English-Turkish Parallel Treebank with Morphological Annotations and its Use in Tree-based SMT

Onur Görgün[1], Olcay Taner Yıldız[2], Ercan Solak[2] and Razieh Ehsani[2]
[1]*Alcatel-Lucent Teletaş Telekomünikasyon A.Ş., İstanbul, Turkey*
[2]*Department of Computer Engineering, Işık University, İstanbul, Turkey*

Keywords: Machine Translation, Tree-based Translation.

Abstract: In this paper, we report our tree based statistical translation study from English to Turkish. We describe our data generation process and report the initial results of tree-based translation under a simple model. For corpus construction, we used the Penn Treebank in the English side. We manually translated about 5K trees from English to Turkish under grammar constraints with adaptations to accommodate the agglutinative nature of Turkish morphology. We used a permutation model for subtrees together with a word to word mapping. We report BLEU scores under simple choices of inference algorithms.

1 INTRODUCTION

Statistical machine translation have come a long way in the recent years. After IBM model's superior performance over classical rule-based approaches, machine translation research has flourished with statistical models of increasing complexity. With the increase of computational power and availability of the parallel corpora, researchers switch from manually crafted linguistic models to empirically learned statistical models, from string/word based models to tree/phrase based models (Jurafsky and Martin, 2009).

Early approaches (Hutchinson, 1994) in statistical machine translation are word-based models. These models treat the words as the main translation unit and map the source word(s) into target word(s) by inserting, deleting, and reordering. More formally, target word(s) should be aligned to their corresponding word(s) in the target language. On the other hand, since word-based models take words as the basic translation units, alignment gathered by these models sometimes does not reflect the real alignment. For languages with different fertilities like English-Turkish, some words are left unaligned. For example Turkish word "gideceğim" (I will go) should be translated as a verbal phrase. However, its English match is usually extracted as "go" by word-based models. Hence, translation does not reflect the real meaning of the Turkish phrase.

Phrase-based model and its derivatives have been used as machine translation models for English-Turkish language pair. (El-Kahlout and Oflazer, 2006) proposed the first approach in statistical machine translation between English-Turkish. They improve the BLEU score from the baseline 7.52 to 9.13 with 23K sentences. Continuing the efforts, (Yeniterzi and Oflazer, 2010) offer factored phrase-based translation model. In their approach, they define custom complex syntactic tags at the English side and input the augmented sentence into phrase-based translation system. In another work, (El-Kahlout, 2009) investigates the effects of different sub-lexical representational structures. They obtain a BLEU score of 25 with nearly 20K sentences.

Integration of syntactic structure into machine translation models are known to improve the performance. In particular, tree-based models are good at incorporating the recursive structure of language and offer better alignment results compared to phrase-based models (Koehn, 2010). Ordering problems are more difficult in language pairs like English and Turkish, where the unmarked orders are different across the grammar and the latter has many scrambling options.

In this paper, we study a tree-based approach to English-to-Turkish translation. Our contributions are two-fold; (i) we manually translate and annotate syntactically and morphologically over 5K sentences from Penn-Treebank (Marcus et al., 1993), (ii) we propose a tree-based translation approach. Our tree-based translation approach has two phases. In the first

Görgün, O., Yıldız, O., Solak, E. and Ehsani, R.
English-Turkish Parallel Treebank with Morphological Annotations and its Use in Tree-based SMT.
DOI: 10.5220/0005653905100516
In *Proceedings of the 5th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2016)*, pages 510-516
ISBN: 978-989-758-173-1
Copyright © 2016 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

phase, we obtain the best permutation of the English parse tree by permuting subtrees at each node and obtain a Turkish tree up to English leaves. In the second phase, we replace the leaves of the permuted tree with the appropriate tokens from a word-for-word model. Although this is the first tree approach for English-Turkish translation, the results are promising.

