Inducing Sentence Structure from Parallel Corpora for Reordering

John DeNero
Google Research
denero@google.com

Jakob Uszkoreit
Google Research
uszkoreit@google.com

Abstract

When translating among languages that differ substantially in word order, machine translation (MT) systems benefit from syntactic pre-ordering—an approach that uses features from a syntactic parse to permute source words into a target-language-like order. This paper presents a method for inducing parse trees automatically from a parallel corpus, instead of using a supervised parser trained on a treebank. These induced parses are used to pre-order source sentences. We demonstrate that our induced parser is effective: it not only improves a state-of-the-art phrase-based system with integrated reordering, but also approaches the performance of a recent pre-ordering method based on a supervised parser. These results show that the syntactic structure which is relevant to MT pre-ordering can be learned automatically from parallel text, thus establishing a new application for unsupervised grammar induction.

1 Introduction

Recent work in statistical machine translation (MT) has demonstrated the effectiveness of *syntactic pre-ordering*: an approach that permutes source sentences into a target-like order as a pre-processing step, using features of a source-side syntactic parse (Collins et al., 2005; Xu et al., 2009). Syntactic pre-ordering is particularly effective at applying structural transformations, such as the ordering change from a subject-verb-object (SVO) language like English to a subject-object-verb (SOV) language like Japanese. However, state-of-the-art pre-ordering methods require a supervised syntactic parser to provide structural information about each sentence. We propose a method that learns both a parsing model and a reordering model directly from a word-aligned parallel corpus. Our approach, which we call Structure Induction for Reordering (STIR), requires no syntactic annotations to train, but approaches the performance of a recent syntactic pre-ordering method in a large-scale English-Japanese MT system.

STIR predicts a pre-ordering via two pipelined models: (1) parsing and (2) tree reordering. The first model induces a binary parse, which defines the space of possible reorderings. In particular, only trees that properly separate verbs from their object noun phrases will license an SVO to SOV transformation. The second model locally permutes this tree. Our approach resembles work with binary synchronous grammars (Wu, 1997), but is distinct in its emphasis on monolingual parsing as a first phase, and in selecting reorderings without the aid of a target-side language model.

The parsing model is trained to maximize the conditional likelihood of trees that license the reorderings implied by observed word alignments in a parallel corpus. This objective differs from those of previous grammar induction models, which typically focus on succinctly explaining the observed source language corpus via latent hierarchical structure (Pereira and Schabes, 1992; Klein and Manning, 2002). Our convex objective allows us to train a feature-rich log-linear parsing model, even without supervised treebank data.

Focusing on pre-ordering for MT leads to a new

perspective on the canonical NLP task of grammar induction—one which marries the wide-spread scientific interest in unsupervised parsing models with a clear application and extrinsic evaluation methodology. To support this perspective, we highlight several avenues of future research throughout the paper.

We evaluate STIR in a large-scale English-Japanese machine translation system. We measure how closely our predicted reorderings match those implied by hand-annotated word alignments. STIR approaches the performance of the state-of-the-art pre-ordering method described in Genzel (2010), which learns reordering rules for supervised treebank parses. STIR gives a translation improvement of 3.84 BLEU over a standard phrase-based system with an integrated reordering model.

2 Parsing and Reordering Models

STIR consists of two pipelined log-linear models for parsing and reordering, as well as a third model for inducing trees from parallel corpora, trees that serve to train the first two models. This section describes the domain and structure of each model, while Section 3 describes features and learning objectives.

Figure 1 depicts the relationship between the three models. For each aligned sentence pair in a parallel corpus, the *parallel parsing model* selects a binary tree t over the source sentence, such that t licenses the reordering pattern implied by the word alignment (Section 2.2). The *monolingual parsing model* is trained to generate t without inspecting the alignments or target sentences (Section 2.3). The *tree reordering model* is trained to locally permute t to produce the target order (Section 2.4). In the context of an MT system, the monolingual parser and tree reorderer are applied in sequence to pre-order source sentences.

2.1 Unlabeled Binary Trees

Unlabeled binary trees are central to the STIR pipeline. We represent trees via their constituent spans. Let $[k, \ell)$ denote a span of indices of a 0-indexed word sequence $\mathbf{e}$, where $i \in [k, \ell)$ if $k \leq i < \ell$. $[0, n)$ denotes the root span covering the whole sequence, where $n = |\mathbf{e}|$.

A tree $t = (\mathcal{T}, \mathcal{N})$ consists of a set of terminal spans $\mathcal{T}$ and non-terminal spans $\mathcal{N}$. Each non-terminal span $[k, \ell) \in \mathcal{N}$ has a split point m, where $k < m < \ell$ splits the span into child spans $[k, m)$ and $[m, \ell)$. Formally, a pair $(\mathcal{T}, \mathcal{N})$ is a well-formed tree over $[0, n)$ if:

- The root span $[0, n) \in \mathcal{T} \cup \mathcal{N}$.
- For each $[k, \ell) \in \mathcal{N}$, there exists exactly one m such that $\{[k, m), [m, \ell)\} \subset \mathcal{T} \cup \mathcal{N}$.
- Terminal spans $\mathcal{T}$ are disjoint, but cover $[0, n)$.

