preterminal to word) rules by the score of our new model, and replace the score of binary rules by 1, as illustrated in Figure 4.3. The standard translation procedure then takes care of propagating the lexical scores up in the tree when multiplying the scores of sub trees.

By doing this, we are using our new model over a translation search space con-

[4]This decision is again motivated by properties of right-branching languages, although we would not expect to observe a significant change in behaviour should we decide to discard words to the right first. This remains an avenue to pursue in future work.


A
$p(\text{A} \rightarrow \text{B C})$
B
$p(\text{B} \rightarrow \text{D E})$
C
$p(\text{C} \rightarrow \text{F G})$
D E F G
$p(\text{D} \rightarrow \text{d})$
$p(\text{E} \rightarrow \text{e})$
$p(\text{F} \rightarrow \text{f})$
$p(\text{G} \rightarrow \text{g})$
d e f g
1
$p(d|\mathsf{sc}(d))$
$p(e|\mathsf{sc}(e))$
$p(f|\mathsf{sc}(f))$
$p(g|\mathsf{sc}(g))$

(a) Original probability assignment (b) New probability assignment

Figure 4.3: Target side of a fragment pair, with labels indicating the probability assignment for each rule. The probability of the tree equals the product of the probabilities of all of its rules.

strained by the DOT grammars, benefiting both from our lexical features and the strong DOT structural model.

4.4.1 Efficient Decoding

Although the Goodman reduction produces grammars of size linear in the number of nodes in the treebank, these are still considerably large and their direct use on large training data sets is problematic, as using the standard implementations of parsing algorithms such as CYK (Younger, 1967) is in practice too time-consuming.

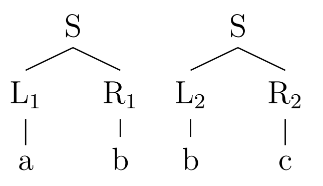
We take two measures to speed up decoding: multilevel coarse-to-fine parsing, and constituent constraining. We now describe these methods in detail.

Multilevel Coarse-to-Fine Parsing

Our source-side parser is implemented in a multilevel coarse-to-fine (CTF) fashion (Goodman, 1997; Charniak et al., 2006). We avoid directly parsing with the complete source Goodman grammar by defining a series of incrementally more coarse-grained grammars. Each label in one of the coarser-grained grammars is a projection of a finer-grained label. This means that if we are unable to obtain a parse tree for a sentence (or a constituent of a sentence) using one of the coarse grammars, we can be certain that it would not be possible to obtain a parse tree using the finest-grained grammar. The opposite of this is not true, as even if we are able to parse a sentence

$$\begin{aligned} S &\rightarrow L_1\ R_1 \\ S &\rightarrow L_2\ R_2 \\ L_1 &\rightarrow a \\ L_2 &\rightarrow b \\ R_1 &\rightarrow b \\ R_1 &\rightarrow c \end{aligned}$$

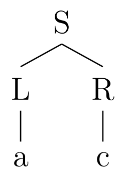
(a) Original grammar



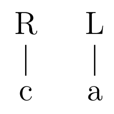
(b) The only two strings accepted by the grammar.

$$\begin{aligned} S &\rightarrow L\ R \\ L &\rightarrow a \mid b \\ R &\rightarrow b \mid c \end{aligned}$$

(c) Grammar obtained using a projection which removes any index in a label.



(d) The projected grammar accepts the string "a c", which was not accepted by the original grammar.



(e) The projected grammar correctly predicts that a complete parse will not be obtained for the string "c a".

Figure 4.4: Effects of projecting a grammar into a more coarse-grained one.

with a coarse-grained grammar, the new constraints imposed by the specialization in finer-grained labels could prevent us from obtaining a parse tree. However, we can take advantage of knowing which constituents will not be able to be parsed by the finest-grained grammar to speed up the parsing process. The effects of obtaining a more coarse-grained grammar can be observed in the example of Figure 4.4. Figure 4.4(a) gives the original grammar, which accepts only two strings, as shown in Figure 4.4(b). Although the projected grammar of Figure 4.4(c) accepts strings not accepted by the original (Figure 4.4(d)), it correctly predicts that using the original grammar a parse will not be obtained for the string "c a" (Figure 4.4(e)).

Given an original PCFG, we obtain a projection of it by applying to each of the labels the function π, which maps original labels into a new label from a reduced set of labels. We can also obtain a function π^{-1}, which maps a coarser-grained label into the set of all possible finer-grained labels that it could have been mapped from. We define a series of projection levels in a way such that the labels obtained in one level are the labels which the next projection maps, as illustrated by Figure 4.5.

We define three projection levels π_1, π_2 and π_3. The first level is the identity function, taking a label from the original Goodman grammar and returning the same

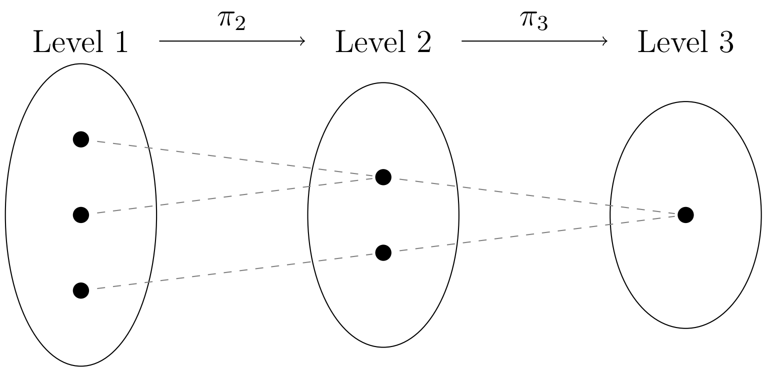


Figure 4.5: Sketch of the domain sets for the different projections. Dots represent labels which are mapped (by dashed lines) to the domain of the following projection.

label. The second-level projection takes a Goodman grammar label and removes any Goodman indexes from it, as illustrated in Figure 4.6 This projection is similar to the one used by Bansal and Klein (2010) for monolingual DOP parsing. The projection in the third level takes a label returned by the second-level projection. Its behaviour depends on whether the label is a preterminal or not. If the projected label (as obtained by π_2) was observed in a preterminal rule in the grammar obtained by π_2, the label is left intact, i.e. π_2 and π_3 share the same preterminal label sets. If the label is not a preterminal, then any target-grammar category after the "=" symbol is removed, as in Figure 4.6.

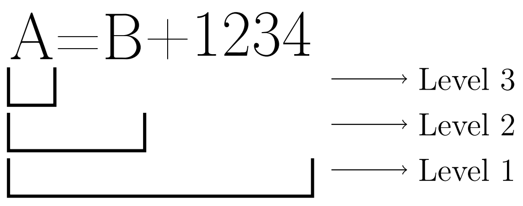


Figure 4.6: Transformations made by the different projection levels to an original Goodman grammar label.

To parse an input sentence with this series of grammars, the coarsest-grained grammar (i.e. the one obtained by π_3) is used to obtain an initial chart encoding all possible parses of the sentence (using a standard CYK algorithm). Once a chart is obtained with this simplified grammar, a second-level chart is obtained by using π_3^{-1} to map each symbol in the chart to the possible labels in the second-level grammar, and removing those labels which would not be allowed by binary rules in the second-

Method	Bleu	Time (seconds)	Backoffs
CYK	12.57	387,171	-
CTF	12.43	162,943	-
CTF+1best	11.80	3,655	14
CTF+2best	12.43	3,732	12
CTF+3best	12.40	4,058	8

Table 4.1: Translation time and Bleu scores for a 200-sentence development set using different source-language parsing algorithms: standard CYK, and multilevel coarse-to-fine (CTF) with and without constituent constraining. When using constituent constraining, the amount of sentences that required backing off to basic CTF is also presented.

level grammar. Finally, using π_2^{-1} a chart is obtained using this same process, leading to a chart encoding all possible parses with the original Goodman grammar. We do not prune when obtaining a chart given the previous-level chart and so in theory obtain the same results as if directly using the full grammar, although technical differences between the original parsing algorithm and the CTF algorithm lead to (minor) variations in the (translation) scores.

Table 4.1 gives results for an experiment where a small development test set consisting of 200 sentences was translated using the standard CYK algorithm (1st row) and the CTF method (2nd row). As can be seen, a 57.91% improvement in translation speed can be achieved (a 2.37 times speedup), with little loss in translation quality (the difference is not statistically significant). All times were measured on a 2.67GHz Xeon CPU.

Although the speed obtained with the CTF method is still relatively slow, we will show next how combining this method with constituent constraints leads to acceptable translation speeds. However, if desired, greater speed improvements could be achieved by the use of this method alone, if we prune constituents when obtaining a chart from a previous-level chart using their inside and outside probabilities (Goodman, 1997), or if we employ techniques such as beam and global thresholding (Goodman, 1997). While taking this approach could avoid using an external parser before translation (cf. next section), this would have the drawback that multiple threshold values would need to be estimated, as well as the possible alteration of translation scores as a result of the pruning.

Constituent Constraints

The second measure taken to improve decoding time is to use a state-of-the-art parser (Charniak and Johnson, 2005) in a preprocessing stage to obtain a parse tree for the input sentence. This parse tree is used by our decoder to constrain the constituents that are allowed to be considered: only constituents spanning portions of the source sentence which were deemed to be constituents by the monolingual parser are allowed to be created. Constituent labels are ignored in this constraint.