The paper is organized as follows. In Section 2, we give the details of our data preparation steps. We give the training phase of our tree-based algorithm in Section 3, and translation using that trained model in Section 3.3. We give the experimental results in Section 4 and conclude in Section 5.

2 DATA PREPARATION

The Penn Treebank (Marcus et al., 1993) contains more than 40K syntactically annotated sentences in its WSJ section. We used a subset of about 5K of those sentences. Each sentence contains at most 15 tokens, including punctuation.

The tag set in Penn Treebank II is quite extensive. Besides POS and phrase tags, the Treebank also includes markings to identify semantic functions, subject-predicate dependencies and movement traces. At this preliminary stage of our work, we dropped all of these extra markings and used only the bare tags. For example, for the tag NP-SBJ-1, we use only NP.

In our translation application, a parse tree of an English sentence is translated into Turkish through the repeated application of subtree permutation and label replacement. We confine the label replacement to leaf nodes. In training, the model parameters are tuned over a set of parallel trees. In translating a given English parse tree, we go over its every internal node and apply the most probable permutation to its children. For leaf nodes, we use a count-based word-for-word replacement of English words with Turkish stems or meta morphemes. The final surface form of the translated Turkish sentence is generated by applying morphophonemic constraints to meta morphemes.

In constructing our corpus, we manually performed the subtree permutation and leaf replacement of the sentences in our restricted Penn Treebank corpus. We divided the the data generation into three phases. Below, we describe the data in each phase and its relation to our translation task.

2.1 Phase 0

Phase 0 data consists of the original English parse trees from the Penn Treebank II. English trees are used to obtain translated trees in the following phases with respect to the transformation heuristics defined.

2.2 Phase 1

In this phase, subtrees are manually permuted and the leaf tokens are manually replaced with Turkish glosses. In doing so, the translators obeyed the following heuristics as closely as possible.

1. Majority of Turkish sentences have a SOV order. We tried to permute subtrees to reflect this tendency.
2. Turkish is a head final language. In order to reflect this restriction, we permuted the constituents in noun, prepositional, adjective and adverb phrases.
3. We permuted modals, particles and verb tense markers to align with the order of corresponding Turkish morphemes as dictated by the morphotactical rules.
4. We use only whole words in the leaves. We did not allow any dangling morphemes.
5. When we embedded a functional word in the English leaf as a morpheme in the stem of a Turkish leaf, we replaced the English leaf with *NONE*.
6. In Turkish there is no determiner corresponding to “the” in English. We replaced “the” with *NONE*.
7. We usually replaced prepositions with *NONE* and attached their corresponding case to the nominal head of the Turkish phrase.
8. In Turkish grammar, number agreement between the verb and the subject is somewhat relaxed. So, we let the translators deviate from the literal gloss whenever it seems natural to do so.
9. Verb tenses do not usually form exactly matching categories across pairs of languages. We replaced English verb tenses with their closest semantic classes in Turkish.
10. When we translated the English tree for a question sentence, we replaced its wh- constituent with *NONE* and replaced its trace with the appropriate question pronoun in Turkish. This reflects the question formation process in Turkish, where any constituent can be questioned by simply replacing it with a suitable question pronoun and case inflection.
11. We translated the proper nouns as their common Turkish gloss if there is one, kept the original otherwise.

12. We replaced subordinating conjunctions, marked as "IN" in English sentences, with *NONE* and appended the appropriate participle morpheme to the stem in the Turkish translation.
13. As it often happens when translating between languages from different families, a multiword expression in Turkish may correspond to a single English word. Conversely, more than one words in English may correspond to a single word in Turkish. In the latter case, we replaced some of the English words in the expression with *NONE*.

The pairs of English and Turkish parallel trees in Figure 1 illustrate some of the heuristics above.

