To appear in *HLT-NAACL 2003, Edmonton, Canada*

A Weighted Finite State Transducer Implementation of the Alignment Template Model for Statistical Machine Translation

Shankar Kumar and William Byrne
Center for Language and Speech Processing, Johns Hopkins University,
3400 North Charles Street, Baltimore, MD, 21218, USA
{skumar,byrne}@jhu.edu

Abstract

We present a derivation of the alignment template model for statistical machine translation and an implementation of the model using weighted finite state transducers. The approach we describe allows us to implement each constituent distribution of the model as a weighted finite state transducer or acceptor. We show that bitext word alignment and translation under the model can be performed with standard FSM operations involving these transducers. One of the benefits of using this framework is that it obviates the need to develop specialized search procedures, even for the generation of lattices or N-Best lists of bitext word alignments and translation hypotheses. We evaluate the implementation of the model on the French-to-English Hansards task and report alignment and translation performance.

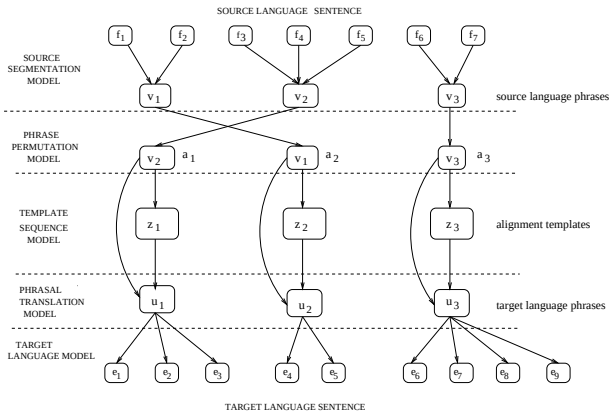

Figure 1: ATTM Architecture.

1 Introduction

The Alignment Template Translation Model (ATTM) (Och et al., 1999) has emerged as a promising modeling framework for statistical machine translation. The ATTM attempts to overcome the deficiencies of word-to-word translation models (Brown et al., 1993) through the use of phrasal translations. The overall model is based on a two-level alignment between the source and the target sentence: a phrase-level alignment between source and target phrases and a word-level alignment between words in these phrase pairs.

The goal of this paper is to reformulate the ATTM so that the operations we intend to perform under a statistical translation model, namely bitext word alignment and translation, can be implementation using standard weighted finite state transducer (WFST) operations. Our main motivation for a WFST modeling framework lies in the resulting simplicity of alignment and translation processes compared to dynamic programming or A^* decoders. The WFST implementation allows us to use standard optimized algorithms available from an off-the-shelf FSM toolkit (Mohri et al., 1997). This avoids the need to develop specialized search procedures, even for the generation of lattices or N-best lists of bitext word alignment or translation hypotheses.

Weighted Finite State Transducers for Statistical Machine Translation (SMT) have been proposed in the literature to implement word-to-word translation models (Knight and Al-Onaizan, 1998) or to perform translation in an application domain such as the call routing task (Bangalore and Ricardi, 2001). One of the objectives of these approaches has been to provide an implementation for SMT that uses standard FSM algorithms to perform model computations and therefore make SMT techniques accessible to a wider community. Our WFST implementation of the ATTM has been developed with similar objectives.

We start off by presenting a derivation of the ATTM that identifies the conditional independence assumptions that underly the model. The derivation allows us to specify each component distribution of the model and implement it as a weighted finite state transducer. We then show that bitext word alignment and translation can be performed with standard FSM operations involving these transducers. Finally we report bitext word alignment and translation performance of the implementation on the Canadian French-to-English Hansards task.

2 Alignment Template Translation Models

We present here a derivation of the alignment template translation model (ATTM) (Och et al., 1999; Och, 2002) and give an implementation of the model using weighted finite state transducers (WFSTs). The finite state modeling is performed using the AT&T FSM Toolkit (Mohri et al., 1997).

In this model, the translation of a source language sentence to a target language sentence is described by a joint probability distribution over all possible segmentations and alignments. This distribution is presented in Figure 1 and Equations 1-7. The components of the overall translation model are the source language model (Term 2), the source segmentation model (Term 3), the phrase permutation model (Term 4), the template sequence model (Term 5), the phrasal translation model (Term 6) and the target language model (Term 7). Each of these conditional distributions is modeled independently and we now define each in turn and present its implementation as a weighted finite state acceptor or transducer.

$$P(e_1^I, u_1^K, z_1^K, a_1^K, v_1^K, K, f_1^J) = \quad (1)$$
$$P(f_1^J) \cdot \quad (2)$$
$$P(v_1^K, K|f_1^J) \cdot \quad (3)$$
$$P(a_1^K|v_1^K, K, f_1^J) \cdot \quad (4)$$
$$P(z_1^K|a_1^K, v_1^K, K, f_1^J) \cdot \quad (5)$$
$$P(u_1^K|z_1^K, a_1^K, v_1^K, K, f_1^J) \cdot \quad (6)$$
$$P(e_1^I|u_1^K, z_1^K, a_1^K, v_1^K, K, f_1^J) \quad (7)$$