These trees include multi-word terminal spans. It is often convenient to refer to a split non-terminal triple (k, m, ℓ) that include a non-terminal span $[k, \ell)$ and its split point m. We denote the set of these triples as

$$\mathcal{N}^+ = \{(k, m, \ell) : \{[k, \ell), [k, m), [m, \ell)\} \in \mathcal{T} \cup \mathcal{N}\}.$$

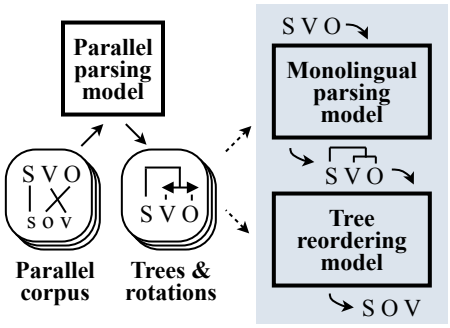

Figure 1: The training and reordering pipeline for STIR contains three models. The inputs and outputs of each model are indicated by solid arrows, while dashed arrows indicate the source of training examples. The parallel parsing model provides tree and reordering examples that are used to train the other models. In an MT system, the trained reordering pipeline (shaded) pre-orders a source sentence without target-side or alignment information.

2.2 Parallel Parsing Model

The first step in the STIR pipeline is to select a binary parse of each source sentence in a parallel corpus, one which licenses the reordering implied by a word alignment. Let the triple $(\mathbf{e}, \mathbf{f}, \mathcal{A})$ be an aligned sentence pair, where $\mathbf{e}$ and $\mathbf{f}$ are word sequences and $\mathcal{A}$ is a set of links (i, j) indicating that e_i aligns to f_j.

The set $\mathcal{A}$ provides ordering information over $\mathbf{e}$. To simplify definitions below, we first adjust $\mathcal{A}$ to

ignore all unaligned words in $\mathbf{f}$.

$$\mathcal{A}' = \{(i, c(j)) : (i,j) \in \mathcal{A}\}$$
$$c(j) = |\{j' : j' < j \ \wedge \ \exists i \text{ such that } (i,j') \in \mathcal{A}\}| \ .$$

$c(j)$ is the number of aligned words in $\mathbf{f}$ prior to position j. Next, we define a projection function:

$$\psi(i) = \left[\min_{j \in J_i} j, \max_{j \in J_i} j + 1\right)$$
$$J_i = \{j : (i,j) \in \mathcal{A}'\} \ ,$$

and let $\psi(i) = \emptyset$ if e_i is unaligned. We can extend this projection function to spans $[k, \ell)$ of $\mathbf{e}$ via union:

$$\psi(k, \ell) = \bigcup_{k \leq i < \ell} \psi(i) \ .$$

We say that a span $[k, \ell)$ aligns contiguously if

$$\forall (i,j) \in \mathcal{A}', j \in \psi(k,\ell) \text{ implies } i \in [k, \ell) \ ,$$

which corresponds to the familiar definition that $[k, \ell)$ is one side of an extractable phrase pair. Unaligned spans do not align contiguously.

Given this notion of projection, we can relate trees to alignments. A tree $(\mathcal{T}, \mathcal{N})$ over $\mathbf{e}$ respects an alignment $\mathcal{A}'$ if all $[k, \ell) \in \mathcal{T} \cup \mathcal{N}$ align contiguously, and for every (k, m, ℓ), the projections $\psi(k, m)$ and $\psi(m, \ell)$ are adjacent. Projections are adjacent if the left bound of one is the right bound of the other, or if either is empty.

The parallel parsing model is a linear model over trees that respect $\mathcal{A}'$, which factors over spans.

$$\mathrm{s}(t) = \sum_{[k,\ell) \in \mathcal{T}} w_T \phi_T(k, \ell) + \sum_{(k,m,\ell) \in \mathcal{N}^+} w_N \phi_N(k, m, \ell)$$

where the weight vector $w = (w_T \ w_N)$ scores features ϕ_T on terminal spans and ϕ_N on non-terminal spans and their split points.

Exact inference under this model can be performed via a dynamic program that exploits the following recurrence. Let $\mathrm{s}(k, \ell)$ be the score of the highest scoring binary tree over the span $[k, \ell)$ that respects $\mathcal{A}'$. Then,

$$\mathrm{s}_T(k,\ell) = \begin{cases} w_T \phi_T(k,\ell) & \text{if } [k,\ell) \text{ aligns contiguously} \\ -\infty & \text{otherwise} \end{cases}$$
$$f(k,m,\ell) = \mathrm{s}(k,m) + \mathrm{s}(m,\ell) + w_N \phi_N(k,m,\ell)$$
$$\mathrm{s}_N(k,\ell) = \max_{m : k < m < \ell} \begin{cases} f(k,m,\ell) & \text{if } \psi(k,m) \text{ is adjacent to } \psi(m,\ell) \\ -\infty & \text{otherwise} \end{cases}$$
$$\mathrm{s}(k,\ell) = \max\left[\mathrm{s}_T(k,\ell), \mathrm{s}_N(k,\ell)\right]$$

Above, s_T scores terminal spans while filtering out those which are not contiguous. The function f scores non-terminal spans by the sum of their child scores and additional features ϕ_N of the parent span. The recursive function s_N maximizes over split points while filtering out non-adjacent children. The recurrence will assign a score of $-\infty$ to any tree that does not respect $\mathcal{A}'$. Section 3 describes the features of this model. $\mathrm{s}(k, \ell)$ can be computed efficiently using the CKY algorithm.