We translated our development test set using this method combined with the coarse-to-fine approach as previously described, and obtained the results in row 3 of Table 4.1. As can be observed, a significant speedup is achieved, although at the expense of translation quality: the difference with standard CYK is statistically significant at $p = 0.05$. The loss in translation quality is due to this constraint being too strict, and therefore not allowing alternative derivations which could yield better translations to be considered. In a bid to soften the restrictions on constituent creation, we consider using the aggregated constituents taken from the n-best trees returned by the parser. When we do this, we observe in the 4th row of Table 4.1 that using the aggregated constituents from the 2-best parse trees is enough to regain the lost translation quality, while maintaining the improvements in speed. The last column in Table 4.1 gives the amount of sentences for which we detected (by counting the empty cells in the chart) that the constraining was overly strict, forcing us to back off to using CTF parsing and lifting the constituent constraints. Since this is something that could potentially bring parsing times up which is clearly something we would like to avoid, we stay on the safe side by using the 3-best parse trees to constrain the parsing. Using this setup (which is the one employed in our experiments in Section 4.6), we obtain a 98.95% improvement in total decoding time, corresponding to an impressive 46.12 times speedup. The loss in translation quality compared to standard CYK is not statistically significant (as determined by paired bootstrap resampling (Koehn, 2004)).

4.5 Experimental Setup

We evaluate the performance of our new model by translating English sentences taken from the Europarl corpus (Koehn, 2005) into Spanish, and comparing performance against a baseline DOT system. We created a parallel treebank for training in a similar way to that in the previous chapter, although we parsed the source side of the parallel corpus using Charniak and Johnson's (2005) parser, the same parser used to obtain the 3-best trees for our test set, used for the previously mentioned constituent constraints (cf. Table 4.1). The target side of the parallel corpus was trained using the same parser used for our experiments in Section 3.3.5, i.e. Bikel's parser with a Spanish language pack (Chrupała and van Genabith, 2006). As noted in Section 3.3.5, this parser produces POS tags which are too fine-grained for our purposes. The set of tags we use is as described in the aforementioned section, and as listed in Appendix A. The parallel treebank was then created using the same tree-to-tree aligner (Tinsley et al., 2007a) used in our previous experiments.

We extract the DOT grammar from 80K randomly selected sentences. This is an increase of almost an order of magnitude compared to our experiments in the previous chapter, which in turn represent an order of magnitude increase compared to previously published results (Hearne and Way, 2003, 2006). We also use the IRSTLM toolkit (Federico and Cettolo, 2007) to train a 5-gram language model on 200K randomly selected sentences which include the target side of the corpus section used to extract the grammar.

Our development test set and our test set contain 200 and 2K randomly selected sentences respectively, and we use an additional 200 sentences to tune the weights in equation (4.3) using Minimum Error Rate Training (Och, 2003), as implemented by Z-MERT (Zaidan, 2009). The length of the sentences in these sets (and on the corpus used to extract our grammars) does not exceed 20 words.

In the experiments in this chapter we translate by first obtaining the 200-best parse trees of the input sentence and then obtaining the 50-best target trees for each source substitution site. From the resulting collection of bilingual derivations, we approximate the most probable translation by summing over all derivations which yield the same target translation.

As in previous experiments statistical significance is tested by paired bootstrap resampling (Koehn, 2004), and absolute scores for Bleu and the Meteor are reported as percentages.

4.6 Experimental Results

In this section we evaluate the translation quality of our new model compared to a baseline DOT system based on the relative frequency estimator.

We begin by determining the best window size for our source lexical windows, and by determining which estimation method is more suitable for obtaining scores for the source lexical contexts. This is followed by experiments in which our best-performing lexical window setup is used alongside the language model feature.

4.6.1 Lexical Window Size

For the purposes of evaluating the impact of the source context window features of equation (4.4), we consider a system whose only feature is h_{sc}, effectively replacing the original DOT probabilities.

Experimental results for Bleu on our development test set are given in Figure 4.7. The baseline system obtains a Bleu score of 12.25. In our experiments in Section 3.3.6 we noted that a similar DOT system was comparable to a phrase-based system which uses no language model and uniform feature weights. Although translating in the opposite direction and training our systems with almost an order of magnitude larger amount of data, we have found this to be the case here also, with

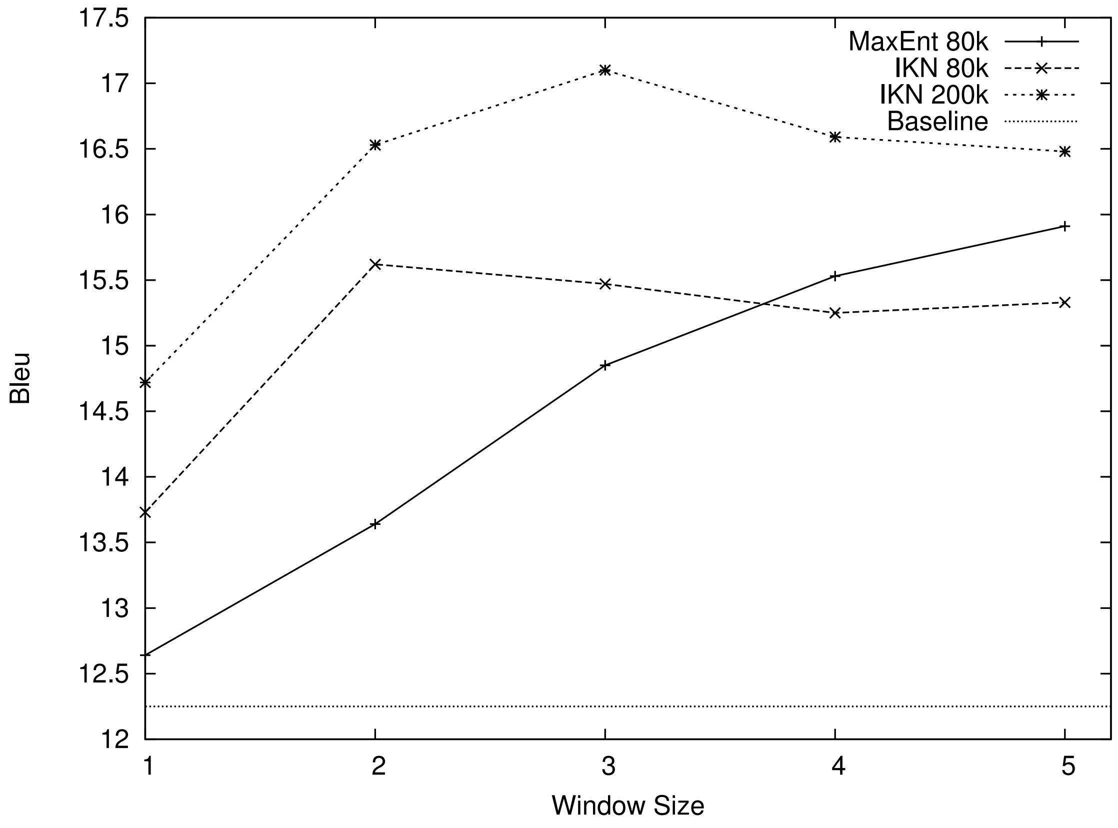


Figure 4.7: Window size vs. Bleu (development test set)

the phrase-based system obtaining a statistically insignificantly worse score: 11.57 Bleu. We should note that although in the present chapter we incorporate a language model into DOT and estimate feature weights via MERT, this still represents preliminary work, as our language model feature is not computed incrementally, and our log-linear model exploits only one additional feature. A proper comparison with PB-SMT would require the maturing of these features and the further optimization of our system, as well as further research orthogonal to that carried out in this thesis such as finding the optimal grammar tag-set or treebank binarization. However, as a guide to the reader we note that when the PB-SMT system is allowed to exploit a language model and to tune its feature weights using MERT, it obtains a Bleu score of 23.47.

System	**Bleu**	**Nist**	**Meteor**
Baseline	12.25	3.5482	38.30
MaxEnt Window Size 1	12.64	$\underline{3.9985}$	39.70
MaxEnt Window Size 2	*13.64*	$\underline{4.1750}$	*40.80*
MaxEnt Window Size 3	$\underline{14.85}$	$\underline{4.2062}$	$\underline{41.06}$
MaxEnt Window Size 4	$\underline{15.53}$	$\underline{4.3303}$	$\underline{41.50}$
MaxEnt Window Size 5	$\underline{15.91}$	$\underline{4.3536}$	$\underline{42.13}$

Table 4.2: Results on development test set using Maximum Entropy models trained on 80K sentence pairs with different window sizes. Underlined results and results in italics are statistically significantly better than the baseline at $p < 0.01$ and $p < 0.05$ respectively.

Maximum Entropy Model

When considering the system with source context windows estimated using MaxEnt over 80K sentence pairs, it can be observed in Figure 4.7 that increasing window sizes has a direct impact on translation quality, with Bleu increasing along with window size, and the best system having a score of 15.91 (a 3.66 Bleu points increase, 29.87% relative). Except for the system with window size 1, whose improvement is statistically insignificant, and the system with window size 2 which brings a statistically significant improvement at $p = 0.05$, all other results are statistically significantly better than the baseline at $p = 0.01$. The remaining evaluation metrics (shown in Table 4.2) present a similar pattern.

Although the improvement curve is decreasingly steep, it appears that we could achieve even better results by continuing to increase the window size, or by training these models using additional training data. Nevertheless, the training time for the MaxEnt system with window size 5 is over 33 hours.[5] Increasing window or training data sizes beyond this point brings training times to the order of days, or even weeks. If we consider that in the future we will continue optimizing our systems so that training corpora of larger sizes can be used, a more efficient alternative for the estimation of this feature would be desirable. We therefore investigate in the following section how the improved Kneser-Ney models, which can be estimated

[5]All times in this section were measured on a 2.93GHz Xeon CPU.

System	**Bleu**	**Nist**	**Meteor**
Baseline	12.25	3.5482	38.30
IKN 80K Window Size 1	*13.73*	4.1633	40.73
IKN 80K Window Size 2	15.62	4.3332	41.75
IKN 80K Window Size 3	15.47	4.3430	41.90
IKN 80K Window Size 4	15.25	4.3075	41.41
IKN 80K Window Size 5	15.33	4.3146	41.63

Table 4.3: Results on development test set using improved Kneser-Ney models trained on 80K sentence pairs with different window sizes. Underlined results and results in italics are statistically significantly better than the baseline at $p < 0.01$ and $p < 0.05$ respectively.

significantly faster, compare to the MaxEnt models.

