A Syntactified Direct Translation Model with Linear-time Decoding

Hany Hassan
Cairo TDC
IBM
Cairo, Egypt
hanyh@eg.ibm.com

Khalil Sima'an
Language and Computation
University of Amsterdam
Amsterdam, The Netherlands
k.simaan@uva.nl

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

Abstract

Recent syntactic extensions of statistical translation models work with a synchronous context-free or tree-substitution grammar extracted from an automatically parsed parallel corpus. The decoders accompanying these extensions typically exceed quadratic time complexity.

This paper extends the Direct Translation Model 2 (DTM2) with syntax while maintaining linear-time decoding. We employ a linear-time parsing algorithm based on an eager, incremental interpretation of Combinatory Categorial Grammar (CCG). As every input word is processed, the local parsing decisions resolve ambiguity eagerly, by selecting a single supertag–operator pair for extending the dependency parse incrementally. Alongside translation features extracted from the derived parse tree, we explore syntactic features extracted from the incremental derivation process. Our empirical experiments show that our model significantly outperforms the state-of-the art DTM2 system.

1 Introduction

Syntactic structure is gradually showing itself to constitute a promising enrichment of state-of-the-art Statistical Machine Translation (SMT) models. However, it would appear that the decoding algorithms are bearing the brunt of this improvement in terms of time and space complexity. Most recent extensions work with a synchronous context-free or tree-substitution grammar extracted from an automatically parsed parallel corpus. While attractive in many ways, the decoders that are needed for these types of grammars usually have time and space complexities that are far beyond linear. Leaving pruning aside, there is a genuine question as to whether syntactic structure necessarily implies more complex decoding algorithms. This paper shows that this need not necessarily be the case.

In this paper we extend the Direct Translation Model (DTM2) (Ittycheriah and Roukos, 2007) with target language syntax while maintaining linear-time decoding. With this extension we make three novel contributions to SMT. Our first contribution is to define a linear-time syntactic parser that works as incrementally as standard SMT decoders (Tillmann and Ney, 2003; Koehn, 2004a). At every word position in the target language string, this parser spans *at most a single parse-state* to augment the translation states in the decoder. The parse state summarizes previous parsing decisions and imposes constraints on the set of valid future extensions such that a well-formed sequence of parse states unambiguously defines a dependency structure. This approach is based on an *incremental interpretation* of the mechanisms of Combinatory Categorial Grammar (CCG) (Steedman, 2000).

Our second contribution lies in extending the DMT2 model with a novel set of syntactically-oriented feature functions. Crucially, these feature functions concern the derived (partial) dependency structure as well as local aspects of *the derivation process*, including such information as the CCG lexical categories (supertag), the CCG operators and the intermediate parse states. This accomplishment is interesting both from a linguistic and technical point of view.

Our third contribution is the extension of the standard phrase-based decoder with the syntactic structure and definition of new *grammar-specific pruning techniques* that control the size of the search space. Interestingly, because it is eager, the incremental parser used in this work is hard pushed to perform at a parsing level close to state-

of-the-art cubic-time parsers. Nevertheless, the parsing information it provides allows for significant improvement in translation quality.

We test the new model, called the Dependency-based Direct Translation Model (DDTM), on standard Arabic–English translation tasks used in the community, including LDC and GALE data. We show that our DDTM system provides significant improvements in BLEU (Papineni et al., 2002) and TER (Snover et al., 2006) scores over the already extremely competitive DTM2 system. We also provide results of manual, qualitative analysis of the system output to provide insight into the quantitative results.

This paper is organized as follows. Section 2 reviews the related work. Section 3 discusses the DTM2 baseline model. Section 4 presents the general workings of the incremental CCG parser laying the foundations for integrating it into DTM2. Section 5 details our own DDTM, the dependency-based extension of the DTM2 model. Section 6 reports on extensive experiments and their results. Section 7 provides translation output to shed further detailed insight into the characteristics of the systems. Finally, Section 8 concludes, and discusses future work.

2 Related Work

In (Marcu et al., 2006), it is demonstrated that 'syntactified' target language phrases can improve translation quality for Chinese–English. A stochastic, top-down transduction process is employed that assigns a joint probability to a source sentence and each of its alternative syntactified translations; this is done by specifying a rewriting process of the target parse-tree into a source sentence.

Likewise, the model in (Zollmann and Venugopal, 2006) extends (Chiang, 2005) by augmenting the hierarchical phrases with syntactic categories derived from parsing the target side of a parallel corpus. They use an existing parser to parse the target side of the parallel corpus in order to extract a syntactically motivated, bilingual synchronous grammar as in (Chiang, 2005).