2.3 Monolingual Parsing Model

The monolingual parsing model is trained to select the same trees as the parallel model, but without any features or constraints that reference word alignments. Hence, it can be applied to a source sentence before its translation is known.

This model also scores untyped binary trees according to a linear model parameterized by some $w = (w_T \ w_N)$ that weights features on terminal and non-terminal spans, respectively. We impose a maximum terminal length of L, but otherwise allow any binary tree. The score $\mathrm{s}(k, \ell)$ of the maximal tree over a span $[k, \ell)$ satisfies the familiar recurrence:

$$\mathrm{s}_M(k,\ell) = \begin{cases} w_T \phi_T(k,\ell) & \text{if } \ell - k \leq L \\ -\infty & \text{otherwise} \end{cases}$$
$$\mathrm{s}(k,\ell) = \max\left[\mathrm{s}_L(k,\ell), \max_{m : k < m < \ell} f(k,m,\ell)\right]$$

Inference under this recurrence can also be performed using the CKY algorithm. Section 3 describes the feature functions and training method.

2.4 Tree Reordering Model

Given a binary tree $(\mathcal{T}, \mathcal{N})$ over a sentence $\mathbf{e}$, we can reorder $\mathbf{e}$ by (a) permuting the children of non-terminals and (b) permuting the words of terminal spans. Formally, a reordering r assigns each terminal $[k, \ell) \in \mathcal{T}$ a permutation $\sigma(k, \ell)$ of its words and each split non-terminal (k, m, ℓ) a permutation $b(k, m, \ell)$ of its subspans, which can be either monotone or inverted, in the case of a binary tree. The permutation $\sigma(k, \ell)$ of a non-terminal span $[k, \ell) \notin \mathcal{T}$ is defined recursively as:

$$\begin{cases} \sigma(k, m)\ \sigma(m, \ell) & \text{if } b(k, m, \ell) \text{ is monotone} \\ \sigma(m, \ell)\ \sigma(k, m) & \text{if } b(k, m, \ell) \text{ is inverted} \end{cases}$$

In this paper, we use a reordering model that selects each terminal $\sigma(k, \ell)$ and each split non-terminal $b(k, m, \ell)$ independently, conditioned on the sentence $\mathbf{e}$. While the sub-problems of choosing $\sigma(k, \ell)$ and $b(k, m, \ell)$ are formally similar, we consider and evaluate them separately because the former deals only with local reordering, while the latter involves long-distance structural reordering.

Because our trees are binary, selecting $b(k, m, \ell)$ is a binary classification problem. Selecting $\sigma(k, \ell)$ for a terminal is a multiclass prediction problem that chooses among the $(\ell - k)!$ permutations of terminal $[k, \ell)$. Development experiments in English-Japanese yielded the best results with a maximum terminal span length $L = 2$. Hence, in experiments, terminal reordering is also binary classification.

Because each permutation is independent of all the others, reordering inference via a single pass through the tree is optimal. However, a more complex search procedure would be necessary to maintain optimality if the decision of $b(k, m, \ell)$ referenced other permutations, such as $\sigma([k, m))$ or $\sigma([m, \ell))$. Coupling together inference in this way represents a possible area of future study.

3 Features and Training Objectives

Each of these linear models factors over features on either terminal spans $[k, \ell)$ or split non-terminals (k, m, ℓ). Features vary in concert with the learning objectives and search spaces of each model.

Figure 2 shows an example sentence from our development corpus, including the target (Japanese) sentence, alignment, projections, parallel parser prediction, monolingual parser prediction, and predicted permutation. The feature descriptions below reference this example.

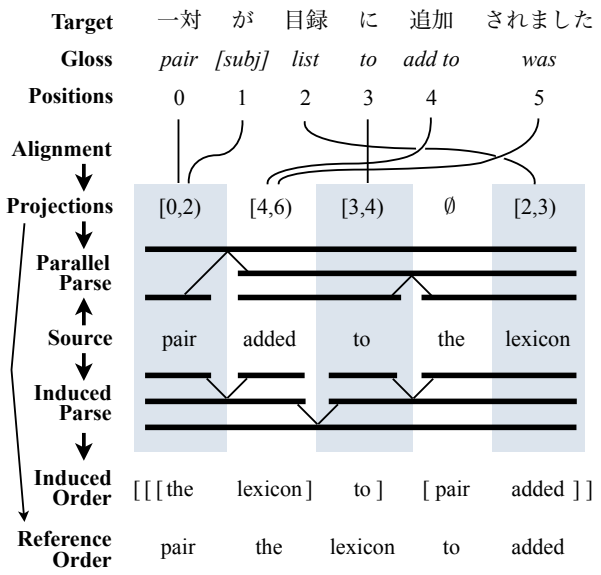

Figure 2: An example from our development corpus, annotated with the information flow (left) and annotations and predictions (right). Alignments inform projections, which are spans of the target associated with each source word. The parallel parse may only include contiguous spans. On the other hand, the induced parse may only condition on the source sentence. The induced order is restricted by the induced parse. In this example, the induced order is incorrect because the subject and verb form a constituent in the induced parse that cannot be separated correctly by the reordering model. This example demonstrates the important role of the induced parser in the STIR pipeline.