Improved Kneser-Ney

Figure 4.7 also shows results for the system which uses improved Kneser-Ney (IKN) smoothing for context score estimation over the same 80K tree pairs. Although at a window size of 2 it gives a statistically significant improvement over the corresponding MaxEnt system, increasing window sizes has a negative effect on translation quality, indicating that data-sparseness might be a problem.

However, the estimation method used by this model is significantly simpler than MaxEnt and it can thus be trained significantly faster, scaling more gracefully to larger training data sizes. We exploit this fact by making use of 200K tree pairs from our parallel treebank to train this model (our grammars extracted from 80K parallel trees remain unchanged). The results in Figure 4.7 show that although IKN does not scale as smoothly as the MaxEnt model when increasing window sizes, the additional training data is enough for it to outperform both the baseline and the MaxEnt system. The best result is obtained at a window size of 3, with a Bleu score of 17.1 which is statistically significantly better than the MaxEnt model with window size 5 ($p < 0.03$). The difference with the baseline system is of 4.85 absolute Bleu points, corresponding to a 39.59% relative improvement.

We note that the toolkit used to estimate our feature (Federico and Cettolo,

System	**Bleu**	**Nist**	**Meteor**
Baseline	12.25	3.5482	38.30
IKN 200K Window Size 1	14.72	4.2184	*41.19*
IKN 200K Window Size 2	16.53	4.4373	42.43
IKN 200K Window Size 3	17.10	4.5449	43.31
IKN 200K Window Size 4	16.59	4.5450	43.36
IKN 200K Window Size 5	16.48	4.5210	43.05

Table 4.4: Results on development test set using improved Kneser-Ney models trained on 200K sentence pairs with different window sizes. Underlined results and results in italics are statistically significantly better than the baseline at $p < 0.01$ and $p < 0.05$ respectively.

2007) is able to split the training data so that training can be performed in parallel, allowing larger amounts of training data to be used as more CPUs are available. Furthermore, we note that this model was trained in under 2 hours.

Tables 4.3 and 4.4 give the results obtained by the IKN systems using our remaining evaluation metrics. NIST and Meteor show a behaviour similar to the one presented by Bleu, with the exception that the scores obtained with window sizes 3 and 4 show little difference (particularly when training on 200K sentence pairs).

Alignment-Constrained Windows

When taking the source context of a target word t_i, we note that the median size of the span covered by the corresponding source-side node (e.g. node A in Figure 4.1) is 1 word. This means that when taking window sizes of 2 or more, on average we are including not only the words that t_i is aligned to, but also the lexical context in which they appear. This is reminiscent of the use of context-informed features in phrase-based MT, e.g. (Stroppa et al., 2007; Haque et al., 2009a).

To isolate the benefit of exploiting source-context information in our windows, we experimented with limiting windows to only include words dominated by the fragment-root node we are currently considering. That is, whenever a window would include words outside of the span covered by the fragment-root node, we ignored those words lying outside of the span. For example, if considering a window of

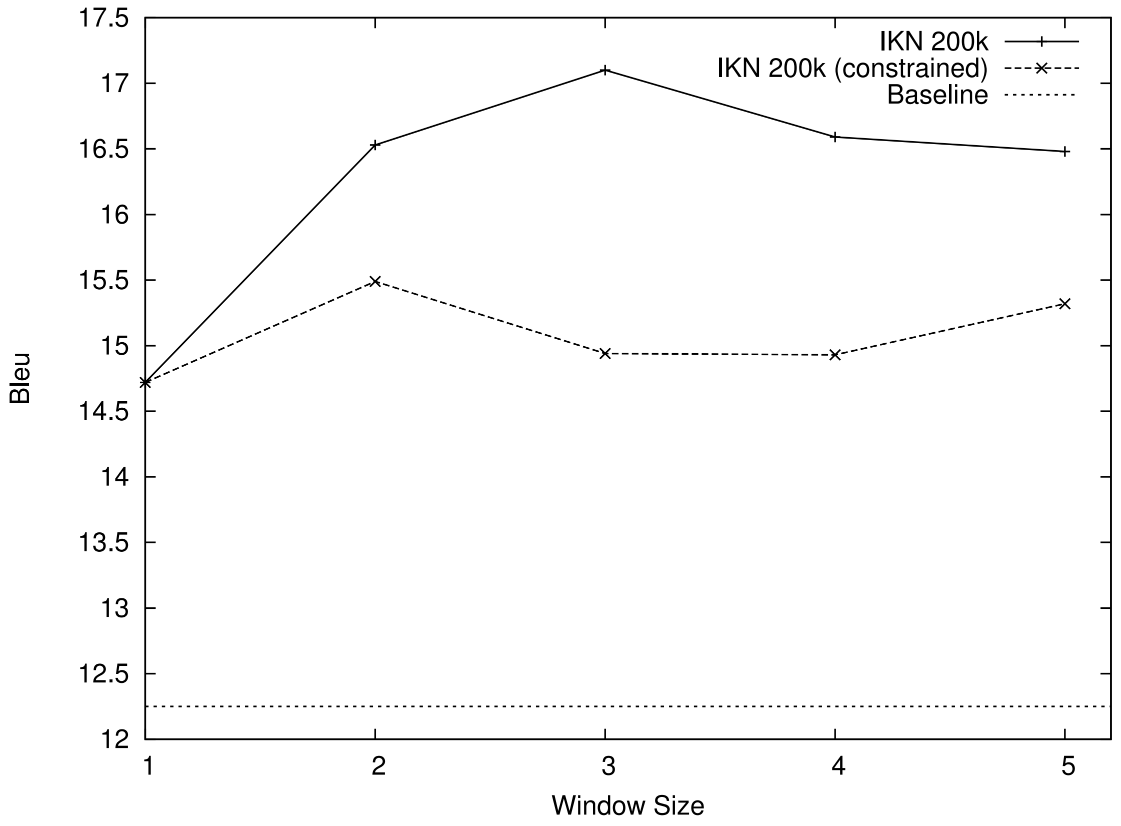


Figure 4.8: Impact on Bleu (development test set) of constraining lexical windows to the aligned span

size 4 for the example in Figure 4.1, the window would be limited to the words (s_{h-1}, s_h, s_{h+1}).

Results are given in Figure 4.8. We used our best-performing models for these experiments, i.e. the ones estimated using improved Kneser-Ney over 200K sentence pairs. While the constrained system consistently outperforms the baseline, it also consistently underperforms compared to the unconstrained one, indicating that the inclusion of source-context information is beneficial and can lead to significant gains in translation quality. The statistical significance of the differences in Bleu is $p < 0.04$ for the difference between the improved systems at window sizes 2 and 5, and $p < 0.01$ for all other differences. Table 4.5 gives results for the remaining evaluation metrics, from which we arrive at the same conclusions: the constrained windows significantly outperform the baseline, but significantly underperform compared to the unconstrained ones. Note that although this result shows that scoring fragments by taking into account their context is beneficial, this

System	Bleu	Nist	Meteor
Baseline	12.25	3.5482	38.30
IKN 200K Window Size 1 (constrained)	<u>14.72</u>	<u>4.2184</u>	*41.19*
IKN 200K Window Size 2 (constrained)	<u>15.49</u>	<u>4.2294</u>	<u>41.20</u>
IKN 200K Window Size 3 (constrained)	<u>14.94</u>	<u>4.2342</u>	*41.16*
IKN 200K Window Size 4 (constrained)	<u>14.93</u>	<u>4.2156</u>	*40.89*
IKN 200K Window Size 5 (constrained)	<u>15.32</u>	<u>4.2551</u>	*41.16*

Table 4.5: Results on development test set constraining windows to only include words aligned to the current target word, and using improved Kneser-Ney on 200K sentence pairs for estimation. Underlined results and results in italics are statistically significantly better than the baseline at $p < 0.01$ and $p < 0.05$ respectively.

does not give sufficient evidence to conclude that limiting translation rules to satisfy syntactic constrains is detrimental. However, this does suggest that incorporating translation rules which directly capture a context which spans the linguistic notion of a constituent could be desirable, as suggested in some of the syntax-based MT literature, e.g. (Marton and Resnik, 2008; Chiang, 2010).

4.6.2 Log-linear Model

Having explored different estimation methods and sizes for the lexical context windows, in this section we integrate them with our remaining features, estimating the λ weights required in equation (4.3) using MERT.

In particular we are interested in whether the source-context window feature is able to give an improvement over the stronger baseline of the system which uses the h_{rf} and the h_{LM} features. However, as a sanity check we first investigate whether the improvement we observed in the previous section when using h_{IKN} to translate our development test set carries over to the case of translating the larger test set. Table 4.6 gives experimental results using different combinations of features. For these experiments we used the context-window setups which led to the best results when translating the development test set, namely the system with a window size of 5 estimated using MaxEnt over 80K tree pairs (denoted h_{ME} in Table 4.6), and the one estimated using improved Kneser-Ney over windows of size 3 from 200K tree

Features	**Bleu**	**Nist**	**Meteor**
h_{rf}	12.44	3.9407	37.21
h_{ME}	$\underline{14.72}$	$\underline{4.7889}$	$\underline{40.10}$
h_{IKN}	*15.11*	$\underline{4.8996}$	$\underline{40.90}$
$h_{\mathsf{rf}} + h_{\mathsf{LM}}$	$\underline{16.70}$	4.6484	39.53
$h_{\mathsf{rf}} + h_{\mathsf{LM}} + h_{\mathsf{IKN}}$	$\underline{17.27}$	$\underline{4.9658}$	$\underline{41.25}$

Table 4.6: Results on test set. Underlined results and results in italics are statistically significantly better than the ones above them at $p < 0.01$ and $p < 0.03$ respectively.

pairs (h_{IKN}). Note that, unlike in the preceding tables, underlined and italic results in Table 4.6 denote statistical significance compared to all of the preceding rows and not only to the baseline. As can be observed, using h_{ME} instead of the relative-frequency baseline (h_{rf}) gives a significant improvement across all evaluation metrics. The difference in Bleu is 2.28 absolute points (18.32% relative). However, as expected, using h_{IKN} brings further improvements, which are statistically significantly better than the results obtained with h_{ME}. In this case the difference with h_{rf} is of 2.67 Bleu points, corresponding to a 21.46% relative improvement.