We begin by distinguishing words and phrases. We assume that u is a phrase in the target language sentence that has length M and consists of words $e_1, e_2, ..., e_M$. Similarly, a phrase v in the source language sentence contains words $f_0, f_1, ..., f_N$, where f_0 is the NULL token. We assume that each word in each language can be assigned to a unique class so that u unambiguously specifies a class sequence E_1^M and v specifies the class sequence F_0^N. Throughout the model, if a sentence f_1^J is segmented into phrases v_1^K, we say $v_1^K = f_1^J$ to indicate that the words in the phrase sequence agree with the original sentence.

Source Language Model The model assigns probability to any sentence f_1^J in the source language; this probability is not actually needed by the translation process when f_1^J is given. As the first component in the model, a finite state acceptor S is constructed for f_1^J.

Source Segmentation Model We introduce the phrase count random variable K which specifies the number of phrases in a particular segmentation of the source language sentence. For a sentence of length J, there are $\binom{J-1}{K-1}$ ways to segment it into K phrases. Motivated by this, we choose the distribution $P(K|f_1^J)$ as

$$P(K|f_1^J) = \frac{\binom{J-1}{K-1}}{2^{J-1}}; K \in \{1, 2, ..., J\}, \quad (8)$$

so that $\sum_K P(K|f_1^J) = 1$.

We construct a joint distribution over all phrase segmentations $v_1^K = v_1, v_2, ..., v_K$ as

$$P(v_1^K, K|f_1^J) = P(v_1^K|K, f_1^J)P(K|f_1^J) \quad (9)$$

where

$$P(v_1^K|K, f_1^J) = \begin{cases} \frac{1}{Z_K}\prod_{k=1}^K p_u(v_k) & v_1^K = f_1^J \\ 0 & otherwise \end{cases}.$$

The normalization constant $Z_K = \sum_{\bar{v}_1^K}\prod_{k=1}^K p_u(\bar{v}_k)$, is chosen so that $\sum_{K, v_1^K} P(v_1^K, K|f_1^J) = 1$.

Here, $p_u(v_k)$ is a "unigram" distribution over source language phrases; we assume that we have an inventory of phrases from which this quantity can be estimated. In this way, the likelihood of a particular segmentation is determined by the likelihood of the phrases that result.

We now describe the finite state implementation of the source segmentation model and show how to compute the most likely segmentation under the model:
$\{\hat{v}_1^{\hat{K}}, \hat{K}\} = \text{argmax}_{v_1^K, K} P(v_1^K|K, f_1^J)P(K|f_1^J)$.

1. For each source language sentence f_1^J to be translated, we implement a weighted finite state transducer Ω that segments the sentence into all possible phrase sequences v_1^K permissible given the inventory of phrases. The score of a segmentation v_1^K under Ω is given by $\prod_{k=1}^K p_u(v_k)$. We then generate a lattice of segmentations of f_1^J (implemented as an acceptor S) by composing it with the transducer Ω, i.e. $\mathcal{V} = S \circ \Omega$.

2. We then decompose $\mathcal{V}$ into J disjoint subsets $\mathcal{V}_K; K \in \{1, 2, ..., J\}, \cup_{K=1}^J \mathcal{V}_K = \mathcal{V}$ so that $\mathcal{V}_K$ contains all segmentations of the source language sentence with exactly K phrases. To construct $\mathcal{V}_K$, we create an unweighted acceptor P_K that accepts any phrase sequence of length K; for efficiency, the phrase vocabulary is restricted to the phrases in $\mathcal{V}$. $\mathcal{V}_K$ is then obtained by the finite state composition $\mathcal{V}_K = \mathcal{V} \circ P_K$.

3. For $K = 1, 2, ..., J$
   The normalization factors Z_K are obtained by summing the probabilities of all segmentations in $\mathcal{V}_K$. This sum can be computed efficiently using lattice forward probabilities (Wessel et al., 1998). For a fixed K, the most likely segmentation in $\mathcal{V}_K$ is found as

$$\hat{V}_K = \underset{V' \in \mathcal{V}_K}{\text{argmax}} \frac{1}{Z_K}\prod_{k=1}^K P_u(v'_k). \quad (10)$$

4. Finally we select the optimal segmentation as

$$\hat{V} = \underset{K \in \{1,2,...,J\}}{\text{argmax}} P(\hat{V}_K|K, f_1^J)P(K|f_1^J). \quad (11)$$

A portion of the segmentation transducer Ω for the French sentence *nous avons une inflation galopante* is presented in Figure 2. When composed with S, Ω generates the following two phrase segmentations: *nous avons une_inflation_galopante* and *nous_avons_une inflation_galopante*. The "_" symbol is used to indicate phrases formed by concatenation of consecutive words. The phrases specified by the source segmentation model remain in the order that they appear in the source sentence.