3.1 Tree Reordering Features

The tree reordering model consists of two local classifiers: the first can invert the two children of a non-terminal span, while the second can permute the words of a terminal span. The non-terminal classifier is trained on the trees that are selected by the parallel parsing model; the weights are chosen to minimize log loss of the correct permutation of each span (i.e., a maximum entropy model).

The terminal model is a multi-class maximum entropy model over the $n!$ possible permutations of the words in a terminal span. To make reordering more robust to monolingual parsing errors, the terminal

model is trained on all contiguous spans of each sentence up to length L, not just the terminal spans included in the parallel parsing tree.

The feature templates we apply to each span can be divided into the following five categories. Most features are shared across the two models.

Statistics. From a large aligned parallel corpus, we compute two statistics.

- $\mathrm{P}_C(e) = \frac{\text{count(e aligns contiguously)}}{\text{count}(e)}$ is the fraction of the time that a phrase e aligns contiguously to some target phrase, for all phrases up to length 4.
- $\mathrm{P}_D(e_i, e_j)$ is the fraction of the time that two co-occuring source words e_i and e_j align to adjacent positions in the target.

The first statistic indicates whether a contiguous phrase in the source should stay contiguous after reordering. Features based on this statistic apply to both terminal and short non-terminal spans. The second statistic indicates when a possibly discontiguous pair of words should be adjacent after reordering. This statistic is applied to pairs of words that would end up adjacent after an inversion: e_k and $e_{\ell-1}$ for span $[k, \ell)$. For instance, $\mathrm{P}_C(\text{added to}) = 0.68$ and $\mathrm{P}_D(\text{lexicon}, \text{to}) = 0.19$.

Cluster. All source word types are clustered into word classes, which together maximize likelihood of the source side of a large parallel corpus under a hidden Markov model, as in Uszkoreit and Brants (2008). Indicator features based on clusterings over c classes are defined over words e_k, e_{m-1}, e_m and $e_{\ell-1}$, as well as word sequences for spans up to length 4. Features are included for a variety of clusterings with sizes $c \in \{2^3, 2^4, \ldots, 2^{11}\}$.

POS. A supervised part-of-speech (POS) tagger provides coarse tags drawn from a 12 tag set $T = \{$Verb, Noun, Pronoun, Conjunction, Adjective, Adverb, Adposition, Determiner, Number, Particle/Function word, Punctuation, Other$\}$ (Petrov et al., 2011). Features based on these tags are computed identically to the features based on word classes.

Lexical. For a list of very common words in the source language, we include lexical indicator features for the boundary words e_k and $e_{\ell-1}$. For instance, the word "to" triggers a reordering, as do prepositions in general.

Length. Length computed as $\ell - k$, length as a fraction of sentence length, and quantized length features all contribute structural information.

All features except POS are computed directly from aligned parallel corpora. The Cluster and POS features play a similar role of expressing reordering patterns over collections of similar words. The ablation study in Section 5 compares these two feature sets directly.

3.2 Monolingual Parsing Features

The monolingual parsing model is also trained discriminatively, but involves structured prediction, as in a conditional random field (Lafferty et al., 2001). Conditional likelihood objectives have proven effective for supervised parsers (Finkel et al., 2008; Petrov and Klein, 2008). Recall that the score of a tree $t = (\mathcal{T}, \mathcal{N})$ factors over spans.

$$\mathrm{s}(t) = \sum_{[k,\ell) \in \mathcal{T}} w_T \phi_T(k, \ell) + \sum_{[k,\ell) \in \mathcal{N}} w_N \phi_N(k, m, \ell)$$

$$\mathrm{P}(t|\mathbf{e}) = \frac{\exp\left[\mathrm{s}(t)\right]}{\sum_{(t') \in \mathcal{B}(\mathbf{e})} \exp\left[\mathrm{s}(t')\right]}$$

where $\mathcal{B}(\mathbf{e})$ is the set of well-formed trees over $\mathbf{e}$.

The parallel parsing model (Section 2.2) produces a tree over the source sentence of each aligned sentence pair; these trees serve as our training examples. We can maximize their conditional likelihood according to this model via gradient methods. Each tree t over sentence $\mathbf{e}$ has a cumulative feature vector of dimension $|w| = |w_T| + |w_N|$, formed by stacking the terminal and non-terminal vectors:

$$\phi(t, \mathbf{e}) = \left(\sum_{[k,\ell) \in \mathcal{T}} \phi_T(k, \ell) \qquad \sum_{[k,\ell) \in \mathcal{N}} \phi_N(k, m, \ell) \right)$$

The contribution to the gradient objective from a tree t for a sentence $\mathbf{e}$ is the difference between observed

and expected feature vectors.

$$\mathcal{L}(w) = \sum_{(t,\mathbf{e})} \log \mathrm{P}(t|\mathbf{e})$$

$$\nabla\mathcal{L}(w) = \sum_{(t,\mathbf{e})} \left[\phi(t,\mathbf{e}) - \sum_{t' \in \mathcal{B}(\mathbf{e})} \mathrm{P}(t'|\mathbf{e}) \cdot \phi(t',\mathbf{e}) \right]$$