The above-mentioned approaches for incorporating syntax into Phrase-based SMT (Marcu et al., 2006; Zollmann and Venugopal, 2006) share common drawbacks. Firstly, they are based on syntactic phrase-structure parse trees incorporated into a Synchronous CFG or Tree-Substitution Grammar, which makes for a difficult match with non-constituent phrases that are common within Phrase-based SMT. These approaches usually resort to ad hoc solutions to enrich the non-constituent phrases with syntactic structures. Secondly, they deploy chart-based decoders with a high computational cost compared with the phrase-based beam search decoders, e.g., (Tillmann and Ney, 2003; Koehn, 2004a). Thirdly, due to the large parse space, some of the proposed approaches are forced to employ small language models compared to what is usually used in phrase-based systems. To circumvent these computational limitations, various pruning techniques are usually needed, e.g., (Huang and Chiang, 2007).

Other recent approaches, e.g., (Birch et al., 2007; Hassan et al., 2007; Hassan et al., 2008a) incorporate a linear-time supertagger into SMT to take the role of a syntactic language model alongside the standard language model. While these approaches share with our work the use of lexicalized grammars, they never seek to build a full dependency tree or employ syntactic features in order to directly influence the reordering probabilities in the decoder. In the current work, we expand our previous work in (Hassan et al., 2007; Hassan et al., 2008a) to introduce the capabilities of building a full dependency structure and employing syntactic features to influence the decoding process.

Recently, (Shen et al., 2008) introduced an approach for incorporating a dependency-based language model into SMT. They proposed to extract String-to-Dependency trees from the parallel corpus. As the dependency trees are not constituents by nature, they handle non-constituent phrases as well. While this work is in the same general direction as our work, namely aiming at incorporating dependency parsing into SMT, there remain three major differences. Firstly, (Shen et al., 2008) resorted to heuristics to extract the String-to-Dependency trees, whereas our approach employs the well formalized CCG grammatical theory. Secondly, their decoder works bottom-up and uses a chart parser with a limited language model capability (3-grams), while we build on the efficient, linear-time decoder commonly used in phrase-based SMT. Thirdly, (Shen et al., 2008) deploys the dependency language model to augment the lexical language model probability be-

tween two head words but never seek a full dependency graph. In contrast, our approach integrates an incremental parsing capability, that produces the partial dependency structures incrementally while decoding, and thus provides for better guidance for the search of the decoder for more grammatical output. To the best of our knowledge, our approach is the first to incorporate incremental dependency parsing capabilities into SMT while maintaining the linear-time and -space decoder.

3 Baseline: Direct Translation Model 2

The Direct Translation Model (DTM) (Papineni et al., 1997) employs the *a posteriori* conditional distribution $P(T|S)$ of a target sentence T given a source sentence S, instead of the common inversion into $P(S|T)$ based on the source channel approach (Brown et al., 1990). DTM2, introduced in (Ittycheriah and Roukos, 2007), expresses the phrase-based translation task in a unified log-linear probabilistic framework consisting of three components: (i) a prior conditional distribution $P_0(.|S)$, (ii) a number of feature functions $\phi_i()$ that capture the translation and language model effects, and (iii) the weights of the features λ_i that are estimated under MaxEnt (Berger et al., 1996), as in (1):

$$P(T|S) = \frac{P_0(T, J|S)}{Z} \exp \sum_i \lambda_i \phi_i(T, J, S) \quad (1)$$

Here J is the skip reordering factor for the phrase pair captured by $\phi_i()$ and represents the jump from the previous source word, and Z is the per source sentence normalization term. The prior probability P_0 is the prior distribution for the phrase probability which is estimated using the phrase normalized counts commonly used in conventional Phrase–based SMT systems, e.g., (Koehn et al., 2003).

DTM2 differs from other Phrase–based SMT models in that it extracts from a word-aligned parallel corpus only a *non-redundant* set of *minimal phrases* in the sense that no two phrases overlap with each other.

Baseline DTM2 Features: The baseline employs the following five types of features (beside the language model):

- *Lexical Micro Features* examining source and target words of the phrases,
- *Lexical Context Features* encoding the source and target phrase context (i.e. previous and next source and previous target),
- *Source Morphological Features* encoding morphological and segmentation characteristics of source words.
- *Part-of-Speech Features* encoding source and target POS tags as well as the POS tags of the surrounding contexts of phrases.

The DTM2 approach based on MaxEnt provides a flexible framework for incorporating other available feature types as we demonstrate below.