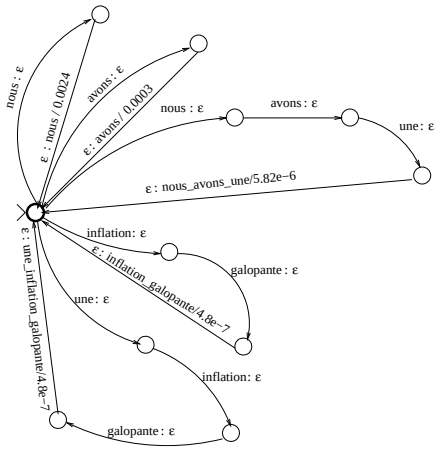

Figure 2: A portion of the phrase segmentation transducer Ω for the sentence "nous avons une inflation galopante".

Phrase Permutation Model We now define a model for the reordering of phrase sequences as determined by the previous model. The phrase alignment sequence a_1^K specifies a reordering of phrases into target language phrase order; the words within the phrases remain in the source language order. The phrase sequence v_1^K is reordered into $v_{a_1}, v_{a_2}, ..., v_{a_K}$. The phrase alignment sequence is modeled as a first order Markov process

$$\begin{aligned} P(a_1^K|v_1^K, K, f_1^J) &= P(a_1^K|v_1^K) \quad (12) \\ &= P(a_1)\prod_{k=2}^{K} P(a_k|a_{k-1}, v_1^K). \end{aligned}$$

with $a_k \in \{1, 2, ..., K\}$. The alignment sequence distribution is constructed to assign decreasing likelihood to phrase re-orderings that diverge from the original word order. Suppose $v_{a_k} = f_l^{l'}$ and $v_{a_{k-1}} = f_m^{m'}$, we set the Markov chain probabilities as follows (Och et al., 1999)

$$\begin{aligned} P(a_k|a_{k-1}) &\propto p_0^{|l-m'-1|} \\ P(a_1 = k) &= \frac{1}{K}; k \in \{1, 2, ..., K\}. \quad (13) \end{aligned}$$

In the above equations, p_0 is a tuning factor and we normalize the probabilities $P(a_k|a_{k-1})$ so that $\sum_{j=1, j\neq a_{k-1}}^{K} P(a_k = j|a_{k-1}) = 1$.

The finite state implementation of this model involves two acceptors. We first build a unweighted permutation acceptor Π_V that contains all permutations of the phrase sequence v_1^K in the source language (Knight and Al-Onaizan, 1998) . We note that a path through Π_V corresponds to an alignment sequence a_1^K. Figure 3 shows the acceptor Π_V for the source phrase sequence *nous avons une_inflation_galopante*.

A source phrase sequence V of length K words requires a permutation acceptor Π_V of 2^K states. For long phrase sequences we compute a score $\max_j P(a_k = i|a_{k-1} = j)$ for each arc and then prune the arcs by this score, i.e. phrase alignments containing $a_k = i$ are included only if this score is above a threshold. Pruning can therefore be applied while Π_V is constructed.

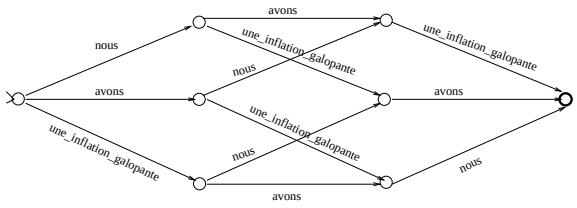

Figure 3: The permutation acceptor Π_V for the source-language phrase sequence *nous avons une_inflation_galopante*.

The second acceptor H in the implementation of the phrase permutation model assigns alignment probabilities (Equation 13) to a given permutation a_1^K of the source phrase sequence v_1^K (Figure 4). In this example, the phrases in the source phrase sequence are specified as follows: $v_1 = f_1$ (*nous*), $v_2 = f_2$ (*avons*) and $v_3 = f_3^5$ (*une_inflation_galopante*). We now show the computation of some of the alignment probabilities (Equation 13) in this example ($p_0 = 0.9$)

$$\begin{aligned} P(a_3 = 1|a_2 = 3) \propto \quad p_0^{|1-5-1|} = \quad 0.59 \\ P(a_3 = 2|a_2 = 3) \propto \quad p_0^{|2-5-1|} = \quad 0.66. \end{aligned}$$

Normalizing these terms gives $P(a_3 = 1|a_2 = 3) = 0.47$ and $P(a_3 = 2|a_2 = 3) = 0.53$.

Template Sequence Model Here we describe the main component of the model. An alignment template $z = (E_1^M, F_0^N, A)$ specifies the allowable alignments between the class sequences E_1^M and F_0^N. A is a $M \times (N + 1)$ binary, 0/1 valued matrix which is constructed as follows: If E_i can be aligned to F_j, then $A_{ij} = 1$; otherwise $A_{ij} = 0$. This process may allow E_i to align with the NULL token F_0, i.e. $A_{i0} = 1$, so that words can be freely inserted in translation. Given a pair of class sequences E_1^M and F_0^N, we specify exactly one matrix A.