A language model is an important component in MT which has proved to be useful in a wide range of different MT engines. It is no surprise then that combining the h_{rf} feature with an LM ($h_{\mathsf{rf}} + h_{LM}$) leads to a dramatic increase in translation quality compared to using only h_{rf}. Interestingly, although this system improves in terms of Bleu compared to h_{IKN}, it falls short in our remaining evaluation metrics. Given this discrepancy, we set aside these two systems and perform a pairwise comparison between them. We use the smoothed sentence-level Bleu metric defined in Section 3.4.1 (page 75) to determine the amount of sentences in which one of these systems is better, and the amount of sentences in which both systems obtain the same sentence-level score. From these subsets of sentences we also obtain the average of the sentence-level scores. Results for this evaluation are given in Table 4.7. As can be observed, the amount of sentences in which h_{IKN} receives a better score is far larger than the amount of sentences in which the opposite is the case. This is

Metric	$h_{\mathsf{rf}} + h_{\mathsf{LM}}$ **better**	**Equal**	h_{IKN} **better**
Sentences	780	349	871
$h_{\mathsf{rf}} + h_{\mathsf{LM}}$ avg. BLEU	21.03	18.69	10.65
h_{IKN} avg. BLEU	11.43	18.69	17.06

Table 4.7: Sentence-level evaluation using smoothed BLEU to determine the sentences in which $h_{\mathsf{rf}} + h_{\mathsf{LM}}$ or h_{IKN} is better.

consistent with the document-level scores assigned by the metrics NIST and Meteor in Table 4.6. However, the BLEU score received by $h_{\mathsf{rf}} + h_{LM}$ over the (smaller) subset of sentences in which this system is better is larger than the corresponding score assigned to h_{IKN} when that is the best system. These differences in scores are probably what leads to an overall better BLEU score for $h_{\mathsf{rf}} + h_{LM}$. The average sentence-level score over the entire test set is 15.62 for $h_{\mathsf{rf}} + h_{LM}$ and 14.37 for h_{IKN}, which is consistent with the document-level scores in Table 4.6.

Finally, we used our best-performing context-window setup along with the LM and the h_{rf} feature. The result is a system which benefits from each of these components, improving over any single feature across all our evaluation metrics. Compared to $h_{\mathsf{rf}} + h_{LM}$, this system achieves a difference of 0.57 BLEU points (3.41% relative), while the difference with h_{rf} is 4.83 BLEU points (38.82% relative).

Our training corpora are of a relatively small size, which leads to baseline systems which obtain rather low absolute translation quality scores. The question arises then as to whether the improvements we have observed when incorporating a language model and source-context features would be maintained if training our systems using larger training data sizes. The DOT system extracts a large amount of fragments whose size is not restricted. This means that this system is prone to data sparseness problems, and that increasing training data sizes should see the absolute scores of the baseline system increase. However, when also increasing the training data sizes of our context windows and of the language model (which, if desired, can be trained using a larger amount of data than the one used to extract the grammars) we would expect the relative differences between the systems which we have observed to be maintained, and the conclusions to which we arrived to hold. The reason for this is that the behaviour observed in Figure 4.7 for the IKN system, where translation quality starts to decrease when increasing the window size beyond 2 (when training with 80K sentence pairs), and beyond 3 when training with 200K sentence pairs, can also be attributed to data sparseness problems. The maximum entropy model treats each lexical window as a separate feature, and is therefore less prone to data sparseness problems, as it can assign a low feature weight to windows for which no enough data has been obtained. This results in the smooth improvement curve observed in Figure 4.7. In contrast, the IKN system needs to estimate scores for each *list* of context windows, which results in large amounts of training data to be needed. We would expect therefore that as training data increases, the behaviour of the IKN model would approach that of the MaxEnt one, with translation quality improving as window sizes do. As regards the language model feature, translation quality is known to continue improving as the size of the LM training data does, even for very large training data sizes (Brants et al., 2007), which would indicate that we should continue to observe an improvement in translation quality as we increase both our system and our LM training data sizes.

4.7 Discussion

We give in Figure 4.9 example translations from the relative frequency-based system (h_{rf}) and the system whose only feature consists of the source-context windows estimated using improved Knesser-Ney over 200K sentences (h_{IKN}). h_{rf} produces a poor translation: the verb "condemn" has not been translated, and there are spurious words which make it hardly understandable. In contrast, h_{IKN} produces a sentence which is both fluent and an acceptable translation of the source (it reads: "we must condemn the human rights violations in burma"):

To understand the reasons for the differences in performance, we examine in Figure 4.10 the best derivation yielding the corresponding translation from each

Source: we must continue to condemn human rights abuses in burma
Reference: debemos continuar condenando los abusos contra los derechos humanos en birmania

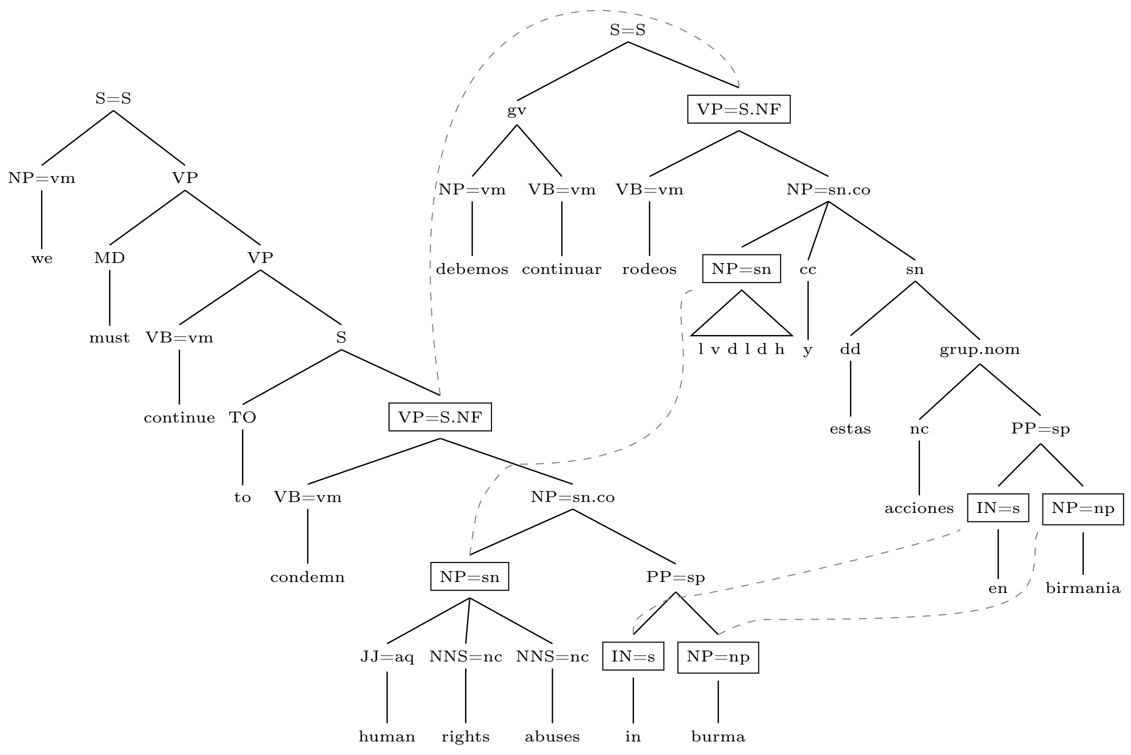
h_{rf}**:** debemos continuar rodeos las violaciones de los derechos humanos y estas acciones en birmania
h_{IKN}**:** debemos condenar las violaciones de los derechos humanos en birmania

Figure 4.9: Output translations from our systems

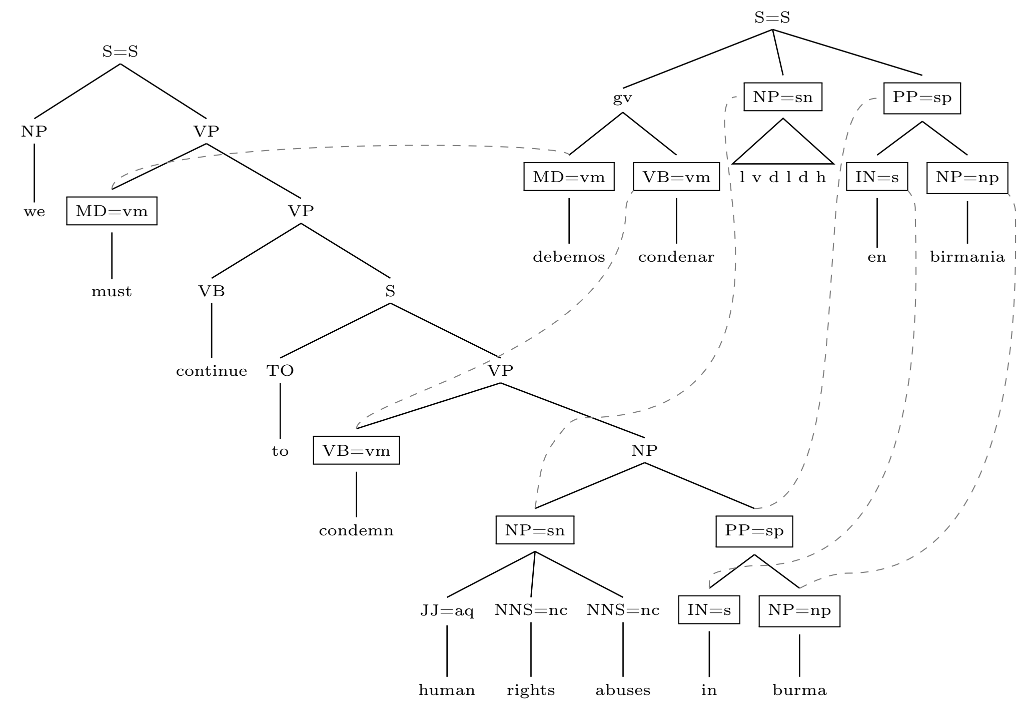
system. In the baseline system, the first derivation obtains the highest score because it uses few fragments whose roots are fairly frequent, obtaining a log-score of -44.91 under the Goodman PCFG model, while the second derivation obtains a lower score of -51.43. The reason why the second derivation is chosen in h_{IKN} is because under this model, every target word must not only fit in the structure imposed by the grammar, but it must also be evaluated against the source words it is aligned to. Every one of the target words the second derivation generates is highly related to the source context it is aligned to. On the other hand, the first derivation introduces many target words (under the node rooted by NP=sn.co) which have no source-side counterpart. While this goes unpenalized in h_{rf} , it is poorly scored by h_{IKN}, which assigns a log-score of -57.15 to the whole derivation. The corresponding score for the second derivation is a much higher -28.92.