At the end of Phase 1, two sets of trees are generated. The first set has only the permuted trees where the leaves are still English tokens. We call this data set Phase-1-English. The second set has either the leaves replaced with Turkish glosses or the *NONE* tag. We call this data set Phase-1-Turkish. Having two such sets of data proved useful in pinning down sources of errors in the final translation sentence. The tree generated in Phase-1-English phase for the trees in Figure 1 is given in Figure 2.

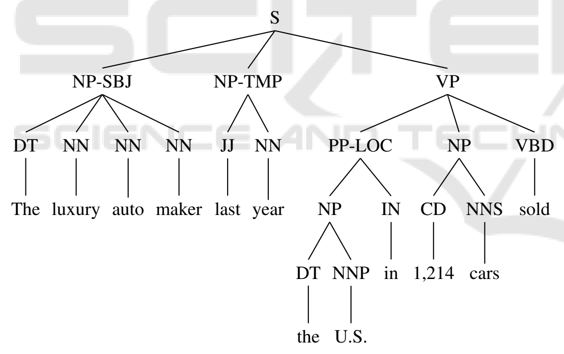


Figure 1: Phase-1-English tree for the parallel pair in Figure 2.

2.3 Phase 2

Many constituents in English sentences are translated into inflectional morphemes in corresponding Turkish words. Thus, any task of automatic translation to or from Turkish needs to use morphological structure of Turkish words.

In the Phase 2 of our data generation, the human annotators generated the morphological analyses of the Turkish words at the leaf nodes of the trees generated in Phase 1. In order to speed up the annotation and reduce the number of errors, we built a FST based morphological analyzer for this step. The analyzer is embedded within our annotation tool. When the annotator selects a Turkish word, the tool displays all the analyses of the word and the annotator selects the correct analysis. Thus, the analysis is automatic and the disambiguation is manual.

Figure 3 illustrates the morphological analysis of the sentence

(1) Stewart & Stevenson dizel ve gaz türbinleri ile çalıştırılan ekipman üretir.

Stewart & Stevenson makes equipment powered with diesel and gas turbines.

2.4 Phase 3

Inflectional morphemes of Turkish words usually correspond to functional words in English sentences. For example, the following pair of English sentence and its translation has the morpheme-word pairs given below.

(2) İş-e git-me-yecek-sin.
Work.DAT go.NEG.FUT.2SG
You will not go to work.

Thus, a natural extension of permute-and-replace translation is to allow the English functional words to be replaced by suitable morphemes in Turkish. In this phase, we detach the individual morphological tags of the words analyzed in Phase 2. Then, the annotators move these tags to the suitable empty slots identified by the tag *NONE* at the leaves. The tag movement respects the morphotactics of Turkish. In most cases, the order of morphemes in the Turkish word matches the order of leaves vacated by the English functional words in the permuted tree. Figure 3 illustrates the process.

In Phase 3, we generate two sets of data. The first one has the tags identifying the morphemes. In the second one, the tags are replaced by their canonical forms. for example, the tag -ABL is replaced by -dAn and the tag -NEG is replaced by -mA and so on. We identify the latter set of trees as Phase-3-Canonical. We used canonical forms instead of symbols because the human annotators found working with canonical morphemes more intuitive in moving them around. Moreover, some tags correspond to null morphemes and do not have visible surface forms.

Although we described the 3 level process as if they are completely independent, in practice we saw that the human annotators usually go back and forth among the phases and use feedback across the phases. For example, in trying to move canonical morphemes in Phase 3, human annotators sometimes encounter