We say that $z = (E_1^M, F_0^N, A)$ is *consistent* with the target language phrase u and the source language phrase

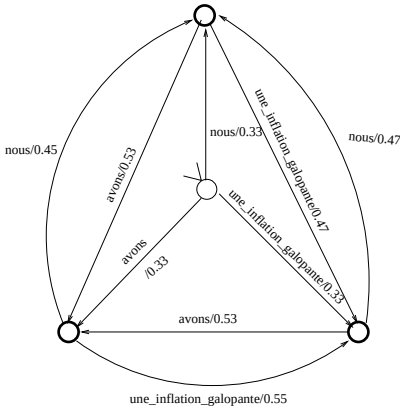

Figure 4: Acceptor H that assigns probabilities to permutations of the source language phrase sequence *nous avons une_inflation_galopante* ($p_0 = 0.9$).

v if E_1^M is the class sequence for u and F_0^N is the class sequence for v.

In Section 4.1, we will outline a procedure to build a library of alignment templates from bitext word-level alignments. Each template $z = (E_1^M, F_0^N, A)$ used in our model has an index i in this template library. Therefore any operation that involves a mapping to (from) template sequences will be implemented as a mapping to (from) a sequence of these indices.

We have described the segmentation and permutation processes that transform a source language sentence into phrases in target language phrase order. The next step is to generate a consistent sequence of alignment templates. We assume that the templates are conditionally independent of each other and depend only on the source language phrase which generated each of them

$$\begin{aligned} P(z_1^K|a_1^K, v_1^K, K, f_1^J) &= \prod_{k=1}^{K} P(z_k|a_1^K, v_1^K, K, f_1^J) \\ &= \prod_{k=1}^{K} P(z_k|v_{a_k}). \end{aligned} \quad (14)$$

We will implement this model using the transducer Y that maps any permutation $v_{a_1}, v_{a_2}, ..., v_{a_K}$ of the phrase sequence v_1^K into a template sequence z_1^K with probability as in Equation 14. For every phrase v, this transducer allows only the templates z that are consistent with v with probability $P(z|v)$, i.e. $P(z_k|v_{a_k})$ enforces the consistency between each source phrase and alignment template.

Phrasal Translation Model We assume that a target phrase is generated independently by each alignment template and source phrase

$$\begin{aligned} & P(u_1^K|z_1^K, a_1^K, v_1^K, K, f_1^J) \\ &= \prod_{k=1}^{K} P(u_k|z_1^K, a_1^K, v_1^K, K, f_1^J) \\ &= \prod_{k=1}^{K} P(u_k|z_k, v_{a_k}). \end{aligned} \quad (15)$$

This allows us to describe the *phrase-internal translation model* $P(u|v, z)$ as follows. We assume that each word in the target phrase is produced independently and that the consistency is enforced between the words in u and the class sequence E_1^M so that $P(e_i|z, v) = 0$ if $e_i \notin E_i$.

We now introduce the word alignment variables $\phi_i, i = 1, 2, ..., M$, which indicates that e_i is aligned to f_{ϕ_i} within u and v.

$$\begin{aligned} & P(u|z = (E_1^M, F_0^N, A), v) \\ &= \prod_{i=1}^{M} P(e_i|z, v) = \prod_{i=1}^{M} \sum_{j=0}^{N} P(e_i, \phi_i = j|z, v) \\ &= \prod_{i=1}^{M} \sum_{j=0}^{N} P(e_i|\phi_i = j, z, v) P(\phi_i = j|A, v) \\ &= \prod_{i=1}^{M} \sum_{j=0}^{N} P(e_i|f_j) P(\phi_i = j|A) 1_{E_i}(e_i). \end{aligned} \quad (16)$$

The term $P(e_i|f_j)$ is a translation dictionary (Och and Ney, 2000) and $P(\phi_i = j, A)$ is obtained as

$$P(\phi_i = j|A) = \frac{A_{ij}}{\sum_{j'} A_{ij'}}. \quad (17)$$

We have assumed that $P(\phi_i|v, A) = P(\phi_i|A)$, i.e. that given the template, word alignments do not depend on the source language phrase.

For a given phrase v and a consistent alignment template $z = (E_1^M, F_0^N, A)$, a weighted acceptor Z can be constructed to assign probability to translated phrases according to Equations 16 and 17. Z is constructed from four component machines O, I, D and C, constructed as follows.

The first acceptor O implements the alignment matrix A. It has $M+1$ states and between any pair of states $i-1$ and i, each arc j corresponds to a word alignment variable $\phi_i = j$. Therefore the number of transitions between states i and $i+1$ is equal to the number of non-zero values of ϕ_i. The j^{th} arc from state $i-1$ to i has probability $P(\phi_i = j|A)$ (Equation 17).