DTM2 Decoder: The decoder for the baseline is a beam search decoder similar to decoders used in standard phrase-based log-linear systems such as (Tillmann and Ney, 2003) and (Koehn, 2004a). The main difference between the DTM2 decoder and the standard Phrase–based SMT decoders is that DTM2 deploys Maximum Entropy probabilistic models to obtain the translation costs and various feature costs by deploying the features described above in a discriminative MaxEnt fashion.

In the rest of this paper we adopt the DTM2 formalization of translation as a discriminative task, and we describe the CCG-based incremental dependency parser that we use for extending the DTM2 decoder, and then list a new set of syntactic dependency feature functions that extend the DTM2 feature set. We also discuss pruning and other details of the approach.

4 The Incremental Dependency Parser

As it processes an input sentence left-to-right word-by-word, the incremental dependency model builds—for each prefix of the input sentence—a partial parse that is a subgraph of the partial parse that it builds for a longer prefix. The dependency graph is constructed incrementally, in that the subgraph constructed at a preceding step is never altered or revised in any later steps. The following schematic view in (2) exhibits the general workings of this parser:

$$S_0 \xrightarrow[w_1, st_1]{o_1} S_1 \xrightarrow[w_2, st_2]{o_2} S_2 \cdots\cdots S_i \xrightarrow[w_i, st_i]{o_i} S_{i+1} \cdots\cdots S_n \quad (2)$$

The syntactic process is represented by a sequence of transitions between adjacent syntactic states S_i.

A transition from state S_{i-1} to S_i scans the current word w_i and stochastically selects a complex lexical descriptor/category st_i and an operator o_i given the local context in the transition sequence. The syntactic state S_i summarizes all the syntactic information about fragments that have already been processed and registers the syntactic arguments which are to be expected next. Only an impoverished deterministic procedure (called a 'State-Realizer') is needed in order to compose a state S_i with the previous states $S_0 \ldots S_{i-1}$ in order to obtain a *fully connected intermediate dependency structure* at every position in the input.

To implement the incremental parsing scheme described above we use the parser described in (Hassan et al., 2008b; Hassan et al., 2009), which is based on Combinatory Categorial Grammar (CCG) (Steedman, 2000). We only briefly describe this parser as its full description is beyond the scope of this paper. The notions of a *supertag* as a lexical category and the process of *supertagging* are both crucial here (Bangalore and Joshi, 1999). Fortunately, CCG specifies the desired kind of lexical categories (supertags) st_i for every word and a small set of combinatory operators o_i that combine the supertag st_i with a previous parse state S_{i-1} into the next parse state S_i. In terms of CCG representations, the parse state is a CCG composite category which specifies either a functor and the arguments it expects to the right of the current word, or is itself an argument for a functor that will follow it to the right. At the first word in the sentence, the parse state consists solely of the supertag of that word.

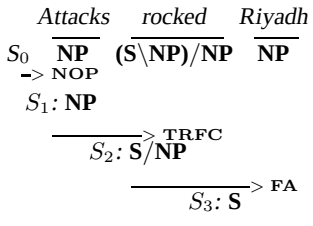

Figure 1: A sentence and possible supertag-, operator- and state-sequences. NOP: No Operation; TRFC: Type Raise-Forward Composition; FA: Forward Application. The CCG operators used show that `Attacks` *and* `Riyadh` *are both dependents of* `rocked`*.*

Figure 1 exhibits an example of the workings of this parser. Practically speaking, after POS tagging the input sentence, the parser employs two components:

- A Supertag-Operator Tagger which proposes a supertag–operator pair for the current word,
- A deterministic State-Realizer, which realizes the current state by applying the current operator to the previous state and the current supertag.

The Supertag-Operator Tagger is a probablistic component while the State-Realizer is a deterministic component. The generative model underlying this component concerns the probability $P(W, S)$ of a word sequence $W = w_1^n$ and a parse-state sequence $S = S_1^n$, with associated supertag sequence $ST = st_1^n$ and operator sequence $O = o_1^n$, which represents a possible derivation. Note that given the choice of supertags st_i and operator o_i, the state S_i is calculated deterministically by the State-Realizer.

A generative version of this model is described in (3):

$$P(W,S) = \prod_{i=1}^{n} \overbrace{P(w_i|W_{i-1}S_{i-1})}^{Word\ Predictor} . \overbrace{P(st_i|W_i)}^{Supertagger} . \overbrace{P(o_i|W_i, S_{i-1}, ST_i)}^{Operator\ Tagger} \quad (3)$$

In (3):

- $P(W, S)$ represents the product of the production probabilities at each parse-state and is similar to the structured language model representation introduced in (Chelba, 2000).
- $P(w_i|W_{i-1}S_{i-1})$ is the probability of w_i given the previous sequence of words W_{i-1} and the previous sequence of states S_{i-1},
- $P(st_i|W_i)$: is the supertag st_i probability given the word sequence W_i up to the current position. Basically, this represents a sequence tagger (a 'supertagger').
- $P(o_i|W_i, S_{i-1}, ST_i)$ represents the probability of the operator o_i given the previous words, supertags and state sequences up to the current position. This represents a CCG operator tagger.