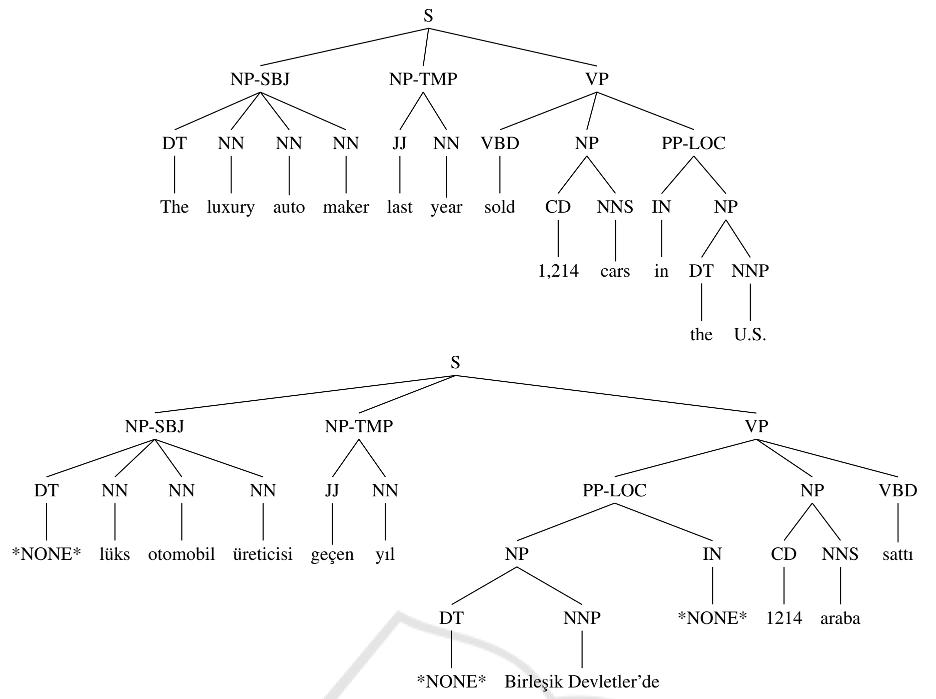

S
NP-SBJ
NP-TMP
VP
DT NN NN NN JJ NN VBD NP PP-LOC
The luxury auto maker last year sold
CD NNS IN NP
1,214 cars in DT NNP
the U.S.
S
NP-SBJ
NP-TMP
VP
DT NN NN NN JJ NN PP-LOC NP VBD
NONE lüks otomobil üreticisi geçen yıl
NP IN CD NNS sattı
NONE 1214 araba
DT NNP
NONE Birleşik Devletler'de

Figure 2: A pair of trees illustrating the heuristics given in 1, 2, 5, 6, 7, 8 and 11.

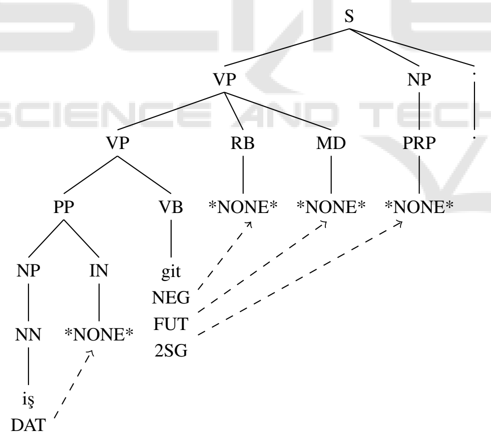


Figure 3: Movement of morphemes to empty slots in Phase 3.

difficulties. In some cases, they end up with morphemes for which they can not find empty *NONE* slots where they can move them. In yet other cases, they are left with too many *NONE* slots which they can not fill with suitable morphemes. Often, they realize that they had made a mistake when choosing a morphologically complex gloss for a phrase in Phase 1. Therefore, they go back and modify the translation in Phase 1.

3 APPLICATION TO TREE-BASED SMT

In order to demonstrate the use of our morphologically annotated parallel treebank in SMT, we trained and tested a model that learns to imitate the permutation and leaf replacement operations performed by human annotators in data generation. Our simple model consists of a distribution over the permutations for the children of an internal tree node and a distribution of Turkish words and morphemes for an English word. Next we describe each training step in detail.