The second machine I is an unweighted transducer that maps the index $i \in \{0, 1, ..., N\}$ in the phrase $v = f_0^N$ to the corresponding word f_i.

The third transducer is the lexicon transducer D that maps the source word $f \in V_F$ to the target word $e \in V_E$ with probability $P(e|f)$.

The fourth acceptor C is unweighted and allows all target word sequences e_1^M which can be specified by the

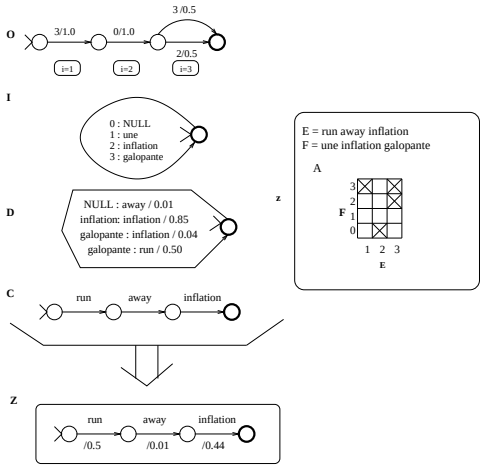

Figure 5: Component transducers to construct the acceptor Z for an alignment template z.

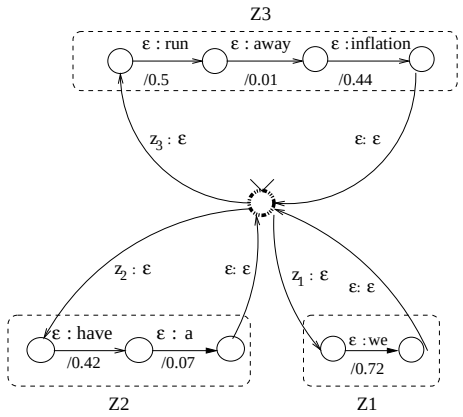

Figure 6: Transducer W that maps the source template sequence z_1^3 into target phrase sequences u_1^3.

class sequence E_1^M. C has $M+1$ states. The number of transitions between states $i-1$ and i is equal to the number of target language words with class specified by E_i.

Figure 5 shows all the four component FSTs for building the transducer Z corresponding to an alignment template from our library. Having built these four machines, we obtain Z as follows. We first compose the four transducers, project the resulting transducer onto the output labels, and determinize it under the $(+, \times)$ semiring. This is implemented using AT&T FSM tools as follows

fsmcompose O I D C | *fsmproject* -o |\
fsmrmepsilon | *fsmdeterminize* $> Z$.

Given an alignment template z and a consistent source phrase v, we note that the composition and determinization operations assign the probability $P(u|z,v)$ (Equation 16) to each consistent target phrase u. This summarizes the construction of a transducer for a single alignment template.

We now implement a transducer W that maps sequences of alignment templates to target language word sequences. We identify all templates consistent with the phrases in the source language phrase sequence v_1^K. The transducer W is constructed via the FSM union operation of the transducers that implement these templates.

For the source phrase sequence v_1^3 (*nous avons une_inflation_galopante*), we show the transducer W in Figure 6. Our example library consists of three templates z_1, z_2 and z_3. z_1 maps the source word *nous* to the target word *we* via the word alignment matrix A specified as $\phi_1 = 1$. z_2 maps the source word *avons* to the target phrase *have a* via the word alignment matrix A specified as $\phi_1 = 1, \phi_2 = 0$. z_3 maps the source phrase *une_inflation_galopante* to the target phrase *run_away_inflation* via the word alignment matrix A specified as $\phi_1 = 3, \phi_2 = 0, \phi_3 = \{2, 3\}$.

W is built out of the three component acceptors Z_1, Z_2, and Z_3. The acceptor Z_i corresponds to the mapping from the template z_i and the source phrase v_i to all consistent target phrases u_i.

Target Language Model We specify this model as

$$P(e_1^I|u_1^K, z_1^K, a_1^K, v_1^K, K, f_1^J) = P_E(e_1^I)1\{e_1^I = u_1^K\},$$

where $1\{e_1^I = u_1^K\}$ enforces the requirement that words in the translation agree with those in the phrase sequence. We note that $P_E(e_1^I)$ is modeled as a standard backoff trigram language model (Stolcke, 2002). Such a language model can be easily compiled as a weighted finite state acceptor (Mohri et al., 2002).

3 Alignment and Translation Via WFSTs

We will now describe how the alignment template translation model can be used to perform word-level alignment of bitexts and translation of source language sentences.