The different local conditional components (for every i) in (3) are estimated as discriminative MaxEnt submodels trained on a corpus of *incremental CCG derivations*. This corpus was extracted from the CCGbank (Hockenmaier, 2003)

by transforming every normal form derivation into strictly left-to-right CCG derivations, with the CCG operators only slightly redesigned to allow incrementality while still satisfying the dependencies in the CCGbank (cf. (Hassan et al., 2008b; Hassan et al., 2009)).

As mentioned before, the State-Realizer is a deterministic function. Starting at the first word with (obviously) a null previous state, the realizer performs the following deterministic steps for each word in turn: (i) set the current supertag and operator to those of the current word; (ii) at the current state, apply the current operator to the previous state and current supertag; (iii) add edges to the dependency graphs between words that are linked as CCG arguments; and (iv) if not at the end of the sentence, set the previous state to the current one, then set the current word to the next one, and iterate from (i).

It is worth noting that the proposed dependency parser is deterministic in the sense that it maintains *only one parse state per word*. This characteristic is crucial for its incorporation into a large-scale SMT system to avoid explosion of the translation space during decoding.

5 Dependency-based DTM (DDTM)

In this section we extend the DTM2 model with incremental target dependency-based syntax. We call the resulting model the Dependency-based Direct Translation Model (DDTM). This extension takes place by (i) extracting syntactically enriched minimal phrase pairs, (ii) including a new set of syntactic feature functions among the exponential model features, and (iii) adapting the decoder for dealing with syntax, including various pruning strategies and enhancements. Next we describe each extension in turn.

5.1 Phrase Table: Incremental Syntax

The target-side sentences in the word-aligned parallel corpus used for training are parsed using the incremental dependency parser described in section 4. This results in a word-aligned parallel corpus where the words of the target sentences are tagged with supertags and operators. From this corpus we extract the set of minimal phrase pairs using the method described in (Ittycheriah and Roukos, 2007), extracting along with every target phrase the associated sequences of supertags and operators. As shown in (4), a source phrase $s_1, \ldots, s_n$ translates into a target phrase $w_1, \ldots, w_m$ where every word w_i is labeled with a supertag st_i, and a possible parsing operator o_i appearing with it in the parsed parallel corpus:

$$s_1 ... s_n \longrightarrow [w_1, st_1, o_1] ... [w_m, st_m, o_m] \quad (4)$$

Hence, our phrase table associates with every target phrase an *incremental parsing subgraph*. These subgraphs along with their probabilities represent our phrase table augmented with incremental dependency parsing structure.

This representation turns the complicated problem of MT with incremental parsing into a sequential classification problem in which the classifier deploys various features from the source sentence and the candidate target translations to specify a sequence of decisions that finally results in an output target string along with its associated dependency graph. The classification decisions are performed in sequence step-by-step while traversing the input string to provide decisions on possible words, supertags, operators and states. A beam search decoder simultaneously decides which sequence is the most probable.

5.2 DDTM Features

The exponential model and the MaxEnt framework used in DTM2 and DDTM enabled us to explore the utility of incremental syntactic parsing within a rich feature space. In our DDTM system, we implemented a set of features alongside the baseline DTM2 features that were discussed in Section 3. The features described here encode all the probabilistic components in (3) within a log linear interpretation along with some more empirically intuitive features.

- Supertag-Word features: these features examine the target phrase words with their associated supertags and is related to the Supertagger component in (3).
- Supertag sequence features: these features encode *n*-gram supertags (equivalent to the *n*-gram supertags Language Model). This feature is related to the supertagger component as well.
- Supertag-Operator features: these features encode supertags and associated operators which is related to the Operator Tagger component in (3).

- Supertag-State features: these features register state and supertag co-occurrences.
- State sequence features: these features encode *n*-gram state features and are equivalent to an *n*-gram Language Model over parse state sequences which is related to the multiplication in (3).
- Word-State sequence features: these features encode words and states co-occurrences which is related to the Word Predictor component in (3).

The exponential model and the MaxEnt framework used in DTM2 and DDTM enable us to explore the utility of incremental syntactic parsing with the use of minimal phrases within a rich feature space.