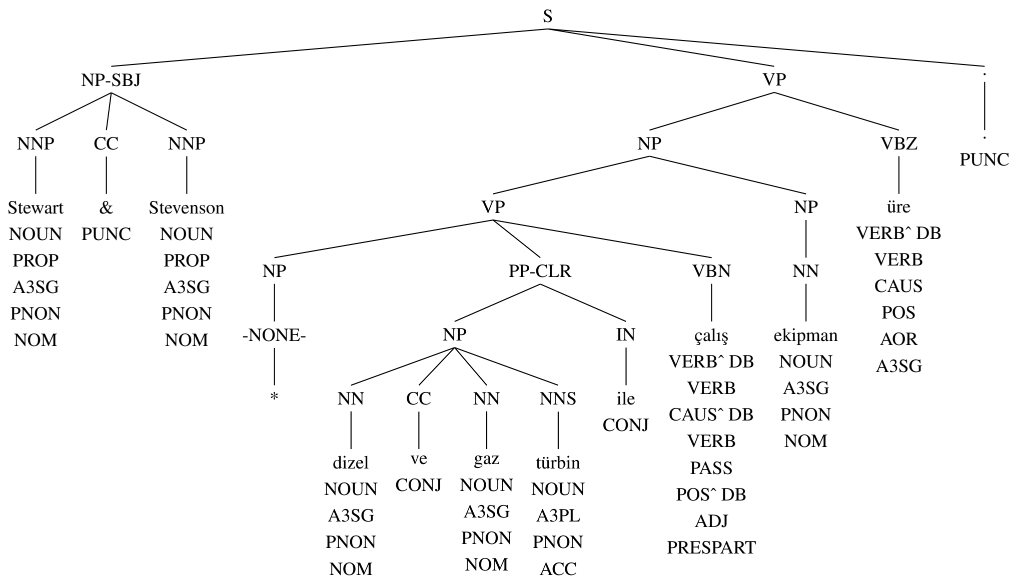
3.1 Permutation Training

Comparing parallel trees between Phase 0 and Phase-1-English, we keep counts of the permutations for each production rule in Phase 0. For example, comparing the trees in Figure 1, we identify the following non-trivial permutations

At the end of training, each rule is associated with a set of permutations and their counts. For each permutation π_i, we calculate its probability as

$$p_{\pi_i} = \frac{c_{\pi_i}}{\sum_i c_{\pi_i}},$$

where c_{π_i} denotes the number of times the permutation π_i is observed for the particular rule.



S
NP-SBJ
VP
.
NNP
CC
NNP
NP
VBZ
PUNC
Stewart NOUN PROP A3SG PNON NOM
& PUNC
Stevenson NOUN PROP A3SG PNON NOM
VP
NP
üre VERBˆ DB VERB CAUS POS AOR A3SG
NP
PP-CLR
VBN
NN
-NONE-
*
NP
IN
çalış VERBˆ DB VERB CAUSˆ DB VERB PASS POSˆ DB ADJ PRESPART
ekipman NOUN A3SG PNON NOM
NN
CC
NN
NNS
ile CONJ
dizel NOUN A3SG PNON NOM
ve CONJ
gaz NOUN A3SG PNON NOM
türbin NOUN A3PL PNON ACC

Figure 4: Morphological analyses of a permuted and leaf-replaced tree.

Table 1: Permutations for trees in Figure 1.

rule	permutation
VP→ VBD NP PP-LOC	(2,1,0)
PP-LOC→ IN NP	(1,0)
S→NP-SBJ NP-TMP VP	(0,1,2)
NP-SBJ→ DT NN NN NN	(0,1,2,3)
NP-TMP→ JJ NN	(0,1)
NP→CD NNS	(0,1)

Going over all the pairs of trees in Phase 1 data, we calculate the probabilities of all permutations of all the rules.