Note that in this example, although the translation produced by h_{IKN} is of good quality, the word "continue" in the source sentence has not been translated. This is due to the conditioning on each target word, which will tend to generate fewer words in the translations as each target word must be supported by source-side evidence. Although our experiments show that this is not an impediment to achieving translation quality gains, in the future this effect could be lessened by the inclusion of features such as word or fragment counts of the translations.

We have observed that the method of relating target words to the *head* of source nodes frequently leads to situations such as the one in Figure 4.10, where top-level rules define the overall sentence structure, and substitution sites lower in the tree are responsible for the translations. We believe this to be a good thing: lexical



(a) Best-scoring derivation for the translation with highest probability under the baseline model



(b) Best-scoring derivation for the translation which obtained the highest probability under the model with lexical windows

Figure 4.10: Best scoring derivations from the h_{rf} system (a) and the h_{lKN} (b). “l v d l d h” stands for “las violaciones de los derechos humanos”. Boxed nodes denote substitution sites.

decisions are made based on related source-side segments, and reordering is handled by higher-level rules which might be head-lexicalized.

4.8 Conclusions

We have introduced a framework for DOT in which source information can be exploited to score translations over a search space constrained by the baseline model. The empirical results show that when introducing lexical context windows as source information, by conditioning every target word to the context in which its source-side equivalences occur, significant translation-quality improvements can be achieved. In addition, the log-linear combination of all features outperforms any individual feature.

Our experiments with MaxEnt and IKN as estimation methods show that the former seems to scale more gracefully with context size. Although its asymptotic properties might be better, in practice the efficiency and scalability of IKN makes it a viable substitution, achieving our best results.

In addition to our improvements in translation quality, we have described how efficient decoding can be achieved without any significant loss in translation quality. This has enabled us to significantly increase the size of the training data used in our experiments.

4.9 Summary

In this chapter we introduced a framework which allows DOT to exploit features other than the relative frequency of its fragment pairs. We used this framework to explore the effects of features which condition each target word to the source context they are associated with. We explored different estimation methods for these features, obtaining significant gains in translation quality with all of them. In addition to these source features, we also showed that DOT can benefit from the inclusion of an n-gram language model, and that the combination of all of these features leads to the best translation quality results.

We were also able to scale the size of our training corpora by defining methods which led to faster decoding times while remaining accurate.

Chapter 5

Conclusions

In this thesis, we have identified a number of aspects in which the state-of-the-art in MT could be improved. We began by noticing that PB-SMT is intrinsically uninformed as regards the syntax and the structure of both source and target sentences, and explained why a more syntax-aware system would be desirable. We suggested that DOT has the potential to overcome these limitations, by explicitly modelling the relationships between the linguistic structure of the source and target sentences. However, we also noted a number of limitations in DOT which prevented a fair comparison with PB-SMT from being made. When it comes to scoring translations, at the heart of these limitations is DOT's exclusive reliance on the frequency in which its translation rules were observed, lacking features which can model the relationship between source and target words, and a strong language model. Additionally, we also noted that both systems presented a mismatch between the criteria used to estimate their features and the ones finally used to evaluate them. We summarized this comparison in Table 1.1, which we repeat in Table 5.1 for convenience.

These observations gave rise to our research questions, which we now revisit:

RQ1 Can features which relate to expected translation quality be incorporated in MT models?

RQ2 Can these translation accuracy-based features improve on state-of-the-art MT?

Criterion	PB-SMT	DOT
Linguistic motivation	✗	✓
Long-distance reorderings	✗	✓
Accuracy-based scoring	∼	✗
Multiple features	✓	✗
Lexical equivalences Model	✓	✗

Table 5.1: Summary of the capabilities of PB-SMT and DOT, according to different criteria

RQ3 Can we incorporate new features into the DOT model of translation in such a way that the contribution of each feature can be scaled so as to optimise translation quality?

RQ4 Can we exploit this combination of features by incorporating new ones which lead to increased translation quality?

RQ1 was addressed in Chapter 3. We proposed a training algorithm which related the translation units of an MT system to the average impact in translation quality when they are used. This impact is measured as the average distance between a translation unit and a unit which was observed translating the same source span in a candidate sentence which maximized a translation quality metric. We successfully adapted this algorithm for DOT and for PB-SMT, obtaining significant translation quality improvements, and showing that it is indeed possible to integrate translation quality-oriented features into an MT model. Our empirical evaluation of the performance of this algorithm was carried out by performing Spanish-to-English translation and comparing the performance of the rescored system with the baseline system. This evaluation was carried out both at the document level, by using standard translation quality metrics, and at the sentence level by counting the amount of sentences for which each system showed improvements or decreases in a translation quality metric.

The experiments with PB-SMT in Chapter 3 address **RQ2** by adapting our accuracy-based scoring algorithm to this system. By doing this we obtained a system which significantly outperformed a state-of-the-art PB-SMT system over a range of evaluation metrics. Our experiments give significant evidence to support an affirmative answer to **RQ2**. Our work regarding **RQ1** and **RQ2** allowed us to conclude the following:

- Significant improvements in translation quality can be achieved by using accuracy-based scoring.
- The incorporation of accuracy-based scores into DOT by means of evenly spreading a fragment's score over the PCFG rules it is composed of, instead of assigning scores to fragments individually, is not an impediment to achieving translation quality gains.
- The use of larger corpora for the estimation of our feature leads to significant improvements compared to using smaller corpora. The improvement appears to be caused by more reliable estimates, rather than by a larger percentage of translation units receiving an accuracy-based score.
- Taking the internal structure of DOT fragment pairs into account in the accuracy-based scores computation is beneficial on the datasets in which our evaluation was performed.
- By means of deleted estimation, accuracy-based scores can be obtained for a PB-SMT system using only the corpus that was used to train the baseline system, requiring no additional parallel data.
- The use of the F-measure as oracle selection metric leads to better results than the use of sentence-level BLEU (sBLEU), although the latter is a viable option which also leads to translation quality gains.
- Our algorithm leads to the identification of a subset of phrase pairs which is significantly smaller than the complete set of phrase pairs in the baseline system (87% smaller in our experiments), but which is sufficient to obtain

translation quality scores on a par with those obtained using the complete phrase table.

In our description of DOT in Chapter 2 we explained how its method of scoring exclusively relies on the relative frequency of fragment pairs. We alleviated this situation in Chapter 4, where we developed a new DOT model which uses a log-linear combination of features to score translations, directly addressing **RQ3**. **RQ4** was also addressed in this chapter, by taking advantage of this model to incorporate new features into this model, namely our lexical-window feature and a language model. This resulted in a system which significantly outperformed the baseline DOT system. Our conclusions as regards **RQ3** and **RQ4** are as follows:

- The architecture of a DOT system which is implemented as described in Chapter 2 allows the incorporation of additional features by using the baseline system to create a search space over which the new model is employed.

- The incorporation of lexical windows in the log-linear model significantly improves the translation quality of the system, and the inclusion of source context in these windows is beneficial.

- The use of n-gram language models in DOT is certainly beneficial. This is the case even if, rather than performing an incremental computation of this feature during translation, their use is limited to the scoring of complete sentences.

- The estimation of the weights for the different features can be performed using off-the-shelf implementations of algorithms such as MERT. The log-linear combination of features outperforms any individual feature.

- MaxEnt presents better asymptotic behaviour than improved Kneser-Ney (IKN), resulting in a smooth improvement curve. However, if the time taken to train the models is an issue, and additional training data is available, IKN is a viable alternative, obtaining improved translation quality results as the amount of training data increases.

Criterion	PB-SMT	DOT
Linguistic motivation	✗	✓
Long-distance reorderings	✗	✓
Accuracy-based scoring	✓	✓
Multiple features	✓	✓
Lexical equivalences model	✓	✓

Table 5.2: Updated summary of the capabilities of PB-SMT and DOT, after taking into account the work carried out in this thesis.

- Efficient decoding can be achieved with DOT by the use of coarse-to-fine parsing and by constraining constituent creation to the constituents found by a monolingual parser. These efficiency improvements, which are of two orders of magnitude compared to the original system, enable us to scale up the training data sizes used to obtain DOT's grammars.

Having addressed some of the issues summarized in Table 5.1, we can now update this table to reflect the current status of both DOT and PB-SMT after the work carried out in this thesis. The updated summary is given in Table 5.2. Compared to Table 5.1, the changes are as follows:

- PB-SMT now has a full check mark on Accuracy-based scoring. The previous half check mark accounted for its ability to estimate its feature weights via MERT.