5.3 DDTM Decoder

In order to support incremental dependency parsing, we extend the DTM2 decoder in three ways: firstly, by constructing the syntactic states during decoding; secondly, by extending the hypothesis structures to incorporate the syntactic states and the partial dependency derivations; and thirdly, by modifying the pruning strategy to handle the large search space.

At decoding time, each hypothesis state is associated with a parse-state which is constructed while decoding using the incremental parsing approach introduced in ((Hassan et al., 2008b; Hassan et al., 2009)). The previous state, the sequences of supertags and CCG incremental operators are deployed in a deterministic manner to realize the parse-states as well as the intermediate dependency graphs between words.

Figure 2 shows the DDTM decoder while decoding a sentence with the English translation "Attacks rocked Riyadh". Each hypothesis is associated with a parse-state S_i and a partial dependency graph (shown for some states only). Moreover, each transition is associated with an operator o that combines the previous state and the current supertag st to construct the next state S_i. The decoder starts from a null state S_1 and then proceeds with a possible expansion with the word "attacks", supertag NP and operator NOP to produce the next hypothesis with state S_2 and category NP. Further expansion for that path with the verb "rocked", supertag '$(S\backslash NP)/NP$ and operator $TRFC$ will produce the state S_5 with category S/NP. The partial dependency graph for state S_5 is shown above the state where a dependency relation between the two words is established. Furthermore, another expansion with the word "Riyadh", supertag NP and operator FA produces state S_7 with category S and a completed dependency graph as shown above the state. Another path which spans the states S_1, S_3 , S_6 and S_8 ends with a state category S/NP and a partial dependency graph as shown under state S_8 where the dependency graph is still missing its object (e.g. "Riyadh attacks rocked the Saudi Govt.").

The addition of parse-states may result in a very large search space due to the fact that the same phrase/word may have many possible supertags and many possible operators. Moreover, the same word sequences may have many parse-state sequences and, therefore, many hypotheses that represent the same word sequence. The search space is definitely larger than the baseline search space. We adopt the following three pruning heuristics to limit the search space.

5.3.1 Grammatical Pruning

Any hypothesis which does not constitute a valid parse-state is discarded, i.e. if the previous parse-state and the current supertag sequence cannot construct a valid state using the associated operator sequence, then the expansion is discarded. Therefore, this pruning strategy maintains only fully connected graphs and discards any partially connected graphs that might result during the decoding process.

As shown in Figure 2, the expansion from state S_1 to state S_4 (with the dotted line) is pruned and not expanded further because the proposed expansion is the verb "attacks", supertag $(S\backslash NP)/NP$ and operator $TRFC$. Since the previous state is NULL, it cannot be combined with the verb using the $TRFC$ operator. This would produce an undefined state and thus the hypothesis is discarded.

5.3.2 Supertags and Operators Threshold

We limit the supertag and operator variants per target phrase to a predefined number of alternatives. We tuned this on the MT03 DevSet for the best accuracy while maintaining a manageable search space. The supertags limit was set to four alternatives while the operators limit was set to three.

As shown in Figure 2, each word can have many alternatives with different supertags. In this example the word "attacks" has two forms, namely a

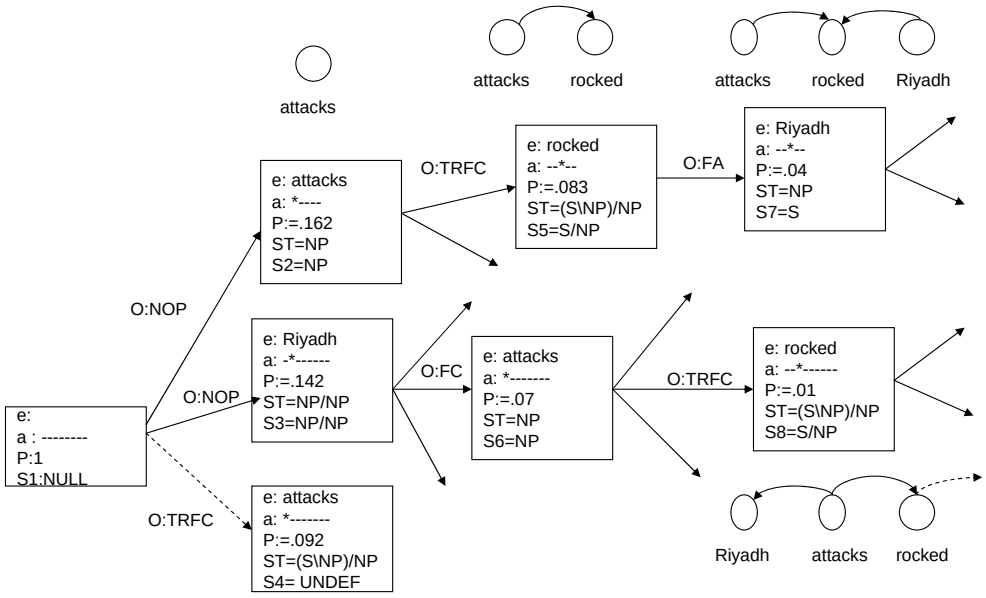

Figure 2: DDTM Decoder: each hypothesis has a parse state and a partial dependency structure.

noun and a verb, with different supertags and operators. The proposed thresholds limit the possible alternatives to a reasonable number.