The second term in the gradient—the expected feature vector—can be computed efficiently because the feature vector $\phi(t')$ decomposes over the spans of t'. In particular, the inside-outside algorithm provides the quantities needed to compute the posterior probability of each terminal span $[k,\ell)$ and each split non-terminal (k,m,ℓ). Let, $\alpha(k,\ell)$ and $\beta(k,\ell)$ be the outside and inside scores of a span, respectively, computed using a log-sum semiring. Then, the log probablility that a terminal span $[k,\ell)$ appears in the tree for $\mathbf{e}$ under the posterior distribution $\mathrm{P}(t|\mathbf{e})$ is $\alpha(k,\ell) + w_T\phi_T(k,\ell)$. Note that this terminal posterior does not include the inside score of the span.

The log probability that a non-terminal span $[k,\ell)$ appears with split point m is

$$\alpha(k,\ell) + \beta(k,m) + \beta(m,\ell) + w_N\phi_N(k,m,\ell)$$

By the linearity of expectations, the expected feature vector for $\mathbf{e}$ can be computed by averaging the feature vectors of each terminal and split non-terminal span, weighted by their posterior probabilities.

In future work, one may consider training this model to maximize the likelihood of an entire forest of trees, in order to maintain uncertainty over which tree licensed a particular alignment.

We are currently using l-BFGS to optimize this objective over a relatively small training corpus, for 35 iterations. For this reason, we only include lexical features for very common words. Distributed or online training algorithms would perhaps allow for more training data (and therefore more lexicalized features) to be used in the future.

The features of this parsing model share the same types as the tree reordering models, but vary in their definition. The differences stem primarily from the different purpose of the model: here, features are not meant to decide how to reorder the sentence, but instead how to bracket the sentence hierarchically so that it can be reordered.

In particular, terminal spans have features on the sequence of POS tags and word clusters they contain, while a split non-terminal (k,m,ℓ) is scored based on the tags/clusters of the following words and word pairs: $e_k, e_{m-1}, e_m, e_{\ell-1}, (e_k,e_m), (e_k,e_{\ell-1})$, and (e_{m-1},e_m). The head word of a constituent often appears at one of its boundary positions, and so these features provide a proxy for explicitly tracking constituent heads in a parser.

Context features also appear, inspired by the constituent-context model of Klein and Manning (2001). For a span $[k,\ell)$, we add indicator features on the POS tags and word clusters of the words (e_{k-1},e_ℓ) which directly surround the constituent.

Features based on the statistic $\mathrm{P}_C(e)$ are also scored in the parsing model on all spans of length up to 4.

Length features score various structural aspects of each non-terminal (k,m,ℓ), such as $\frac{m-k}{\ell-k}$, $\frac{m-k}{k-m}$, etc.

One particularly interesting direction for future work is to train a single parsing model that licenses the reordering for several different languages. We might expect that a reasonable syntactic bracketing of English would simultaneously license the head-final transformations necessary to produce a Japanese or Korean ordering, and also the verb-subject-object ordering of formal Arabic.[1]

3.3 Parallel Parsing Features

The parallel parsing model does not run at translation time, but instead provides training examples to the other two models. Hence, defining an appropriate learning objective for this model is more challenging.

In the end, we are interested in selecting trees that we can learn to reproduce without an alignment (via the monolingual parsing model) and which can be reordered reliably (via the tree reordering model). Note that by construction, any tree selected by the parallel parsing model can be reordered perfectly. However, some of those trees will be easier to reproduce and reorder than others.

[1] An astute reviewer pointed out that no binary tree over an S-V-O sentence can license both S-O-V and V-S-O orderings. Hence, parse trees that are induced for multilingual reordering will need n-ary branches.

3.3.1 Reordering Loss Function

In order to measure the effectiveness of a reordering pipeline, we would like a metric over permutations. Fortunately, permutation loss for machine translation is already an established component of the METEOR metric, called a fragmentation penalty (Lavie and Agarwal, 2007). We define a slight variant of METEOR's fragmentation penalty that ranges from 0 to 1.

Given a sentence $\mathbf{e}$, a reference permutation σ^* of $(0, \cdots, |\mathbf{e}| - 1)$, and a hypothesized permutation $\hat{\sigma}$, let chunks$(\hat{\sigma}, \sigma^*)$ be the minimum number of "chunks" in $\hat{\sigma}$: the number of elements in a partition of $\hat{\sigma}$ such that each contiguous subsequence is also contiguous in σ^*.

We can define the reordering score between two permutations in terms of chunks.

$$R(\hat{\sigma}, \sigma^*) = \frac{|\sigma^*| - \text{chunks}(\hat{\sigma}, \sigma^*)}{|\sigma^*| - 1} \quad (1)$$

If $\hat{\sigma} = \sigma^*$, then chunks$(\hat{\sigma}, \sigma^*) = 1$. If no two adjacent elements of $\hat{\sigma}$ are adjacent in σ^*, then chunks$(\hat{\sigma}, \sigma^*) = |\sigma|$. Hence, the metric defined by Equation 1 ranges from 0 to 1.