- DOT now has Accuracy-based scoring.

- DOT can now exploit multiple features.

- DOT now incorporates a feature which explicitly accounts for lexical translation equivalences.

We expect that having made DOT's scoring model more flexible, allowing the incorporation of what we believe are the key components of the state-of-the-art approach, will allow fairer comparisons between these models to be made, as well as (by the incorporation of additional features into this model) further improvements in the performance of this syntax-aware model to be achieved.

5.1 Future Work

During the work carried out in this thesis some questions which merit further investigation arose. We now revisit these avenues for future work.

Having obtained translation quality improvements using accuracy-based scoring on a Spanish-to-English translation task, the question of what the behaviour of this algorithm is on different language pairs or data sets arises. Although our algorithm is language pair-agnostic and a priori we would expect our results to be consistent across different language pairs, we consider worthwhile an investigation of the effects of accuracy-based scoring for language pairs other than Spanish-English.

In our experiments with accuracy-based scoring for DOT, we noted that the normalization of the edit distance score surprisingly resulted in an underperforming system. As we explained, we believe this to be caused by our assignment of weights on a manual basis. The corresponding normalized feature for the accuracy-based PB-SMT system, whose weight is automatically determined by MERT, shows the best results for that system. Having now developed a framework in which the weight of the features in DOT can be estimated via MERT, it would be interesting to evaluate the performance of the normalized feature under this weight-assignment method.

Our accuracy-based scoring experiments with DOT required the use of additional unseen data to create an estimation corpus, while we were able to avoid this requirement when adapting the technique to PB-SMT. It would be interesting to investigate the impact of adapting this technique to the case of DOT, and contrast its performance to that obtained using additional data. When adapting deleted estimation for DOT, a method should be developed to solve the problem of the overlap between the unique Goodman indexes assigned to each system, so that when incorporating the accuracy-based score of each fragment into the baseline system, the Goodman index of the fragments match.

As noted in Section 3.2.4, our accuracy-based scoring algorithm could be improved by searching for the sequence of target units (or words) which maximize translation quality, as opposed to using a complete candidate sentence. In addition, the effects of directly using the reference as an oracle could be measured.

It would be interesting to investigate the effects of more sophisticated methods for assigning default scores to unscored units, such as partitioning the set of units according to properties of the units, such as size or labels used in the fragment.

Having successfully adapted our accuracy-based scoring algorithm to PB-SMT and DOT, we see no reason why this method could not be exploited in other log-linear based frameworks, such as the Hierarchical Phrase-Based model (Chiang, 2005, 2007) or the Syntax-Augmented MT system (Zollmann and Venugopal, 2006).

Final avenues for further research of our accuracy-based method are the investigation of the tradeoff between N-best list size and the corresponding amount of scored phrase pairs and translation quality improvement, the effect on oracle selection of different evaluation metrics such as TER (Snover et al., 2006), and the comparison of our phrase-table filtering method with techniques specifically designed for this purpose, e.g. (Johnson et al., 2007; Sánchez-Martínez and Way, 2009).

The log-linear framework we have introduced for DOT enables it to benefit from a range of improvements found in the SMT literature. In particular, all kinds of source information can be exploited as source contexts, such as POS tags (Stroppa et al., 2007), recursiveness of the parse tree, dependency information (Haque et al., 2009b), or head word of the sentence (Haque et al., 2009b). Features of the properties of derivations can also be incorporated, such as minimum amount of fragments or minimum distance to the shortest derivation among all of the derivations that generate a translation, or word length of the translation.

When obtaining our lexical windows, we have prioritized words which lie to the right of the head of a constituent, by including an additional word to the right of the head when window sizes are even, and by discarding words to the left of the head when backing off. This decisions were grounded on properties of right-branching languages. It would be interesting to evaluate the extent to which these decisions impact on the behaviour of our features, by experimenting with prioritising words to the left of the head, and by evaluating performance on non-right-branching languages.

As explained in Chapter 4, we expect further improvements to be possible by the use of an n-gram language model in DOT, if this feature is allowed to be properly computed while building target translations. This will allow an early disposal of derivations for which the boundaries between the words dominated by their fragments are not fluent according to the language model. As noted, an efficient procedure such as cube pruning (Chiang, 2007) would need to be implemented.

The improved DOT model of Chapter 4 could in turn be used as a baseline system under the accuracy-based scoring method. As the rescoring method improves the structural features of DOT by directly modifying the scores in the grammar, and as our source windows aim at improving lexical selection, we expect these methods to be complementary and to lead to even further improvements when combined.

Finally, having incorporated accuracy-based features into DOT, once the implementation of our language model feature has matured and once additional source-context features have been incorporated into our new log-linear model, a comparison with a standard PB-SMT system could be made.

Appendix A

Spanish Tag Set

After obtaining constituency parse trees with the Spanish language pack developed for Bikel's (2002) parser by Chrupała and van Genabith (2006), we post-process the trees discarding all but the first sub-property of part of speech tags. Table A.1 lists the POS tags we obtained.

POS Tag	Description
aq	Qualifying adjective
ao	Ordinal adjective
cc	Coordinating conjunction
cs	Subordinating conjunction
da	Article determiner
dd	Demonstrative determiner
de	Exclamative determiner
di	Indefinite determiner
dn	Numeral determiner
dp	Possessive determiner
dt	Interrogative determiner
Fa	Punctuation: ¡ [
Fc	Punctuation: , (comma)
Fd	Punctuation: :
Fe	Punctuation: ' (single quote)

continued on next page

continued from previous page

POS Tag	Description
Fg	Punctuation: -
Fh	Punctuation: /
Fi	Punctuation: ¿ ?
Fp	Punctuation: .)
Fs	Punctuation: ...
Fx	Punctuation: ;
i	Interjection
nc	Common noun
np	Proper noun
p0	Pronoun (particular cases, cf. (Civit, 2003))
pd	Demonstrative pronoun
pi	Indefinite pronoun
pn	Numeral pronoun
pp	Personal pronoun
pr	Relative pronoun
pt	Interrogative pronoun
px	Possessive pronoun
rg	General adverb
rn	Negative adverb
s	Preposition
va	Auxiliary verb
vm	Main verb
vs	Semiauxiliary verb
w	Date
y	Abbreviation
z	Number

Table A.1: Spanish part-of-speech tags

The constituent labels we use are the ones used by (Chrupała and van Genabith, 2006), which are listed in Table A.2. Some labels not needed to understand the examples in this thesis have been omitted. For a complete list see (Chrupała and van Genabith, 2006).

Tag	Description
sn	Noun phrase
gv	Verbal group
sp	Prepositional phrase
sadv	Adverbial phrase
sa	Adjectival phrase
conj.subord	Subordinating conjunction
coord	Coordinating conjunction
interjeccio	Interjection
neg	Negative adverb
morf	*se*
gerundi	Gerund verb
infinitu	Infinitive verb
S	Root node
S.NF	Non-finite clause
S.NF.R	Relative non-finite clause
grup.nom	Nominal group

Table A.2: Spanish sentence and other constituent tags

Bibliography

Ahrenberg, L. (2007). LinES: An English-Swedish parallel treebank. In *Proceedings of the 16th Nordic Conference of Computational Linguistics*, pages 270–274, Tartu, Estonia.

Arun, A. and Koehn, P. (2007). Online learning methods for discriminative training of phrase based statistical machine translation. In *Proceedings of Machine Translation Summit XI*, pages 15–20, Copenhagen, Denmark.

Bahl, L., Jelinek, F., and Mercer, R. (1983). A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5(2):179–190.

Banerjee, S. and Lavie, A. (2005). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, MI.

Bangalore, S., Haffner, P., and Kanthak, S. (2007). Statistical machine translation through global lexical selection and sentence reconstruction. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 152–159, Prague, Czech Republic.

Bansal, M. and Klein, D. (2010). Simple, accurate parsing with an all-fragments grammar. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1098–1107, Uppsala, Sweden.

Berger, A. L., Della Pietra, V. J., and Della Pietra, S. A. (1996). A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

Bikel, D. (2002). Design of a Multi-lingual, parallel-processing statistical parsing engine. In *Proceedings of the Second International Conference on Human Language Technology Research*, pages 178–182, San Diego, CA.

Blunsom, P., Cohn, T., and Osborne, M. (2008). A discriminative latent variable model for statistical machine translation. In *Proceedings of 46th Annual Meeting of the Association for Computational Linguistics*, pages 200–208, Columbus, OH.

Bod, R. (1992). A computational model of language performance: Data oriented parsing. In *Proceedings of the 15th[sic] conference on Computational linguistics*, pages 855–859, Nantes, France.

Bojar, O. and Hajič, J. (2008). Phrase-Based and Deep Syntactic English-to-Czech Statistical Machine Translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 143–146, Columbus, OH.

Bojar, O., Mareček, D., Novák, V., Popel, M., Ptáček, J., Rouš, J., and Žabokrtský, Z. (2009). English-Czech MT in 2008. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 125–129, Athens, Greece.

Bonnema, R., Buying, P., and Scha, R. (1999). A new probability model for Data Oriented Parsing. In *The Twelfth Amsterdam Colloquium*, Amsterdam, Netherlands.

Brants, T., Popat, A. C., Xu, P., Och, F. J., and Dean, J. (2007). Large language models in machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 858–867, Prague, Czech Republic.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and Regression Trees.* Chapman and Hall/CRC, 1 edition.

Brown, P. F., Cocke, J., Della Pietra, S. A., Della Pietra, V. J., Jelinek, F., Lafferty, J., Mercer, R. L., and Roosin, P. (1990). A Statistical Approach to Machine Translation. *Computational Linguistics*, 16(2):79–85.