5.3.3 Merging Hypotheses

Standard Phrase–based SMT decoders merge translation hypotheses if they cover the same source words and share the same *n*-gram language model history. Similarly, DDTM decoder merges translation hypotheses if they cover the same source words, share the same *n*-gram language model history and share the same parse-state history. This helps in reducing the search space by merging paths that will not constitute a part of the best path.

6 Experiments

We conducted experiments on an Arabic-to-English translation task using LDC parallel data and GALE parallel data. We used the UN parallel corpus and LDC news corpus together with the GALE parallel corpus, totaling 7.8M parallel sentences. The 5-gram Language Model was trained on the English Gigaword Corpus and the English part of the parallel corpus. Our baseline system is similar to the system described in (Ittycheriah and Roukos, 2007). We report results on NIST MT05 and NIST MT06 evaluations test sets using BLEU and TER as automatic evaluation metrics.

To train the DDTM model, we use the incremental parser introduced in (Hassan et al., 2008b; Hassan et al., 2009) to parse the target side of the parallel training data. Each sentence is associated with supertag, operator and parse-state sequences. We then train models with different feature sets.

Results: We compared the baseline DTM2 (Ittycheriah and Roukos, 2007) with our DDTM system with the features listed above. We examine the effect of all features on system performance. In this set of experiments we used LDC parallel data only which is composed of 3.7M sentences and the results are reported on MT05 test set. Each of the examined systems deploys DTM2 features in addition to a number of newly added syntactic features. The systems examined are:

- DTM2: Direct Translation model 2 baseline.
- D-SW: DTM2 + Supertag-Word features.
- D-SLM: DTM2 + Supertag-Word and supertag *n*-gram features.
- D-SO: DTM2+ Supertag-Operator features.
- D-SS : DTM2 + supertags and states features with parse-state construction.
- D-WS : DTM2 + words and states features with parse-state construction.
- D-STLM: DTM2 + state *n*-gram features with parse-state construction.
- DDTM: fully fledged system with all features that proved useful above which are: Supertag-Word features, supertag *n*-gram

features, supertags and states features and state *n*-gram features .

System	BLEU Score on MT05
DTM2-Baseline	52.24
D-SW	52.28
D-SLM	52.29
D-SO	52.01
D-SS	52.39
D-WS	52.03
D-STLM	52.53
DDTM	**52.61**

Table 1: DDTM Results with various features.

As shown in Table 1, the DTM baseline system demonstrates a very high BLEU score, unsurprisingly given its top-ranked performance in two recent major MT evaluation campaigns. Among the features we tried, supertags and *n*-gram supertags systems (D-SW and D-SLM systems) give slight yet statistically insignificant improvements. On the other hand, the states *n*-gram sequence features (D-SS and DDTM systems) give small yet statistically significant improvements (as calculated via bootstrap resampling (Koehn, 2004b)). The D-WS system shows a small degradation in performance, probably due to the fact that the states-words interactions are quite sparse and could not be estimated with good evidence. Similarly, the D-SO system shows a small degradation in performance. When we investigated the features types, we found out that all features that deploy the operators had bad effect on the model. We think this is due to the fact that the operator set is a small set with high evidence in many training instances such that it has low discriminative power on it is own. However, it implicitly helps in producing the state sequence which proved useful.

System	DTM2-Baseline	DDTM
MT05 (BLEU)	55.28	55.66
MT05 (*TER*)	38.79	38.48
MT06 (BLEU)	43.56	43.91
MT06 (*TER*)	49.08	48.65

Table 2: DDTM Results on MT05 and MT06.

We examined a combination of the best features in our DDTM system on a larger training data comprising 7.8M sentences from both NIST and GALE parallel corpora. Table 2 shows the results on both MT05 and MT06 test sets. As shown, DDTM significantly outperforms the state-of-the-art baseline system. It is worth noting that DDTM outperforms this baseline even when very large amounts of training data are used. Despite the fact that the actual scores are not so different, we found that the baseline translation output and the DDTM translation outout are significantly different. We measured this by calculating the TER between the baseline translation and the DDTM translation for the MT05 test set, and found this to be 25.9%. This large difference has not been realized by the BLEU or TER scores in comparison to the baseline. We believe that this is due to the fact that most changes that match the syntactic constraints do not bring about the best match where the automatic evaluation metrics are concerned. Accordingly, in the next section we describe the outcome of a detailed manual analysis of the output translations.