Given a source language sentence f_1^J and a target sentence e_1^I, the word-to-word alignment between the sentences can be found as

$$\{\hat{u}_1^{\hat{K}}, \hat{z}_1^{\hat{K}}, \hat{a}_1^{\hat{K}}, \hat{v}_1^{\hat{K}}, \hat{K}\} = \operatorname*{argmax}_{u_1^K, z_1^K, a_1^K, v_1^K, K} P(u_1^K, z_1^K, a_1^K, v_1^K, K|e_1^I, f_1^J).$$

The variables $\{\hat{u}_1^{\hat{K}}, \hat{a}_1^{\hat{K}}, \hat{v}_1^{\hat{K}}, \hat{K}\}$ specify the alignment between source phrases and target phrases while $\hat{z}_1^{\hat{K}}$ gives the word-to-word alignment within the phrase sequences.

Given a source language sentence f_1^J, the translation can be found as

$$\{\hat{e}_1^{\hat{I}}, \hat{u}_1^{\hat{K}}, \hat{z}_1^{\hat{K}}, \hat{a}_1^{\hat{K}}, \hat{v}_1^{\hat{K}}, \hat{K}\} = \operatorname*{argmax}_{e_1^I, u_1^K, z_1^K, a_1^K, v_1^K, K} P(e_1^I, u_1^K, z_1^K, a_1^K, v_1^K, K|f_1^J),$$

where $\hat{e}_1^I$ is the translation of f_1^J.

We implement the alignment and translation procedures in two steps. We first segment the source sentence into phrases, as described earlier

$$\{\hat{v}_1^{\hat{K}}, \hat{K}\} = \underset{v_1^K, K}{\operatorname{argmax}} P(v_1^K|K, f_1^J)P(K|f_1^J). \quad (18)$$

After segmenting the source sentence, the alignment of a sentence pair (e_1^I, f_1^J) is obtained as

$$\{\hat{u}_1^{\hat{K}}, \hat{z}_1^{\hat{K}}, \hat{a}_1^{\hat{K}}\} = \quad (19)$$
$$\underset{u_1^{\hat{K}}, z_1^{\hat{K}}, a_1^{\hat{K}}}{\operatorname{argmax}} P(u_1^{\hat{K}}, z_1^{\hat{K}}, a_1^{\hat{K}}|\hat{v}_1^{\hat{K}}, \hat{K}, f_1^J, e_1^I).$$

The translation is the same way as

$$\{\hat{e}_1^{\hat{I}}, \hat{u}_1^{\hat{K}}, \hat{z}_1^{\hat{K}}, \hat{a}_1^{\hat{K}}\} = \quad (20)$$
$$\underset{e_1^I, u_1^{\hat{K}}, z_1^{\hat{K}}, a_1^{\hat{K}}}{\operatorname{argmax}} P(e_1^I, u_1^{\hat{K}}, z_1^{\hat{K}}, a_1^{\hat{K}}|\hat{v}_1^{\hat{K}}, \hat{K}, f_1^J).$$

We have described how to compute the optimal segmentation $\hat{V} = \hat{v}_1^{\hat{K}}$ (Equation 18) in Section 2. The segmentation process decomposes the source sentence f_1^J into a phrase sequence $\hat{v}_1^{\hat{K}}$. This process also tags each source phrase $\hat{v}_k$ with its position k in the phrase sequence. We will now describe the alignment and translation processes using finite state operations.

3.1 Bitext Word Alignment

Given a collection of alignment templates, it is not guaranteed that every sentence pair in a bitext can be segmented into phrases for which there exist the consistent alignment templates needed to create an alignment between the sentences. We find in practice that this problem arises frequently enough that most sentence pairs are assigned a probability of zero under the template model. To overcome this limitation, we add several types of "dummy" templates to the library that serve to align phrases when consistent templates could not otherwise be found.

The first type of dummy template we introduce allows any source phrase $\hat{v}_k$ to align with any single word target phrase u_i. This template is defined as a triple $z_{ik} = \{u_i, \hat{v}_k, A\}$ where $k \in \{1, 2, ..., \hat{K}\}$ and $i \in \{1, 2, ..., I\}$. All the entries of the matrix A are specified to be ones. The second type of dummy template allows source phrases to be deleted during the alignment process. For a source phrase $\hat{v}_k$ we specify this template as $z_k = (\hat{v}_k, \epsilon), k = 1, 2, ..., \hat{K}$. The third type of template allows for insertions of single word target phrases. For a target phrase u_i we specify this template as $z_i = (\epsilon, u_i), i = 1, 2, ..., I$. The probabilities $P(z|v)$ for these added templates are not estimated; they are fixed as a global constant which is set so as to discourage their use except when no other suitable templates are available.

A lattice of possible alignments between e_1^I and f_1^J is then obtained by the finite state composition

$$\mathcal{B} = \Pi_{\hat{V}} \circ H \circ Y \circ W \circ T. \quad (21)$$

where T is an acceptor for the target sentence e_1^I. We then compute the ML alignment $\hat{B}$ (Equation 19) by obtaining the path with the highest probability, in $\mathcal{B}$. The path $\hat{B}$ determines three types of alignments: phrasal alignment between the source phrase $\hat{v}_k$ and the target phrase $\hat{u}_k$; deletions of source phrases $\hat{v}_k$; and insertions of target words e_i. To determine the word-level alignment between the sentences e_1^I and f_1^J,we are primarily interested in the first of these types of alignments. Given that the source phrase $\hat{v}_k$ has aligned to the target phrase $\hat{u}_k$, we look up the hidden template variable $\hat{z}_k$ that yielded this alignment. $\hat{z}_k$ contains the the word-to-word alignment between these phrases.