Brown, P. F., Cocke, J., Della Pietra, S. A., Della Pietra, V. J., Jelinek, F., Mercer, R. L., and Rossin, P. (1988). A statistical approach to language translation. In *Proceedings of the 12th International Conference on Computational Linguistics*, pages 71–76, Budapest, Hungary.

Brown, P. F., Della Pietra, S. A., Della Pietra, V. J., and Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Callison-Burch, C., Koehn, P., Monz, C., Peterson, K., Przybocki, M., and Zaidan, O. (2010). Findings of the 2010 joint workshop on statistical machine translation and metrics for machine translation. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 17–53, Uppsala, Sweden.

Carpuat, M. and Wu, D. (2007). Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 61–72, Prague, Czech Republic.

Charniak, E. and Johnson, M. (2005). Coarse-to-Fine n-Best Parsing and MaxEnt Discriminative Reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 173–180, Ann Arbor, MI.

Charniak, E., Johnson, M., Elsner, M., Austerweil, J., Ellis, D., Haxton, I., Hill, C., Shrivaths, R., Moore, J., Pozar, M., and Vu, T. (2006). Multilevel coarse-to-fine PCFG parsing. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 168–175, New York, NY.

Chen, S. F. and Goodman, J. (1998). An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Center for Research in Computing Technology, Harvard University, Cambridge, MA.

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

Chiang, D. (2007). Hierarchical Phrase-Based Translation. *Computational Linguistics*, 33(2):201–228.

Chiang, D. (2010). Learning to translate with source and target syntax. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1443–1452, Uppsala, Sweden.

Chiang, D., Marton, Y., and Resnik, P. (2008). Online large-margin training of syntactic and structural translation features. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 224–233, Honolulu, HI.

Chrupała, G. and van Genabith, J. (2006). Using Machine-Learning to Assign Function Labels to Parser Output for Spanish. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 136–143, Sydney, Australia.

Civit, M. (2003). *Criterios de etiquetación y desambiguación morfosintáctica de corpus en español.* PhD thesis, Universidad de Barcelona.

Civit, M. and Martí, M. A. (2004). Building Cast3LB: A Spanish Treebank. *Research on Language and Computation*, 2(4):549–574.

Čmejrek, M., Cuřín, J., Havelka, J., Hajič, J., and Kuboň, V. (2004). Prague Czech-English Dependency Treebank. Syntactically Annotated Resources for Machine

Translation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation*, pages 1597–1600, Lisbon, Portugal.

Collins, M. (1999). *Head-Driven Statistical Models for Natural Language Parsing*. PhD thesis, University of Pennsylvania.

Collins, M., Koehn, P., and Kučerová, I. (2005). Clause restructuring for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 531–540, Ann Arbor, MI.

Cowan, B., Kučerová, I., and Collins, M. (2006). A discriminative model for tree-to-tree translation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 232–241, Sydney, Australia.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2006). Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585.

Damerau, F. J. (1964). A technique for computer detection and correction of spelling errors. *Commun. ACM*, 7(3):171–176.

Dandapat, S., Forcada, M. L., Groves, D., Penkale, S., Tinsley, J., and Way, A. (2010). OpenMaTrEx: A Free/Open-Source Marker-Driven Example-Based Machine Translation System. In *Proceedings of the 7th International Conference on Natural Language Processing (IceTAL 2010)*, pages 121–126, Reykjavik, Iceland.

DeNero, J., Gillick, D., Zhang, J., and Klein, D. (2006). Why generative phrase models underperform surface heuristics. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 31–38, New York, NY.

Ding, Y. and Palmer, M. (2005). Machine translation using probabilistic synchronous dependency insertion grammars. In *Proceedings of 43rd Annual Meeting of the Association for Computational Linguistics*, pages 541–548, Ann Arbor, MI.

Doddington, G. (2002). Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145, San Diego, CA.

Du, J., He, Y., Penkale, S., and Way, A. (2009). MaTrEx: the DCU MT System for WMT 2009. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 95–99, Athens, Grece.

Eisner, J. (2003). Learning non-isomorphic tree mappings for machine translation. In *The Companion Volume to the Proceedings of 41st Annual Meeting of the Association for Computational Linguistics*, pages 205–208, Sapporo, Japan.

Emms, M. (2006). Variants of tree similarity in a question answering task. In *Proceedings of the Workshop on Linguistic Distances*, pages 100–108, Sydney, Australia.

Federico, M. and Cettolo, M. (2007). Efficient Handling of N-gram Language Models for Statistical Machine Translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 88–95, Prague, Czech Republic.

Galley, M., Graehl, J., Knight, K., Marcu, D., DeNeefe, S., Wang, W., and Thayer, I. (2006). Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 961–968, Sydney, Australia.

Galley, M., Hopkins, M., Knight, K., and Marcu, D. (2004). What's in a translation rule? In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 273–280, Boston, MA.

Galron, D., Penkale, S., Way, A., and Melamed, I. D. (2009). Accuracy-Based Scoring for DOT: Towards Direct Error Minimization for Data-Oriented Translation.

In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 371–380, Singapore.

Gimpel, K. and Smith, N. A. (2008). Rich Source-Side Context for Statistical Machine Translation. In *Proceedings of the third Workshop on Statistical Machine Translation*, pages 9–17, OH, USA.

Goodman, J. (1996). Efficient algorithms for parsing the DOP model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 143–152, Philadelphia, PA.

Goodman, J. (1997). Global thresholding and multiple-pass parsing. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 11–25, Providence, RI.

Graehl, J. and Knight, K. (2004). Training tree transducers. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 105–112, Boston, MA.

Hajič, J. (2004). Complex Corpus Annotation: The Prague Dependency Treebank. Bratislava, Slovakia. Jazykovedný ústav Ľ. Štúra, SAV.

Hajič, J., Čmejrek, M., Dorr, B., Ding, Y., Eisner, J., Gildea, D., Koo, T., Parton, K., Penn, G., Radev, D., and Rambow, O. (2004). Natural language generation in the context of machine translation. Technical report, final report from the 2002 CLSP summer workshop. Center for Language and Speech Processing, Johns Hopkins University, Baltimore, MD.

Han, C., Han, N., Ko, E., and Palmer, M. (2002). Development and evaluation of a Korean treebank and its application to NLP. In *In Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC '02)*, pages 1635–1642, Las Palmas, Spain.

Hansen-Schirra, S., Neumann, S., and Vela, M. (2006). Multi-dimensional annotation and alignment in an English-German translation corpus. In *Proceedings of the 5th Workshop on NLP and XML: Multi-Dimensional Markup in Natural Language Processing*, pages 35–42, Trento, Italy.

Haque, R., Naskar, S., Ma, Y., and Way, A. (2009a). Using Supertags as Source Language Context in SMT. In *Proceedings of the 13th Annual Conference of the European Association for Machine Translation*, pages 234–241, Barcelona, Spain.

Haque, R., Naskar, S. K., van den Bosch, A., and Way, A. (2009b). Dependency Relations as Source Context in Phrase-Based SMT. In *Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation (PACLIC'09)*, pages 170–179, Hong Kong.

Haque, R., Naskar, S. K., van den Bosch, A., and Way, A. (2010). Supertags as Source Language Context in Hierarchical Phrase-Based SMT. In *Proceedings of the The Ninth Conference of the Association for Machine Translation in the Americas (AMTA 2010)*, pages 210–219, Denver, CO.

Hassan, H., Sima'an, K., and Way, A. (2007). Supertagged Phrase-based Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 288–295, Prague, Czech Republic.

He, Z., Liu, Q., and Lin, S. (2008). Improving statistical machine translation using lexicalized rule selection. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 321–328, Manchester, UK.

Hearne, M. (2005). *Data-Oriented Models of Parsing and Translation.* PhD thesis, Dublin City University, Dublin, Ireland.

Hearne, M. and Way, A. (2003). Seeing the wood for the trees: Data-oriented translation. In *Proceedings of the Ninth Machine Translation Summit*, pages 165–172, New Orleans, LA.

Hearne, M. and Way, A. (2006). Disambiguation strategies for data-oriented translation. In *Proceedings of the 11th Conference of the European Association for Machine Translation*, pages 59–68, Oslo, Norway.

Huang, L., Knight, K., and Joshi, A. (2006). Statistical syntax-directed translation with extended domain of locality. In *Proceedings of the 7th Conference of the Association for Machine Translation of the Americas*, pages 66–73, Cambridge, MA.

Jelinek, F. and Mercer, R. (1985). Probability distribution estimation from sparse data. *IBM Technical Disclosure Bulletin*, 28:2591–2594.

Jiménez, V. M. and Marzal, A. (2000). Computation of the n best parse trees for weighted and stochastic context-free grammars. In *Proceedings of the Joint IAPR International Workshops on Advances in Pattern Recognition*, pages 183–192, London, UK.

Johnson, H., Martin, J., Foster, G., and Kuhn, R. (2007). Improving translation quality by discarding most of the phrasetable. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 967–975, Prague, Czech Republic.

Johnson, M. (2002). The DOP estimation method is biased and inconsistent. *Computational Linguistics*, 28(1):71–76.

Khalilov, M. (2009). *New statistical and syntactic models for machine translation.* PhD thesis, Universitat Politècnica de Catalunya.

Knight, K. (1997). Automating knowledge acquisition for machine translation. *AI Magazine*, 18(4):81–96.

Koehn, P. (2004). Statistical significance tests for machine translation evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain.

Koehn, P. (2005). Europarl: A Parallel Corpus for Statistical Machine Translation. In *Machine Translation Summit X*, pages 79–86, Phuket, Thailand.

Koehn, P. (2009). *Statistical Machine Translation.* Cambridge University Press.

Koehn, P., Axelrod, A., Mayne, A. B., Callison-Burch, C., Osborne, M., and Talbot, D. (2005). Edinburgh system description for the 2005 IWSLT speech translation evaluation. In *Proceedings of the International Workshop on Spoken Language Translation*, Pittsburgh, PA.