7 Manual Analysis of Results

Although the BLEU score does not mark a large improvement by the dependency-based system over the baseline system, human inspection of the data gives us important insights into the pros and cons of the dependency-based model. We analyzed a randomly selected set of 100 sentences from the MT05 test set. In this sample, the baseline and the DDTM system perform similarly in 68% of the sentences. The outputs of both system are similar though not identical. In these cases, the systems may choose equivalent paraphrases. However, the translations using syntactic structures are rather similar. It is worth noting that the DDTM system tends to produce more concise syntactic structures which may lead to less BLUE score due to penalizing the translation length although the translation might be equivelent to the baseline if not better.

In 28% of the sentences, the DDTM system produces remarkably better translations. The examples here illustrate the behaviour of the baseline and the DDTM systems which can be observed consistently throughout the test set. We only highlight some of the examples for illustration purposes. DDTM manages to insert verbs which are deleted by any standard phrase-based SMT system. DDTM prefers to deploy verbs since they have complex and more detailed syntactic structures which give better and more likely state se-

quences. Furthermore, the DDTM system avoids longer noun phrases and instead uses some prepositions in-between. Again, this is probably due to the fact that like verbs, prepositions have a complex syntactic description that give rise to more likely state sequences.

On the other hand, the baseline produced better translation in 8% of the analysis sample. We observed that the baseline is doing better mainly in two cases. The first when the produced translation is very poor and producing poor sysntatctic structure due to out of vocabularies or hard to translate sentences. The second case is with sentences with long noun phrases, in such cases the DDTM system prefres to introduce verbs or prepositions in the middle of long noun phrase and thus the baseline would produce better translations. This is maybe due to the fact that noun phrases have relatively simple structure in CCG such that it did not help in constructing long noun phrases.

Source: وخضع بعد ذلك لفحوصات اجراهَا احد اطباء الشرطة
Reference: *He then underwent medical examinations by a police doctor .*
Baseline: *He was subjected after that tests conducted by doctors of the police .*
DDTM: *Then he underwent tests conducted by doctors of the police .*

Source: وقد هز الرياض مساء اليوم هجومان بسيارتين مفخـختين
Reference: *Riyadh was rocked tonight by two car bomb attacks..*
Baseline: *Riyadh rocked today night attacks by two booby - trapped cars.*
DDTM: *Attacks rocked Riyadh today evening in two car bombs.*

Figure 3: DDTM provides better syntactic structure with more concise translations.

Figure 3 shows two examples where DDTM provides better and more concise syntactic structure. As we can see, there is not much agreement between the reference and the proposed translation. However, longer translations enhance the possibility of picking more common *n*-gram matches via the BLEU score and so increases the chance of better scores. This well-known bias does not favour the more concise output derived by our DDTM system, of course.

8 Conclusion and Future Work

In this paper, we presented a novel model of dependency phrase-based SMT which integrates incremental dependency parsing into the translation model while retaining the linear decoding assumed in conventional Phrase–based SMT systems. To the best of our knowledge, this model constitutes the first effective attempt at integrating a linear-time dependency parser that builds a connected tree incrementally into SMT systems with linear-time decoding. Crucially, it turns out that incremental dependency parsing based on lexicalized grammars such as CCG and LTAG can provide valuable incremental parsing information to the decoder even if their output is imperfect. We believe this robustness in the face of imperfect parser output to be a property of the probabilistic formulation and statistical estimation used in the Direct Translation Model. A noteworthy aspect of our proposed approach is that it integrates features from the derivation process as well as the derived tree. We think that this is possible due to the importance of the notion of a derivation in linguistic frameworks such as CCG and LTAG.

Future work will attempt further extensions of our DDTM system to allow for the exploitation of long-range aspects of the dependency structure. We will work on expanding the features set of DDTM system to leverage features from the constructed dependency structure itself. Finally, we will work on enabling the deployment of source side dependency structures to influence the construction of the target dependency structure based on a bilingually enabled dependency parsing mechanism using the discriminative modeling capabilities.

Acknowledgments

We would like to thank Salim Roukos, IBM TJ Watson Research Center, for fruitful, insightful discussions and for his support during this work. We also would like to thank Abe Ittycheriah, IBM TJ Watson Research Center, for providing the DTM2 baseline and for his support during developing this system. Finally, we would like to thank the anonymous reviewers for their helpful and constructive comments.