The reference permutation σ^* of a source sentence $\mathbf{e}$ can be defined from an aligned sentence pair $(\mathbf{e}, \mathbf{f}, \mathcal{A})$ by sorting the words e_i of $\mathbf{e}$ by the left bound of their projection $\psi(i)$. Null-aligned words are placed to the left of the next aligned word to their right in the original order.

The reordering-specific loss functions defined in Equation 1 has been shown to correlate with human judgements of translation quality, especially for language pairs with substantial reordering like English-Japanese (Talbot et al., 2011). Other reordering-specific loss functions also correlate with human judgements (Birch et al., 2010). Future research could experiment with alternative reordering-based loss functions, such as Kendall's Tau, as suggested by Birch and Osborne (2011).

3.3.2 Parallel Parsing Objective

We can train our reordering pipeline by dividing an aligned parallel corpus into two halves, A and B, where the monolingual parsing and tree reordering models are trained on A, and their effectiveness is evaluated on held-out set B. Then, the effectiveness of the parallel parsing model is best measured on B, given fully trained parsing and reordering models.

$$\sum_{(\mathbf{e},\sigma^*) \in B} R\left(\sigma\left(\arg\max_{t \in \mathcal{B}(\mathbf{e})} [w \cdot \phi(t)]\right), \sigma^*\right) \quad (2)$$

Evaluating this objective involves training the other two models. Therefore, we can only hope to optimize this objective directly over a small dimensional space, for instance using a grid search. For this reason, we currently only include 4 features in the parallel parsing model for a tree t:

1. The sum of $\log \text{P}_C(e)$ for all terminals e in t with length greater than 1.
2. The count of length-1 terminal spans in t.
3. The count of terminals of length greater than k.
4. An indicator feature of whether parentheses and brackets are balanced in each span.

The model weights of features 3 and 4 above are fixed to large negative constants to prefer terminal spans of length up to k and spans with balanced punctuation. The weight of feature 1 is fixed to 1, and weight 2 was set via line search to 0.3. Ties among trees were broken randomly.

Of course, the problem of selecting training trees need not be directly tied to the end task of reordering, as in Equation 2. Instead, we might consider selecting trees according to a likelihood objective on the source side of a parallel corpus, similar to how monolingual grammar induction models often optimize corpus likelihood. In such a case, we could imagine training models with far more parameters, but we leave this research direction to future work.

4 Related Work

Our approach to inducing hierarchical structure for pre-ordering relates to several areas of previous work, including other pre-ordering methods, reordering models more generally, and models for the unsupervised induction of syntactic structure.

4.1 Pre-Ordering Models

Our reordering pipeline is intentionally similar to approaches that use a treebank-trained supervised

parser to reorder source sentences at training and translation time (Xia and McCord, 2004; Collins et al., 2005; Lee et al., 2010). Given a supervised parser, a rule-based pre-ordering procedure can either be specified by hand (Xu et al., 2009) or learned automatically (Genzel, 2010). We consider our approach to be a direct extension of these approaches, but one which induces structure from parallel corpora rather than relying on a treebank.

Tromble (2009) show that some pre-ordering benefits can be realized without a parsing step at all, by instead casting pre-ordering as a permutation modeling problem. While not splitting the task of pre-ordering into parsing and tree rordering, that work shows that pre-ordering models can be learned directly from parallel corpora.

4.2 Integrated Reordering Models

Distortion models have been primary components in machine translation models since the advent of statistical MT (Brown et al., 1993). In modern systems, reordering models are integrated into decoders as additional features in a discriminative log-linear model, which also includes a language model, translation features, etc. In these cases, reordering models interact with the strong signal of a target-side language model. Because ordering prediciton is conflated with target-side generation, evaluations are conducted on the entire generated output, which cannot isolate reordering errors from other sorts of errors, like lexical selection.

Despite these differences, certain integrated reordering models are similar in character to syntactic pre-ordering models. In particular, the tree rotation model of Yamada and Knight (2001) posited that reordering decisions involve rotations of a source-side syntax tree. The parameters of such a model can be trained by treating tree rotations as latent variables in a factored translation model, which parameterizes reordering and transfer separately but performs joint inference (Dyer and Resnik, 2010). Syntactic reordering and transfer can also be modeled jointly, for instance in a tree-to-string translation system parameterized by a transducer grammar.

While the success of integrated reordering models certainly highlights the importance of reordering in machine translation systems, we see several advantages to a pipelined, pre-ordering approach. First, the pre-ordering model can be trained and evaluated directly. Second, pre-ordering models need not factor according to the same dynamic program as the translation model. Third, the same reordering can be applied during training (for word alignment and rule extraction) and translation time without adding complexity to the extraction and decoding algorithms. Of course, integrating our model into translation inference represents a potentially fruitful avenue of future research.

4.3 Grammar Induction

The language processing community actively works on the problem of automatically inducing grammatical structure from a corpus of text (Pereira and Schabes, 1992). Some success in this area has been demonstrated via generative models (Klein and Manning, 2002), which often benefit from well-chosen priors (Cohen and Smith, 2009) or posterior constraints (Ganchev et al., 2009). In principle, these models must discover the syntactic patterns that govern a language from the sequences of word tokens alone. These models are often evaluated relative to reference treebank annotations.