Koehn, P. and Hoang, H. (2007). Factored Translation Models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 868–876, Prague, Czech Republic.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the Annual Meeting of the ACL, demonstation session*, pages 177–180, Prague, Czech Republic.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 48–52, Edmonton, Canada.

Lavie, A. and Agarwal, A. (2007). METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, Prague, Czech Republic.

Lavie, A. and Denkowski, M. J. (2009). The Meteor metric for automatic evaluation of machine translation. *Machine Translation*, 23(2-3):105–115.

Liang, P., Bouchard-Côté, A., Klein, D., and Taskar, B. (2006). An end-to-end discriminative approach to machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 761–768, Sydney, Australia.

Liu, Y., Lü, Y., and Liu, Q. (2009). Improving tree-to-tree translation with packed forests. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 558–566, Singapore.

Ma, W.-Y. and McKeown, K. (2009). Where's the verb? Correcting machine translation during question answering. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and 4th International Joint Conference on Natural Language Processing of the AFNLP, Short Papers*, pages 333–336, Singapore.

Malouf, R. (2002). A Comparison of Algorithms for Maximum Entropy Parameter Estimation. In *Proceedings of the 6th Conference on Natural Language Learning*, pages 49–55, Taipei, Taiwan.

Marcu, D. and Wong, W. (2002). A Phrase-Based, Joint Probability Model for Statistical Machine Translation. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 133–139, Philadelphia, PA.

Marton, Y. and Resnik, P. (2008). Soft Syntactic Constraints for Hierarchical Phrased-Based Translation. In *Proceedings of 46th Annual Meeting of the Association for Computational Linguistics*, pages 1003–1011, Columbus, OH.

Megyesi, B., Dahlqvist, B., Pettersson, E., and Nivre, J. (2008). Swedish turkish parallel treebank. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation*, pages 470–473, Marrakech, Morocco.

Miller, G. and Fellbaum, C. (2007). Wordnet. http://wordnet.princeton.edu/.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan.

Och, F. J., Gildea, D., Khudanpur, S., Sarkar, A., Yamada, K., Fraser, A., Kumar, S., Shen, L., Smith, D., Eng, K., Jain, V., Jin, Z., and Radev, D. (2003). Syntax for statistical machine translation. Technical report, Johns Hopkins University 2003 Summer Workshop on Language Engineering, Final Report, Baltimore, MD.

Och, F. J. and Ney, H. (2002). Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 295–302, Philadelphia, PA.

Och, F. J. and Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Och, F. J. and Ney, H. (2004). The Alignment Template Approach to Statistical Machine Translation. *Computational Linguistics*, 30(4):417–449.

Och, F. J., Tillmann, C., and Ney, H. (1999). Improved alignment models for statistical machine translation. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 20–28, College Park, MD.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, PA.

Penkale, S., Haque, R., Dandapat, S., Banerjee, P., Srivastava, A. K., Du, J., Pecina, P., Naskar, S. K., Forcada, M. L., and Way, A. (2010a). MATREX: The DCU MT System for WMT 2010. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 149–154, Uppsala, Sweden.

Penkale, S., Ma, Y., Galron, D., and Way, A. (2010b). Accuracy-Based Scoring for Phrase-Based Statistical Machine Translation. In *Proceedings of the The Ninth Conference of the Association for Machine Translation in the Americas (AMTA 2010)*, pages 257, 266, Denver, CO.

Petrov, S. and Klein, D. (2007). Improved inference for unlexicalized parsing. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 404–411, Rochester, NY.

Porter, M. F. (1980). An Algorithm for Suffix Stripping. *Program*, 14(3):130–137.

Poutsma, A. (2000). Data-Oriented Translation. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 635–641, Saarbrücken, Germany.

Quirk, C., Menezes, A., and Cherry, C. (2005). Dependency Treelet Translation: Syntactically Informed Phrasal SMT. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 271–279, Ann Arbor, MI.

Sánchez-Martínez, F. and Way, A. (2009). Marker-based filtering of bilingual phrase pairs for SMT. In *Proceedings of the 13th Annual Conference of the European Association for Machine Translation*, pages 144–151, Barcelona, Spain.

Shieber, S. (2004). Synchronous grammars as tree transducers. In *In Proceedings of the Seventh International Workshop on Tree Adjoining Grammar and Related Formalisms*, pages 88–95, Vancouver, Canada.

Sima'an, K. (1996). Computational Complexity of Probabilistic Disambiguation by means of Tree-Grammars. In *Proceedings of the 16th International Conference on Computational Linguistics*, pages 1175–1180, Copenhagen, Denmark.

Sima'an, K. (2004). Robust data oriented spoken language understanding. In *New* *developments in parsing technology*, pages 323–338. Kluwer Academic Publishers, Norwell, MA.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, pages 223–231, Cambridge, MA.

Srivastava, A., Penkale, S., Groves, D., and Tinsley, J. (2009). Evaluating Syntax-Driven Approaches to Phrase Extraction for MT. In *Proceedings of 3rd Workshop on Example-Based Machine Translation*, pages 19–28, Dublin, Ireland.

Stroppa, N., van den Bosch, A., and Way, A. (2007). Exploiting Source Similarity for SMT using Context-Informed Features. In *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 231–240, Skövde, Sweden.

Sun, Y. (2010). Mining the correlation between human and automatic evaluation at sentence level. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta.

Tai, K. (1979). The tree-to-tree correction problem. *Journal of the ACM (JACM)*, 26(3):422–433.

Tillmann, C. and Zhang, T. (2006). A discriminative global training algorithm for statistical MT. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 721–728, Sydney, Australia.

Tinsley, J. (2010). *Resourcing Machine Translation with Parallel Treebanks.* PhD thesis, Dublin City University, Dublin, Ireland.

Tinsley, J., Zhechev, V., Hearne, M., and Way, A. (2007a). Robust Language-Pair

Independent Sub-Tree Alignment. In *Proceedings of the Machine Translation Summit XI*, pages 467–474, Copenhagen, Denmark.

Tinsley, J., Hearne, M., and Way, A. (2007b). Exploiting Parallel Treebanks to Improve Phrase-Based Statistical Machine Translation. In *Proceedings of the Sixth International Workshop on Treebanks and Linguistic Theories*, pages 175–187, Bergen, Norway.

Tinsley, J., Hearne, M., and Way, A. (2009). Parallel treebanks in phrase-based statistical machine translation. In *Proceedings of the Tenth International Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, pages 318–331, Mexico City, Mexico.

Turian, J. P., Shen, L., and Melamed, I. D. (2003). Evaluation of machine translation and its evaluation. In *Proceedings of the Ninth Machine Translation Summit*, pages 386–393, New Orleans, LA.

Vilar, D., Stein, D., and Ney, H. (2008). Analysing soft syntax features and heuristics for hierarchical phrase-based machine translation. In *Proceedings of the International Workshop on Spoken Language Translation*, pages 190–197, Hawaii, USA.

Volk, M., Göhring, A., Marek, T., and Samuelsson, Y. (2010). SMULTRON (version 3.0) — The Stockholm MULtilingual parallel TReebank. http://www.cl.uzh.ch/research/paralleltreebanks_en.html.

Volk, M. and Samuelsson, Y. (2004). Bootstrapping parallel treebanks. In *Proceedings of the 5th International Workshop on Linguistically Interpreted Corpora*, pages 63–70, Geneva, Switzerland.

Wang, C., Collins, M., and Koehn, P. (2007). Chinese syntactic reordering for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 737–745, Prague, Czech Republic.

Watanabe, T., Suzuki, J., Tsukada, H., and Isozaki, H. (2006). NTT Statistical Machine Translation for IWSLT 2006. In *Proceedings of the International Workshop on Spoken Language Translation*, Kyoto, Japan.

Watanabe, T., Suzuki, J., Tsukada, H., and Isozaki, H. (2007). Online large-margin training for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 764–773, Prague, Czech Republic.

Way, A. (2009). A critique of statistical machine translation. In Daelemans, W. and Hoste, V., editors, *Journal of translation and interpreting studies: Special Issue on Evaluation of Translation Technology*, pages 17–41. Linguistica Antverpiensia (LANS 8/2009), Brussels: Academic and Scientific Publishers, Antwerp, Belgium.

Wu, D. (1997). Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–404.

Xia, F. and McCord, M. (2004). Improving a statistical MT system with automatically learned rewrite patterns. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 508–514, Geneva, Switzerland.

Yamada, K. and Knight, K. (2001). A syntax-based statistical translation model. In *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*, pages 523–530, Toulouse, France.

Yamada, K. and Knight, K. (2002). A decoder for syntax-based statistical MT. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 303–310, Philadelphia, PA.

Younger, D. H. (1967). Recognition and parsing of context-free languages in time n3. *Information and Control*, 10(2):189 – 208.

Zaidan, O. F. (2009). Z-MERT: A fully configurable open source tool for minimum error rate training of machine translation systems. *The Prague Bulletin of Mathematical Linguistics*, 91:79–88.

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

Zhang, M., Jiang, H., Aw, A., Li, H., Tan, C. L., and Li, S. (2008). A tree sequence alignment-based tree-to-tree translation model. In *Proceedings of 46th Annual Meeting of the Association for Computational Linguistics*, pages 559–567, Columbus, OH.

Zhechev, V. (2009). *Automatic Generation of Parallel Treebanks: An Efficient Unsupervised System.* PhD thesis, Dublin City University, Dublin, Ireland.

Zhechev, V. and Way, A. (2008). Automatic Generation of Parallel Treebanks. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 1105–1112, Manchester, UK.

Zollmann, A. and Sima'an, K. (2005). A consistent and efficient estimator for data-oriented parsing. *Journal of Automata, Languages and Combinatorics*, 10(2/3):367–388.

Zollmann, A. and Venugopal, A. (2006). Syntax-Augmented Machine Translation via Chart Parsing. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 138–141, New York City, USA.