References

Bangalore, S. and Joshi, A. (1999). "Supertagging: An Approach to Almost Parsing", *Computational Linguistics* **25**(2):237–265, 1999.

Berger, A. and Della Pietra, S. and Della Pietra, V.J. (1996). Maximum Entropy Approach to Natural Language Processing *Computational Linguistics*, **22**(1): 39–71, 1996.

Birch, A., Osborne, M. and Koehn, P. (2007). CCG Supertags in Factored Statistical Machine Translation. In *Proceedings of the Second Workshop on Statistical Machine Translation, ACL-07*, pp.9–16, 2007.

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

Chelba, C. (2000). Exploiting Syntactic Structure for Natural Language Modeling. PhD thesis, Johns Hopkins University, Baltimore, MD.

Chiang, D. (2005). A Hierarchical Phrase-Based Model for Statistical Machine Translation. In *43rd Annual Meeting of the Association for Computational Linguistics (ACL05)*, pp.263–270, Ann Arbor, MI.

Hassan, H., Sima'an, K., and Way, A.. (2009). Lexicalized Semi-Incremental Dependency Parsing. In *Proceedings of RANLP 2009, the International Conference on Recent Advances in Natural Language Processing*, Borovets, Bulgaria (to appear).

Hassan, H., Sima'an, K., and Way, A. (2008a). Syntactically Lexicalized Phrase-Based Statistical Translation. *IEEE Transactions on Audio, Speech and Language Processing*, **6**(7):1260–1273.

Hassan, H., Sima'an, K., and Way, A.. (2008b). A Syntactic Language Model Based on Incremental CCG Parsing. In *Proceedings IEEE Workshop on Spoken Language Technology (SLT) 2008*, Goa, India.

Hassan, H., Sima'an, K., and Way, A. (2007). Integrating Supertags into Phrase-based Statistical Machine Translation. In *Proceedings of the ACL-2007, Prague, Czech Republic, pp.288–295, 2007*.

Hockenmaier, J. (2003). *Data and Models for Statistical Parsing with Combinatory Categorial Grammar*, Ph.D Thesis, University of Edinburgh, UK, 2003.

Huang, L. and Chiang, D. (2007). Forest Rescoring: Faster Decoding with Integrated Language Models. In *Proceedings of the ACL-2007, Prague, Czech Republic, 2007*.

Ittycheriah, A. and Roukos, S. (2007). Direct translation model 2. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pp.57–64, Rochester, NY.

Koehn, P. (2004a). Pharaoh: A Beam Search Decoder for phrase-based Statistical Machine Translation Models. Machine Translation: From Real Users to Research. In *Proceedings of 6th Conference of the Association for Machine Translation in the Americas, AMTA 2004*, pp.115–124, Washington, DC.

Koehn, P. (2004b). Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp.388–395, Edmonton, AB, Canada.

Koehn, P. Och, F.J. and Marcu, D. (2003). Statistical phrase-based translation. In *Proceedings of the Joint Human Language Technology Conference and the Annual Meeting of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2003)*, pp.127–133, Edmonton, AL, Canada.

Marcu, D., Wang, W., Echihabi, A., and Knight, K. (2006). SPMT: Statistical Machine Translation with Syntactified Target Language Phrases. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006)*, pp.44–52, Sydney, Australia.

Papineni, K., Roukos, S., and Ward, T. (1997). Feature-Based Language Understanding. In *Proceedings of 5th European Conference on Speech Communication and Technology EUROSPEECH '97* , pp.1435–1438, Rhodes, Greece.

Papineni, K., Roukos, S., Ward, T. and Zhu, W-J. (2002). BLEU: a Method for Automatic Evaluation of Machine Translation. In *40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, pp.311–318, Philadelphia, PA.

Shen, L., Xu, J., and Weischedel, R. (2008). A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL-08: HLT*, pp.577–585, Columbus, OH.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L. and Makhoul, J. (2006) A Study of Translation Edit Rate with Targeted Human Annotation. In *AMTA 2006: Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, pp.223–231, Cambridege, MA.

Steedman, M. (2000). *The Syntactic Process*. MIT Press, Cambridge, MA.

Tillmann, C. and Ney, H. (2003). Word reordering and a dynamic programming beam search algorithm for statistical machine translation. *Computational Linguistics*, 29(1):97–133.

Zollmann, A. and Venugopal, A. (2006). Syntax augmented machine translation via chart parsing. In *Proceedings of the Workshop on Statistical Machine Translation, HLT/NAACL*, pp.138–141, New York, NY.