Grammar induction in the context of machine translation reordering offers different properties. The alignment patterns in a parallel corpus provide an additional signal to models that is strongly tied to syntactic properties of the aligned languages. Also, the evaluation is straightforward—any syntactic structure that supports the prediction of reordering is rewarded.

Kuhn (2004) applied alignment-based constraints to the problem of inducing probabilistic context-free grammars, and showed an improvement with respect to Penn Treebank annotations over monolingual induction. Their work is distinct from ours because it focused on projecting distituents across languages, but mirrors ours in demonstrating that there is a role for aligned parallel corpora in grammar induction.

Snyder et al. (2009) also demonstrated that parallel corpora can play a role in improving the quality of grammar induction models. Their work differs from ours in that it focuses on multilingual lexical statistics and dependency relationships, rather than reordering patterns.

		Parsing			Tree Reordering			Pipeline
		Prec	Rec	$F1$	acc_N	acc_T	R_O	R
Annotated word alignments	All features	82.0	87.8	84.8	97.3	93.6	87.7	80.5
	All but POS	81.3	87.7	84.4	97.0	92.6	86.6	79.4
	All but Cluster	81.2	87.9	84.4	95.9	93.2	83.8	77.8
	All but POS & Cluster	75.4	82.0	78.5	89.2	89.7	66.8	49.7
Learned alignments	All features	72.5	61.0	66.3	91.6	83.3	72.0	49.5
Monotone order								34.9
Inverted order								30.8
Syntactic pre-ordering (Genzel, 2010)								66.0

Table 1: Accuracy of individual monolingual parsing and reordering models, as well as complete pipelines trained on annotated and learned word alignments.

4.4 Bilingual Grammar Induction

Also related to STIR is previous work on bilingual grammar induction from parallel corpora using ITG (Blunsom et al., 2009). These models have focused on learning phrasal translations — which are the terminal productions of a synchronous ITG — rather than reordering patterns that occur higher in the tree. Hence, while this paper shares formal machinery and data sources with that line of work, the models themselves target orthogonal aspects of the translation problem.

5 Experimental Results

As training data for our models we used 14,000 English sentences that were sampled from the web, translated into Japanese, and manually annotated with word alignments. The annotation was carried out by the original translators to promote consistency of analysis. Talbot et al. (2011) describes this corpus in further detail. A held-out test set of 396 manually aligned sentence pairs was used to evaluate reordering accuracy. Statistics used for features were computed from the full, unreordered, automatically word aligned, parallel training corpus used for the translation experiments described below.

5.1 Individual Model Accuracy

We evaluate the accuracy of the monolingual parsing models by their span F1, relative to the trees induced by the parallel parsing model on the held-out set. The first row of Table 1 shows that the model was able to reliably replicate the parses induced from alignments, at 84.8% F1. The following three lines show that removing either POS or cluster features degrades performance by only 0.4% F1, indicating that POS features are largely redundant in the presence of automatically induced word class features. Hence, no syntactic annotations are necessary at all to train the model.

We report two accuracy measures for the tree reordering model, one for non-terminal spans (acc_N) and one for terminal spans (acc_T). The following column, labeled R_O, is the reordering score of the tree reordering model applied to the oracle parallel parser tree. This score is independent of the monolingual parsing model.

The fifth line, labeled *learned alignments*, shows the impact of replacing manual alignment annotations with learned Model 1 alignments, trained in both directions and combined with the *refined* heuristic (Brown et al., 1993; Och et al., 1999).

The *pipeline* column shows the reordering score of the full STIR pipeline compared to two simple baselines: *Monotone* applies no reordering, while *inverted* simply inverts the word order. STIR outperforms all three other systems.

In the final line, we compare to the syntax-based pre-ordering system described in Genzel (2010). This approach first parses source sentences with a supervised parser, then learns reordering rules that permute those trees.

5.2 Translation Quality

We apply STIR as a pre-ordering step in a state-of-the-art phrase-based translation system from English to Japanese (Koehn et al., 2003). At training

time, pre-ordering is applied to the source side of every sentence pair in the training corpus before word alignment and phrase extraction. Likewise, every input sentence is pre-ordered at translation time.

Our baseline is the same system, but without pre-ordering. Our implementation's integrated distortion model is expressed as a negative exponential function of the distance between the current and previous source phrase, with a maximum jump width of four words. Our in-house decoder is based on the alignment template approach to translation and uses a small set of standard feature functions during decoding (Och and Ney, 2004).

We compare to using an integrated lexicalized reordering model (Koehn and Monz, 2005), a forest-to-string translation model (Zhang et al., 2011) and finally the syntactic pre-ordering technique of Genzel (2010) applied to the phrase-based baseline. We evaluate the impact of the proposed approach on translation quality as measured by the BLEU score on the token level (Papineni et al., 2002).

The translation model is trained on 700 million tokens of parallel text, primarily extracted from the web using automated parallel document identification (Uszkoreit et al., 2010). Alignments were learned using two iterations of Model 1 and two iterations of the HMM alignment model (Vogel et al., 1996). Our *dev* and *test* data sets consist of 3100 and 1000 English sentences, respectively, that were randomly sampled from the web and translated into Japanese. The *eval* set is a larger, heterogenous set containing 12,784 sentences. In all cases, the final log-linear models were optimized on the *dev* set using lattice-based Minimum Error Rate Training (Macherey et al., 2008).