3.2 Translation and Translation Lattices

The lattice of possible translations of f_1^J is obtained using the weighted finite state composition:

$$\mathcal{T} = \Pi_{\hat{V}} \circ H \circ Y \circ W \circ G. \quad (22)$$

The translation with the highest probability (Equation 20) can now be computed by obtaining the path with the highest score in $\mathcal{T}$.

In terms of AT&T FSM tools, this can be done as follows

fsmbestpath $\mathcal{T}$ | *fsmproject* $-o$ | \
fsmrmepsilon $> \hat{T}$

A translation lattice (Ueffing et al., 2002) can be generated by pruning $\mathcal{T}$ based on likelihoods or number of states. Similarly, an alignment lattice can be generated by pruning $\mathcal{B}$.

4 Translation and Alignment Experiments

We now evaluate this implementation of the alignment template translation model.

4.1 Building the Alignment Template Library

To create the template library, we follow the procedure reported in Och (2002). We first obtain word alignments of bitext using IBM-4 translation models trained in each translation direction (IBM-4 F and IBM-4 E), and then forming the union of these alignments (IBM-4 $F \cup E$). We extract the library of alignment templates from the bitext alignment using the *phrase-extract* algorithm reported in Och (2002). This procedure identifies several alignment templates $z = (E_1^M, F_0^N, A)$ that are consistent with a source phrase v. We do not use word classes in the experiments reported here; therefore templates are specified by phrases rather than by class sequences. For a given pair of source and target phrases, we retain only

the matrix of alignments that occurs most frequently in the training corpus. This is consistent with the intended application of these templates for translation and alignment under the maximum likelihood criterion; in the current formulation, only one alignment will survive in any application of the models and there is no reason to retain any of the less frequently occuring alignments. We estimate the probability $P(z|v)$ by the relative frequency of phrasal translations found in bitext alignments. To restrict the memory requirements of the model, we extract only the templates which have at most 5 words in the source phrase. Furthermore, we restrict ourselves to the templates which have a probability $P(z|v) > 0.01$ for some source phrase v.

4.2 Bitext Word Alignment

We present results on the French-to-English Hansards translation task (Och and Ney, 2000). We measured the alignment performance using precision, recall, and Alignment Error Rate (AER) metrics (Och and Ney, 2000).

Our training set is a subset of the Canadian Hansards which consists of $50,000$ French-English sentence pairs (Och and Ney, 2000). The English side of the bitext had a total of $743,633$ words ($18,430$ unique tokens) and the French side contained $816,545$ words ($24,096$ unique tokens). Our template library consisted of $1,071,128$ templates.

Our test set consists of 500 unseen French sentences from Hansards for which both reference translations and word alignments are available (Och and Ney, 2000). We present the results under the ATTM in Table 1, where we distinguish word alignments produced by the templates from the template library against those produced by the templates introduced for alignment in Section 3.1. For comparison, we also align the bitext using IBM-4 translation models.

Model	Alignment Metrics (%)		
	Precision	Recall	AER
IBM-4 F	88.9	89.8	10.8
IBM-4 E	89.2	89.4	10.7
IBM-4 $F \cup E$	84.3	93.8	12.3
ATTM-C	64.2	63.8	36.2
ATTM-A	94.5	55.8	27.3

Table 1: Alignment Performance on the French-to-English Hansards Alignment Task.

We first observe that the complete set of word alignments generated by the ATTM (ATTM-C) is relatively poor. However, when we consider only those word alignments generated by actual alignment templates (ATTM-A) (and discard the alignments generated by the dummy templates introduced as described in Section 3.1), we obtain very high alignment precision. This implies that word alignments within the templates are very accurate. However, the poor performance under the recall measure suggests that the alignment template library has relatively poor coverage of the phrases in the alignment test set.

4.3 Translation and Lattice Quality

We next measured the translation performance of ATTM on the same test set. The translation performance was measured using the BLEU (Papineni et al., 2001) and the NIST MT-eval metrics (Doddington, 2002), and Word Error Rate (WER). The target language model was a trigram language model with modified Kneser-Ney smoothing trained on the English side of the bitext using the SRILM tookit (Stolcke, 2002). The performance of the model is reported in Table 2. For comparison, we also report performance of the IBM-4 translation model trained on the same corpus. The IBM Model-4 translations were obtained using the ReWrite decoder (Marcu and Germann, 2002). The results in Table 2 show that the alignment template model outperforms the IBM Model 4 under all three metrics. This verifies that WFST implementation of the ATTM can obtain a performance that compares favorably to other well known research tools.