In testing, given a candidate permutation of English parse tree, we calculate its probability as the product of the probabilities for the permutations in all its subtrees. For example, the probability of the overall permutation from the English tree to its Turkish translation in Figure 1 is the product of the probabilities of permutations for the rules in Table 1.

3.2 Replacement Training

In order to assign probabilities to possible replacements of leaves in an English tree, we use a simple relative count of occurrences. So, for Turkish word t, and an English word e, the probability of t being a replacement of e is calculated as

$$p_e(t) = \frac{c_e(t)}{c_e},$$

where $c_e(t)$ denotes the the number of times t replaces e in the training data and c_e denotes the total count of e.

As training data, we use the leaves of the parallel trees in Phase-1-English and Phase-3-Canonical. So, not only do we generate probabilities for replacements between whole words, we also generate probabilities for replacing English function words with their canonical Turkish morphemes.

In order to calculate the probability of the replacement of the whole English sentence with a Turkish sentence of the same length, we simply multiply the probability of the individual replacements. Note that deleting an English word corresponds to replacing it with *NONE*. This replacement has a probability as well.

3.3 Combined Tree Translator

Once we have the probabilities of permutations and replacements, we can calculate the probability of a translation so obtained by just multiplying the two probabilities. Thus, for every translation candidate, we have an associated probability. We could just output the translation with the highest probability as the designated translation among all possible translations. However, generating all the translations is computationally expensive. Moreover, it ignores the constraints of language model statistics and morphotactics of the Turkish word formation. In this work, we ignore the language model mainly for the lack of a Turkish LM that incorporates statistics with morphotactics. Of course, using such model in ranking translation candidates would improve the results. We use our FST based morphological analyzer to eliminate impossible morpheme combinations.

For the initial exploration, we used a simplified search for the best translation. We first choose the most probable permuted tree and by reading off its leaves, we obtain a scrambled English sentence. Secondly, we trace this sentence left to right and replace words with Turkish stems and morphemes. For the initial word, we keep the list of its *N* most probable replacements. For the full list of candidate replacements of second word, we rank their combinations with the *N* candidates of the first word according to the product of their probabilities. We prune the new product list by keeping the first *N* combinations. Continuing like this, we maintain an *N*-best list of candidate partial translations.

Some of the candidate replacements are morphemes that need to combine with the previous stem or morpheme. So, while going over the candidate replacements, whenever we encounter a non-morpheme gloss, we check the morphotactical consistency of the partial translation up to but not including the current word. If it fails, we discard the candidate. For example, suppose a partial translation candidate up to the current candidate is “gel -PAST -NEG” and a candidate for the current word is the non-morpheme word “şimdi”. We currently check the consistency of “gel -PAST -NEG” and seeing that it fails, we drop the candidate “şimdi”.

4 EXPERIMENTS

We tested our translation approach using 10-fold cross-validation. We obtained BLEU scores for different *N*-best list sizes. Our corpus contains 5143 sentences. For each run, we use a random 90% of the them for training and the rest for testing. The mean scores and standard deviations for different list sizes are given in Table 2.

Table 2: BLEU scores for the best tree in the first step and different list sizes *N* in the second step.

N			
1	5	10	50
9.5±0.4	11.5±0.2	12.1±0.3	12.8±0.5

In the second set of experiments, we try to obtain the performance of the second step without the effect of the first step, namely, the tree permutation. In order to remove the effect of the first step, we start the second step of the translation with the optimal tree obtained in Phase-1-English (Section 2.2). In this way, if the second step is optimal, we would obtain a BLEU score of 100.

Table 3 shows BLEU scores for the overall translation when the optimal tree is used. The pattern of results with respect to *N* is similar to Table 2. On the other hand, the difference between the optimal tree and the best tree is not large. It seems that, we need to improve the leaf replacement step of our two-step algorithm more than its tree permutation step.

Table 3: BLEU scores for the optimal tree in the first step and different list sizes *N* in the second step.