Table 2 shows that STIR improves over the baseline system by a large margin of 3.84% BLEU (test). These gains are comparable in magnitude to those reported in Genzel (2010). Our induced parses are competitive with both systems that use syntactic parsers and substantially outperform lexicalized reordering.

	BLEU %		
	dev	test	eval
Baseline	18.65	19.02	13.60
Lexicalized Reordering	19.45	18.92	13.99
Forest-to-String	**23.08**	22.85	**16.60**
Syntactic Pre-ordering	22.59	**23.28**	16.31
STIR: annotated	22.46	22.86	16.39
STIR: learned	20.28	20.66	14.64

Table 2: Translation quality, measured by BLEU, for English to Japanese. STIR results use both manually annotated and learned alignments.

6 Conclusion

We have demonstrated that induced parses suffice for pre-ordering. We hope that future work in grammar induction will also consider pre-ordering as an extrinsic evaluation.

References

Alexandra Birch and Miles Osborne. 2011. Reordering metrics for MT. In *Proceedings of the Association for Computational Linguistics*.

Alexandra Birch, Phil Blunsom, and Miles Osborne. 2010. Metrics for MT evaluation: Evaluating reordering. *Machine Translation*.

Phil Blunsom, Trevor Cohn, Chris Dyer, and Miles Osborne. 2009. A Gibbs sampler for phrasal synchronous grammar induction. In *Proceedings of the Association for Computational Linguistics*.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*.

Shay Cohen and Noah Smith. 2009. Shared logistic normal distributions for soft parameter tying in unsupervised grammar induction. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.

Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the Association for Computational Linguistics*.

Chris Dyer and Philip Resnik. 2010. Context-free reordering, finite-state translation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.

Jenny Rose Finkel, Alex Kleeman, and Christopher D. Manning. 2008. Efficient, feature-based, conditional random field parsing. In *Proceedings of the Association for Computational Linguistics*.

Kuzman Ganchev, Jennifer Gillenwater, and Ben Taskar. 2009. Dependency grammar induction via bitext projection constraints. In *Proceedings of the Association for Computational Linguistics*.

Dmitriy Genzel. 2010. Automatically learning source-side reordering rules for large scale machine translation. In *Proceedings of the Conference on Computational Linguistics*.

Dan Klein and Christopher D. Manning. 2001. Natural language grammar induction using a constituent-context model. In *Proceedings of Neural Information Processing Systems*.

Dan Klein and Christopher D. Manning. 2002. A generative constituent-context model for improved grammar induction. In *Proceedings of the Association for Computational Linguistics*.

Philipp Koehn and Christof Monz. 2005. Shared task: Statistical machine translation between european languages. In *Proceedings of the International Workshop on Spoken Language Translation*.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.

Jonas Kuhn. 2004. Experiments in parallel-text based grammar induction. In *Proceedings of the Association for Computational Linguistics*.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning*.

Alon Lavie and Abhaya Agarwal. 2007. METEOR: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of ACL Workshop on Statistical Machine Translation*.

Young-Suk Lee, Bing Zhao, and Xiaoqiang Luo. 2010. Constituent reordering and syntax models for English-to-Japanese statistical machine translation. In *Proceedings of the Conference on Computational Linguistics*.

Wolfgang Macherey, Franz Och, Ignacio Thayer, and Jakob Uszkoreit. 2008. Lattice-based minimum error rate training for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Franz Josef Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*.

Franz Josef Och, Christopher Tillman, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the Association for Computational Linguistics*.

Fernando Pereira and Yves Schabes. 1992. Inside-outside reestimation from partially bracketed corpora. In *Proceedings of the Association for Computational Linguistics*.

Slav Petrov and Dan Klein. 2008. Sparse multi-scale grammars for discriminative latent variable parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2011. A universal part-of-speech tagset. Technical report.

Benjamin Snyder, Tahira Naseem, and Regina Barzilay. 2009. Unsupervised multilingual grammar induction. In *Proceedings of the Association for Computational Linguistics*.

David Talbot, Hideto Kazawa, Hiroshi Ichikawa, Jason Katz-Brown, Masakazu Seno, and Franz J. Och. 2011. A lightweight evaluation framework for machine translation reordering. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*.

Roy Tromble. 2009. Learning linear ordering problems for better translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Jakob Uszkoreit and Thorsten Brants. 2008. Distributed word clustering for large scale class-based language modeling in machine translation. In *Proceedings of the Association for Computational Linguistics*.

Jakob Uszkoreit, Jay Ponte, Ashok Popat, and Moshe Dubiner. 2010. Large scale parallel document mining for machine translation. In *Proceedings of the Conference on Computational Linguistics*.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *Proceedings of the Conference on Computational linguistics*.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*.

Fei Xia and Michael McCord. 2004. Improving a statistical mt system with automatically learned rewrite patterns. In *Proceedings of the Conference on Computational Linguistics*.

Peng Xu, Jaeho Kang, Michael Ringgard, and Franz Och. 2009. Using a dependency parser to improve smt for subject-object-verb languages. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.

Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of the Association for Computational Linguistics*.

Hao Zhang, Licheng Fang, Peng Xu, and Xiaoyun Wu. 2011. Binarized forest to string translation. In *Proceedings of the Association for Computational Linguistics*.