Model	BLEU	NIST	WER (%)
IBM-4	0.1711	5.0823	67.5
ATTM	0.1941	5.3337	64.7

Table 2: Translation Performance on the French-to-English Hansards Translation Task.

We generate N-best lists from each translation lattice, and show the variation of their oracle-best BLEU scores in Table 3. We observe that the oracle-best BLEU score increases with the size of the N-Best List. We can therefore expect to rescore these lattices with more sophisticated models and achieve improvements in translation quality.

	Size of N-best list				
	1	10	100	400	1000
BLEU	0.1941	0.2264	0.2550	0.2657	0.2735

Table 3: Variation of oracle-Best BLEU scores on N-Best lists generated by the ATTM.

5 Discussion

The main motivation for our investigation into this WFST modeling framework for statistical machine translation lies in the simplicity of the alignment and translation processes relative to other dynamic programming or A^* decoders (Och, 2002). Once the components of the alignment template translation model are implemented as WFSTs, alignment and translation can be performed using

standard FSM operations that have already been implemented and optimized. It is not necessary to develop specialized search procedures, even for the generation of lattices and N-best lists of alignment and translation alternatives.

The derivation of the ATTM was presented with the intent of clearly identifying the conditional independence assumptions that underly the WFST implementation. This approach leads to modular implementations of the component distributions of the translation model. These components can be refined and improved by changing the corresponding transducers without requiring changes to the overall search procedure. However some of the modeling assumptions are extremely strong. We note in particular that segmentation and translation are carried out independently in that phrase segmentation is followed by phrasal translation; performing these steps independently can easily lead to search errors.

It is a strength of the ATTM that it can be directly constructed from available bitext word alignments. However this construction should only be considered an initialization of the ATTM model parameters. Alignment and translation can be expected to improve as the model is refined and in future work we will investigate iterative parameter estimation procedures.

We have presented a novel approach to generate alignments and alignment lattices under the ATTM. These lattices will likely be very helpful in developing ATTM parameter estimation procedures, in that they can be used to provide conditional distributions over the latent model variables. We have observed that that poor coverage of the test set by the template library may be why the overall word alignments produced by the ATTM are relatively poor; we will therefore also explore new strategies for template selection.

The alignment template model is a powerful modeling framework for statistical machine translation. It is our goal to improve its performance through new training procedures while refining the basic WFST architecture.

Acknowledgments

We would like to thank F. J. Och of ISI, USC for providing us the GIZA++ SMT toolkit, the *mkcls* toolkit to train word classes, the Hansards 50K training and test data, and the reference word alignments and *AER* metric software. We thank AT&T Labs - Research for use of the FSM Toolkit and Andreas Stolcke for use of the SRILM Toolkit. This work was supported by an ONR MURI grant N00014-01-1-0685.

References

S. Bangalore and G. Ricardi. 2001. A finite-state approach to machine translation. In *Proc. of the North American Chapter of the Association for Computational Linguistics*, Pittsburgh, PA, USA.

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

G. Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proc. of HLT 2002*, San Diego, CA. USA.

K. Knight and Y. Al-Onaizan. 1998. Translation with finite-state devices. In *Proc. of the AMTA Conference*, pages 421–437, Langhorne, PA, USA.

D. Marcu and U. Germann, 2002. *The ISI ReWrite Decoder Release 0.7.0b*. http://www.isi.edu/licensed-sw/rewrite-decoder/.

M. Mohri, F. Pereira, and M. Riley, 1997. *ATT General-purpose finite-state machine software tools*. http://www.research.att.com/sw/tools/fsm/.

M. Mohri, F. Pereira, and M. Riley. 2002. Weighted finite-state transducers in speech recognition. *Computer Speech and Language*, 16(1):69–88.

F. Och and H. Ney. 2000. Improved statistical alignment models. In *Proc. of ACL-2000*, pages 440–447, Hong Kong, China.

F. Och, C. Tillmann, and H. Ney. 1999. Improved alignment models for statistical machine translation. In *Proc. of the Joint Conf. of Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 20–28, College Park, MD, USA.

F. Och. 2002. *Statistical Machine Translation: From Single Word Models to Alignment Templates*. Ph.D. thesis, RWTH Aachen, Germany.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2001. Bleu: a method for automatic evaluation of machine translation. Technical Report RC22176 (W0109-022), IBM Research Division.

A. Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *Proc. of the International Conference on Spoken Language Processing*, pages 901–904, Denver, CO, USA. http://www.speech.sri.com/projects/srilm/.

N. Ueffing, F. Och, and H. Ney. 2002. Generation of word graphs in statistical machine translation. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pages 156–163, Philadelphia, PA, USA.

F. Wessel, K. Macherey, and R. Schlueter. 1998. Using word probabilities as confidence measures. In *Proc. of ICASSP-98*, pages 225–228, Seattle, WA, USA.