N			
1	5	10	50
10.8±0.1	13.7±0.3	14.5±0.2	16.3±0.5

In the third set of experiments, we compared our translations with Google translate. For each sentence in the English test corpus, we compared its Google translation with our correct Turkish translation. We obtained a BLEU score of 11.6 ± 1.0. Despite having a larger variance, this score falls within the range $N = 5 - 10$ of our results. Of course, Google translate is a phrase-based engine that works with a general lexicon. Thus the comparison might be a bit too coarse. Still, we find it encouraging that even our simple translation approach with our rather limited corpus performs similar to a popular translation engine.

5 CONCLUSION

In this paper, we gave the details of our efforts in constructing a Turkish-English parallel Treebank and used the resulting data in a simple SMT algorithm. The BLEU scores for this initial attempt is not surprisingly low but compared to a general phrase-based tool like Google translate, it fares comparatively well.

A limitation of our parallel treebank construction is that it does not allow the detachment and attachment of subtrees. Even a literal translation is sometimes severely hindered under such a constraint. However, in our corpus, the overall feel of the sentence do not suffer much beyond sounding a bit too formal. Of course, a general tree based approach would have to construct the constituent trees independently in two languages and then train a learner to transform one tree to the other. Unfortunately there is no usable constituent parser for Turkish. Thus, we had to rely on leveraging the already available Penn Treebank and generate our Turkish trees starting from there. The flip side is that, this constrains our translation effort to only the English-to-Turkish direction.

An obvious challenge for any automatic translation task is the multiword expression and idioms.

Our translation model relies on a literal approach to sentence construction rather than a phrase based approach that can probably accommodate idiomatic expressions. Still in our parallel corpus, a single English word might be translated as more than one Turkish word. However, since this can happen only at the leaves, the Turkish leaf might need to be further expanded into a subtree. Instead, we left this problem as one of a translation lexicon where many-to-one and one-to-many mappings are possible.

When replacing the leaf tags in permuted trees with stems and morphemes, we usually move morphemes from other leaves. We observed some limitations of this approach in our three-phase annotation process. In the manual morphological analysis phase of Phase 2, the annotators use the context of the leaf word in choosing a word. At a later stage when another annotator works with the same leaf in Phase 3, sometimes the annotator realizes that, with the chosen analysis, the morpheme movements are not so easy. In these cases, the annotators usually go back to Phase 1 and change the replacement and have to the analysis again. This loop might benefit some form of automatic feedback across phases.

Exhaustive search over all of the translation trees to find the most probable permutation and replacement combination is computationally very expensive. Thus in this exploratory work, we chose the best permuted tree and searched for the best replacement independently. Even then, optimizing over all possible replacements of all the tokens in the sentence is prohibitively difficult. We had to rely on suboptimal heuristics in searching for the best replacement. However, a dynamic programming approach with early pruning of some branches using morphotactics at the subtrees might yield a faster search.

REFERENCES

El-Kahlout, I. D. (2009). Statistical machine translation from English to Turkish (Ph.D. thesis).

El-Kahlout, I. D. and Oflazer, K. (2006). Initial explorations in English to Turkish statistical machine translation. In *Proceedings of the Workshop on Statistical Machine Translation*, StatMT '06, pages 7–14, Stroudsburg, PA, USA. Association for Computational Linguistics.

Hutchinson, J. (1994). The Georgetown-IBM demonstration. *MT News International, no.8*, pages 15–18.

Jurafsky, D. and Martin, J. H. (2009). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition (Prentice Hall Series in Artificial Intelligence)*. Prentice Hall, 2 edition.

Koehn, P. (2010). *Statistical Machine Translation*. Cambridge University Press, New York, NY, USA, 1st edition.

Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Yeniterzi, R. and Oflazer, K. (2010). Syntax-to-morphology mapping in factored phrase-based statistical machine translation from English to Turkish. In *48th Annual Meeting of the Association for Computational Linguistics*.

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS