Smaller Alignment Models for Better Translations: Unsupervised Word Alignment with the ℓ_0-norm

Ashish Vaswani **Liang Huang** **David Chiang**
University of Southern California
Information Sciences Institute
{avaswani,lhuang,chiang}@isi.edu

Abstract

Two decades after their invention, the IBM word-based translation models, widely available in the GIZA++ toolkit, remain the dominant approach to word alignment and an integral part of many statistical translation systems. Although many models have surpassed them in accuracy, none have supplanted them in practice. In this paper, we propose a simple extension to the IBM models: an ℓ_0 prior to encourage sparsity in the word-to-word translation model. We explain how to implement this extension efficiently for large-scale data (also released as a modification to GIZA++) and demonstrate, in experiments on Czech, Arabic, Chinese, and Urdu to English translation, significant improvements over IBM Model 4 in both word alignment (up to +6.7 F1) and translation quality (up to +1.4 B).

1 Introduction

Automatic word alignment is a vital component of nearly all current statistical translation pipelines. Although state-of-the-art translation models use rules that operate on units bigger than words (like phrases or tree fragments), they nearly always use word alignments to drive extraction of those translation rules. The dominant approach to word alignment has been the IBM models (Brown et al., 1993) together with the HMM model (Vogel et al., 1996). These models are unsupervised, making them applicable to any language pair for which parallel text is available. Moreover, they are widely disseminated in the open-source GIZA++ toolkit (Och and Ney, 2004). These properties make them the default choice for most statistical MT systems.

In the decades since their invention, many models have surpassed them in accuracy, but none has supplanted them in practice. Some of these models are partially supervised, combining unlabeled parallel text with manually-aligned parallel text (Moore, 2005; Taskar et al., 2005; Riesa and Marcu, 2010). Although manually-aligned data is very valuable, it is only available for a small number of language pairs. Other models are unsupervised like the IBM models (Liang et al., 2006; Graça et al., 2010; Dyer et al., 2011), but have not been as widely adopted as GIZA++ has.

In this paper, we propose a simple extension to the IBM/HMM models that is unsupervised like the IBM models, is as scalable as GIZA++ because it is implemented on top of GIZA++, and provides significant improvements in both alignment and translation quality. It extends the IBM/HMM models by incorporating an ℓ_0 prior, inspired by the principle of minimum description length (Barron et al., 1998), to encourage sparsity in the word-to-word translation model (Section 2.2). This extension follows our previous work on unsupervised part-of-speech tagging (Vaswani et al., 2010), but enables it to scale to the large datasets typical in word alignment, using an efficient training method based on projected gradient descent (Section 2.3). Experiments on Czech-, Arabic-, Chinese- and Urdu-English translation (Section 3) demonstrate consistent significant improvements over IBM Model 4 in both word alignment (up to +6.7 F1) and translation quality (up to +1.4 B). Our implementation has been released as a simple modification to the GIZA++ toolkit that can be used as a drop-in replacement for GIZA++ in any existing MT pipeline.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 311–319,
Jeju, Republic of Korea, 8-14 July 2012. ©2012 Association for Computational Linguistics

2 Method

We start with a brief review of the IBM and HMM word alignment models, then describe how to extend them with a smoothed ℓ_0 prior and how to efficiently train them.

2.1 IBM Models and HMM

Given a French string $\mathbf{f} = f_1 \cdots f_j \cdots f_m$ and an English string $\mathbf{e} = e_1 \cdots e_i \cdots e_\ell$, these models describe the process by which the French string is generated by the English string via the alignment $\mathbf{a} = a_1, \ldots, a_j, \ldots, a_m$. Each a_j is a hidden variables, indicating which English word e_{a_j} the French word f_j is aligned to.

In IBM Model 1–2 and the HMM model, the joint probability of the French sentence and alignment given the English sentence is

$$P(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) = \prod_{j=1}^{m} d(a_j \mid a_{j-1}, j) t(f_j \mid e_{a_j}). \qquad (1)$$

The parameters of these models are the distortion probabilities $d(a_j \mid a_{j-1}, j)$ and the translation probabilities $t(f_j \mid e_{a_j})$. The three models differ in their estimation of d, but the differences do not concern us here. All three models, as well as IBM Models 3–5, share the same t. For further details of these models, the reader is referred to the original papers describing them (Brown et al., 1993; Vogel et al., 1996).

Let θ stand for all the parameters of the model. The standard training procedure is to find the parameter values that maximize the likelihood, or, equivalently, minimize the negative log-likelihood of the observed data:

$$\hat{\theta} = \arg\min_{\theta} \left(-\log P(\mathbf{f} \mid \mathbf{e}, \theta)\right) \qquad (2)$$

$$= \arg\min_{\theta} \left(-\log \sum_{\mathbf{a}} P(\mathbf{f}, \mathbf{a} \mid \mathbf{e}, \theta)\right) \qquad (3)$$

This is done using the Expectation-Maximization (EM) algorithm (Dempster et al., 1977).

2.2 MAP-EM with the ℓ_0-norm

Maximum likelihood training is prone to overfitting, especially in models with many parameters. In word alignment, one well-known manifestation of overfitting is that rare words can act as "garbage collectors" (Moore, 2004), aligning to many unrelated words. This hurts alignment precision and rule-extraction recall. Previous attempted remedies include early stopping, smoothing (Moore, 2004), and posterior regularization (Graça et al., 2010).

We have previously proposed another simple remedy to overfitting in the context of unsupervised part-of-speech tagging (Vaswani et al., 2010), which is to minimize the size of the model using a smoothed ℓ_0 prior. Applying this prior to an HMM improves tagging accuracy for both Italian and English.

Here, our goal is to apply a similar prior in a word-alignment model to the word-to-word translation probabilities $t(f \mid e)$. We leave the distortion models alone, since they are not very large, and there is not much reason to believe that we can profit from compacting them.

With the addition of the ℓ_0 prior, the MAP (maximum *a posteriori*) objective function is

$$\hat{\theta} = \arg\min_{\theta} \left(-\log P(\mathbf{f} \mid \mathbf{e}, \theta) P(\theta)\right) \qquad (4)$$

where

$$P(\theta) \propto \exp\left(-\alpha \|\theta\|_0^\beta\right) \qquad (5)$$

and

$$\|\theta\|_0^\beta = \sum_{e,f} \left(1 - \exp \frac{-t(f \mid e)}{\beta}\right) \qquad (6)$$

is a smoothed approximation of the ℓ_0-norm. The hyperparameter β controls the tightness of the approximation, as illustrated in Figure 1. Substituting back into (4) and dropping constant terms, we get the following optimization problem: minimize

$$-\log P(\mathbf{f} \mid \mathbf{e}, \theta) - \alpha \sum_{e,f} \exp \frac{-t(f \mid e)}{\beta} \qquad (7)$$

subject to the constraints

$$\sum_{f} t(f \mid e) = 1 \qquad \text{for all } e. \qquad (8)$$

We can carry out the optimization in (7) with the MAP-EM algorithm (Bishop, 2006). EM and MAP-EM share the same E-step; the difference lies in the

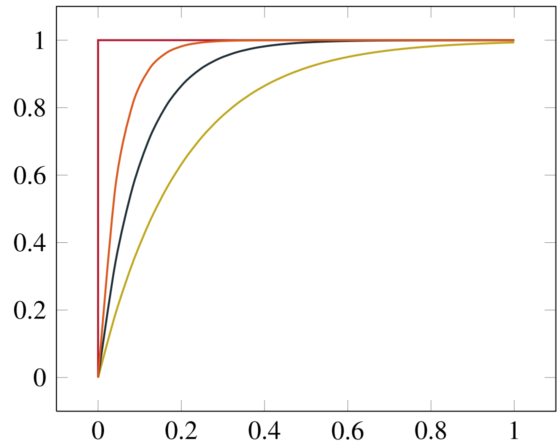

Figure 1: The ℓ_0-norm (top curve) and smoothed approximations (below) for $\beta = 0.05, 0.1, 0.2$.

M-step. For vanilla EM, the M-step is:

$$\hat{\theta} = \arg\min_{\theta} \left(-\sum_{e,f} E[C(e,f)] \log t(f \mid e) \right) \tag{9}$$

again subject to the constraints (8). The count $C(e, f)$ is the number of times that f occurs aligned to e. For MAP-EM, it is:

$$\hat{\theta} = \arg\min_{\theta} \Bigg(-\sum_{e,f} E[C(e,f)] \log t(f \mid e) - \alpha \sum_{e,f} \exp \frac{-t(f \mid e)}{\beta} \Bigg) \tag{10}$$

This optimization problem is non-convex, and we do not know of a closed-form solution. Previously (Vaswani et al., 2010), we used ALGENCAN, a nonlinear optimization toolkit, but this solution does not scale well to the number of parameters involved in word alignment models. Instead, we use a simpler and more scalable method which we describe in the next section.

2.3 Projected gradient descent

Following Schoenemann (2011b), we use projected gradient descent (PGD) to solve the M-step (but with the ℓ_0-norm instead of the ℓ_1-norm). Gradient projection methods are attractive solutions to constrained optimization problems, particularly when the constraints on the parameters are simple (Bertsekas, 1999). Let $F(\theta)$ be the objective function in (10); we seek to minimize this function. As in previous work (Vaswani et al., 2010), we optimize each set of parameters $\{t(\cdot \mid e)\}$ separately for each English word type e. The inputs to the PGD are the expected counts $E[C(e, f)]$ and the current word-to-word conditional probabilities θ. We run PGD for K iterations, producing a sequence of intermediate parameter vectors $\theta^1, \ldots, \theta^k, \ldots, \theta^K$. Each iteration has two steps, a projection step and a line search.

Projection step In this step, we compute:

$$\overline{\theta}^k = \left[\theta^k - s\nabla F(\theta^k)\right]^\Delta \tag{11}$$

This moves θ in the direction of steepest descent (∇F) with step size s, and then the function $[\cdot]^\Delta$ projects the resulting point onto the simplex; that is, it finds the nearest point that satisfies the constraints (8).

The gradient $\nabla F(\theta^k)$ is

$$\frac{\partial F}{\partial t(f \mid e)} = -\frac{E[C(f,e)]}{t(f \mid e)} + \frac{\alpha}{\beta} \exp \frac{-t(f \mid e)}{\beta} \tag{12}$$

In contrast to Schoenemann (2011b), we use an $O(n \log n)$ algorithm for the projection step due to Duchi et. al. (2008), shown in Pseudocode 1.

Pseudocode 1 Project input vector $\mathbf{u} \in \mathbb{R}^n$ onto the probability simplex.

```
v = u sorted in non-increasing order
ρ = 0
for i = 1 to n do
    if $v_i - \frac{1}{i}\left(\sum_{r=1}^{i} v_r - 1\right) > 0$ then
        ρ = i
    end if
end for
$\eta = \frac{1}{\rho}\left(\sum_{r=1}^{\rho} v_r - 1\right)$
$w_r = \max\{v_r - \eta, 0\}$ for $1 \le r \le n$
return w
```

Line search Next, we move to a point between θ^k and $\overline{\theta}^k$ that satisfies the *Armijo condition*,

$$F(\theta^k + \delta_m) \le F(\theta^k) + \sigma\left(\nabla F(\theta^k) \cdot \delta_m\right) \tag{13}$$

where $\delta_m = \gamma^m(\overline{\theta}^k - \theta^k)$ and σ and γ are both constants in $(0, 1)$. We try values $m = 1, 2, \ldots$ until the Armijo condition (13) is satisfied or the limit $m = 20$

Pseudocode 2 Find a point between θ^k and $\overline{\theta}^k$ that satisfies the Armijo condition.

```
$F_{min} = F(\theta^k)$
$\theta_{min} = \theta^k$
for $m = 1$ to $20$ do
    $\delta_m = \gamma^m \left(\overline{\theta}^k - \theta^k\right)$
    if $F(\theta^k + \delta_m) < F_{min}$ then
        $F_{min} = F(\theta^k + \delta_m)$
        $\theta_{min} = \theta^k + \delta_m$
    end if
    if $F(\theta^k + \delta_m) \leq F(\theta^k) + \sigma \left(\nabla F(\theta^k) \cdot \delta_m\right)$ then
        break
    end if
end for
$\theta^{k+1} = \theta_{min}$
return $\theta^{k+1}$
```

is reached. (Note that we don't allow $m = 0$ because this can cause $\theta^k + \delta_m$ to land on the boundary of the probability simplex, where the objective function is undefined.) Then we set θ^{k+1} to the point in $\{\theta^k\} \cup \{\theta^k + \delta_m \mid 1 \leq m \leq 20\}$ that minimizes F. The line search algorithm is summarized in Pseudocode 2.

In our implementation, we set $\gamma = 0.5$ and $\sigma = 0.5$. We keep s fixed for all PGD iterations; we experimented with $s \in \{0.1, 0.5\}$ and did not observe significant changes in F-score. We run the projection step and line search alternately for at most K iterations, terminating early if there is no change in θ^k from one iteration to the next. We set $K = 35$ for the large Arabic-English experiment; for all other conditions, we set $K = 50$. These choices were made to balance efficiency and accuracy. We found that values of K between 30 and 75 were generally reasonable.

3 Experiments

To demonstrate the effect of the ℓ_0-norm on the IBM models, we performed experiments on four translation tasks: Arabic-English, Chinese-English, and Urdu-English from the NIST Open MT Evaluation, and the Czech-English translation from the Workshop on Machine Translation (WMT) shared task. We measured the accuracy of word alignments generated by GIZA++ with and without the ℓ_0-norm, and also translation accuracy of systems trained using the word alignments. Across all tests, we found strong improvements from adding the ℓ_0-norm.

3.1 Training

We have implemented our algorithm as an open-source extension to GIZA++.[1] Usage of the extension is identical to standard GIZA++, except that the user can switch the ℓ_0 prior on or off, and adjust the hyperparameters α and β.

For vanilla EM, we ran five iterations of Model 1, five iterations of HMM, and ten iterations of Model 4. For our approach, we first ran one iteration of Model 1, followed by four iterations of Model 1 with smoothed ℓ_0, followed by five iterations of HMM with smoothed ℓ_0. Finally, we ran ten iterations of Model 4.[2]

We used the following parallel data:

- Chinese-English: selected data from the constrained task of the NIST 2009 Open MT Evaluation.[3]

- Arabic-English: all available data for the constrained track of NIST 2009, excluding United Nations proceedings (LDC2004E13), ISI Automatically Extracted Parallel Text (LDC2007E08), and Ummah newswire text (LDC2004T18), for a total of 5.4+4.3 million words. We also experimented on a larger Arabic-English parallel text of 44+37 million words from the DARPA GALE program.

- Urdu-English: all available data for the constrained track of NIST 2009.

[1] The code can be downloaded from the first author's website at `http://www.isi.edu/~avaswani/giza-pp-l0.html`.

[2] GIZA++ allows changing some heuristic parameters for efficient training. Currently, we set two of these to zero: `mincountincrease` and `probcutoff`. In the default setting, both are set to 10^{-7}. We set `probcutoff` to 0 because we would like the optimization to learn the parameter values. For a fair comparison, we applied the same setting to our vanilla EM training as well. To test, we ran GIZA++ with the default setting on the smaller of our two Arabic-English datasets with the same number of iterations and found no change in F-score.

[3] LDC catalog numbers LDC2003E07, LDC2003E14, LDC2005E83, LDC2005T06, LDC2006E24, LDC2006E34, LDC2006E85, LDC2006E86, LDC2006E92, and LDC2006E93.

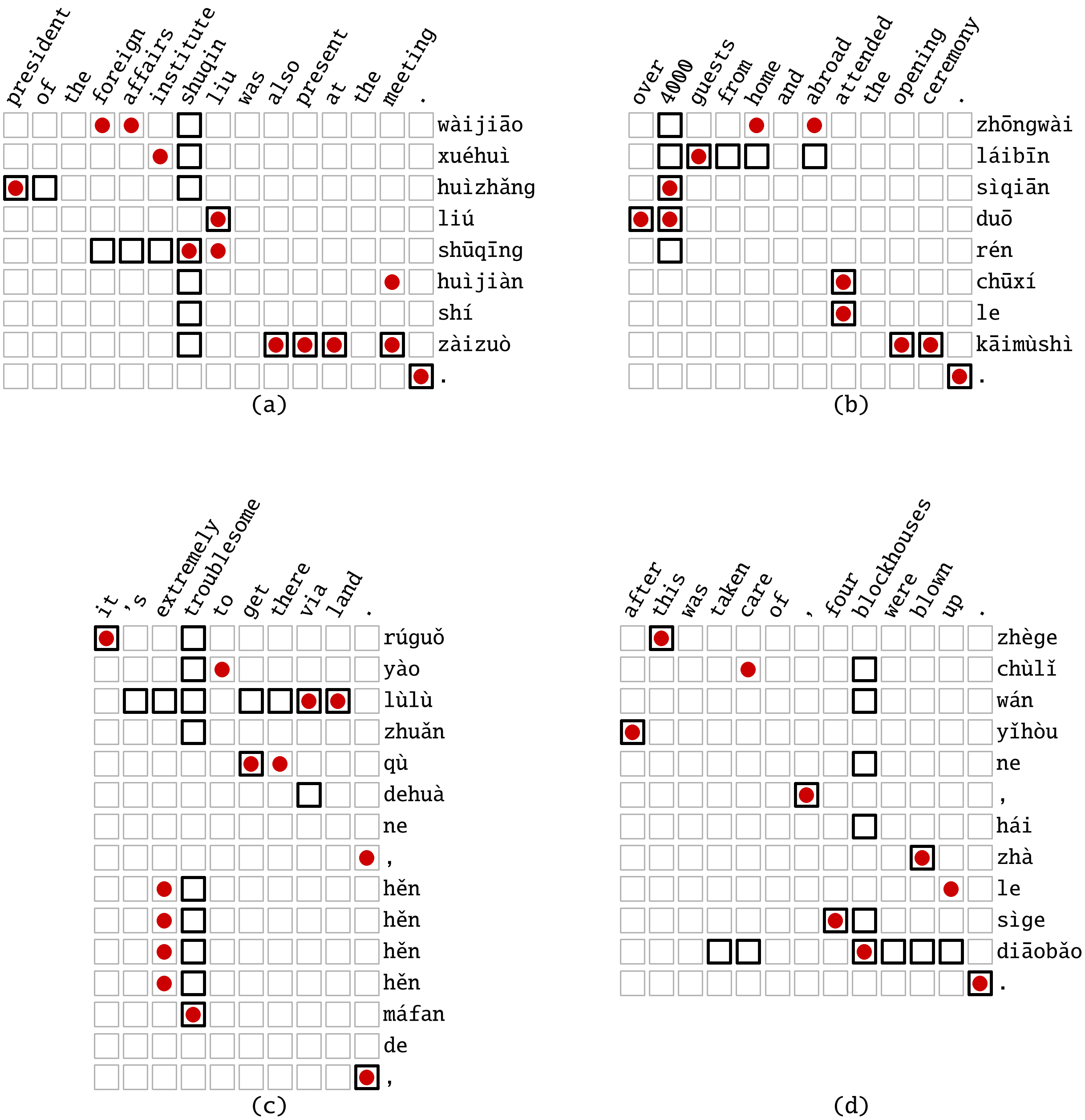
president of the foreign affairs institute shuqin liu was also present at the meeting .
wàijiāo
xuéhuì
huìzhǎng
liú
shūqīng
huìjiàn
shí
zàizuò
.
(a)
over 4000 guests from home and abroad attended the opening ceremony .
zhōngwài
láibīn
sìqiān
duō
rén
chūxí
le
kāimùshì
.
(b)
it 's extremely troublesome to get there via land .
rúguǒ
yào
lùlù
zhuǎn
qù
dehuà
ne
,
hěn
hěn
hěn
hěn
máfan
de
,
(c)
after this was taken care of , four blockhouses were blown up .
zhège
chùlǐ
wán
yǐhòu
ne
,
hái
zhà
le
sìge
diāobǎo
.
(d)

Figure 2: Smoothed-ℓ_0 alignments (red circles) correct many errors in the baseline GIZA++ alignments (black squares), as shown in four Chinese-English examples (the red circles are almost perfect for these examples, except for minor mistakes such as liu-shūqīng and meeting-zàizuò in (a) and .-, in (c)). In particular, the baseline system demonstrates typical "garbage-collection" phenomena in proper name "shuqing" in both languages in (a), number "4000" and word "láibīn" (lit. "guest") in (b), word "troublesome" and "lùlù" (lit. "land-route") in (c), and "blockhouses" and "diāobǎo" (lit. "bunker") in (d). We found this garbage-collection behavior to be especially common with proper names, numbers, and uncommon words in both languages. Most interestingly, in (c), our smoothed-ℓ_0 system correctly aligns "extremely" to "hěn hěn hěn hěn" (lit. "very very very very") which is rare in the bitext.

task	data (M)	system	align F1 (%)	word trans (M)	$\tilde{\phi}_{sing.}$	B (%)		
						2008	2009	2010
Chi-Eng	9.6+12	baseline	73.2	3.5	6.2	28.7		
		ℓ_0-norm	76.5	2.0	3.3	29.5		
		difference	+3.3	−43%	−47%	+0.8		
Ara-Eng	5.4+4.3	baseline	65.0	3.1	4.5	39.8	42.5	
		ℓ_0-norm	70.8	1.8	1.8	41.1	43.7	
		difference	+5.9	−39%	−60%	+1.3	+1.2	
Ara-Eng	44+37	baseline	66.2	15	5.0	41.6	44.9	
		ℓ_0-norm	71.8	7.9	1.8	42.5	45.3	
		difference	+5.6	−47%	−64%	+0.9	+0.4	
Urd-Eng	1.7+1.5	baseline		1.7	4.5	25.3*	29.8	
		ℓ_0-norm		1.2	2.2	25.9*	31.2	
		difference		−29%	−51%	+0.6*	+1.4	
Cze-Eng	2.1+2.3	baseline	65.6	1.5	3.0		17.3	18.0
		ℓ_0-norm	72.3	1.0	1.4		17.9	18.4
		difference	+6.7	−33%	−53%		+0.6	+0.4

Table 1: Adding the ℓ_0-norm to the IBM models improves both alignment and translation accuracy across four different language pairs. The *word trans* column also shows that the number of distinct word translations (i.e., the size of the lexical weighting table) is reduced. The $\tilde{\phi}_{sing.}$ column shows the average fertility of once-seen source words. For Czech-English, the year refers to the WMT shared task; for all other language pairs, the year refers to the NIST Open MT Evaluation. *Half of this test set was also used for tuning feature weights.

- Czech-English: A corpus of 4 million words of Czech-English data from the News Commentary corpus.[4]

We set the hyperparameters α and β by tuning on gold-standard word alignments (to maximize F1) when possible. For Arabic-English and Chinese-English, we used 346 and 184 hand-aligned sentences from LDC2006E86 and LDC2006E93. Similarly, for Czech-English, 515 hand-aligned sentences were available (Bojar and Prokopová, 2006). But for Urdu-English, since we did not have any gold alignments, we used $\alpha = 10$ and $\beta = 0.05$. We did not choose a large α, as the dataset was small, and we chose a conservative value for β.

We ran word alignment in both directions and symmetrized using *grow-diag-final* (Koehn et al., 2003). For models with the smoothed ℓ_0 prior, we tuned α and β separately in each direction.

3.2 Alignment

First, we evaluated alignment accuracy directly by comparing against gold-standard word alignments. The results are shown in the *alignment F1* column of Table 1. We used balanced F-measure rather than alignment error rate as our metric (Fraser and Marcu, 2007).

Following Dyer et al. (2011), we also measured the average fertility, $\tilde{\phi}_{sing.}$, of once-seen source words in the symmetrized alignments. Our alignments show smaller fertility for once-seen words, suggesting that they suffer from "garbage collection" effects less than the baseline alignments do.

The fact that we had to use hand-aligned data to tune the hyperparameters α and β means that our method is no longer completely unsupervised. However, our observation is that alignment accuracy is actually fairly robust to the choice of these hyperparameters, as shown in Table 2. As we will see below, we still obtained strong improvements in translation quality when hand-aligned data was unavailable.

We also tried generating 50 word classes using the tool provided in GIZA++. We found that adding word classes improved alignment quality a little, but more so for the baseline system (see Table 3). We used the alignments generated by training with word classes for our translation experiments.

[4] This data is available at `http://statmt.org/wmt10`.

β	model	α								
		0	10	25	50	75	100	250	500	750
–	HMM	47.5								
	M4	52.1								
0.5	HMM		46.3	48.4	52.8	55.7	57.5	61.5	62.6	**62.7**
	M4		51.7	53.7	56.4	58.6	59.8	63.3	64.4	64.8
0.1	HMM		55.6	60.4	61.6	62.1	61.9	61.8	60.2	60.1
	M4		58.2	62.4	64.0	64.4	64.8	65.5	65.6	**65.9**
0.05	HMM		59.1	61.4	62.4	62.5	62.3	60.8	58.7	57.7
	M4		61.0	63.5	64.6	65.3	65.3	65.4	65.7	65.7
0.01	HMM		59.7	61.6	60.0	59.5	58.7	56.9	55.7	54.7
	M4		62.9	65.0	65.1	65.2	65.1	65.4	65.3	65.4
0.005	HMM		58.1	59.0	58.3	57.6	57.0	55.9	53.9	51.7
	M4		62.0	64.1	64.5	64.5	64.5	65.0	64.8	64.6
0.001	HMM		51.7	52.1	51.4	49.3	50.4	46.8	45.4	44.0
	M4		59.8	61.3	61.5	61.0	61.8	61.2	61.0	61.2

Table 2: Almost all hyperparameter settings achieve higher F-scores than the baseline IBM Model 4 and HMM model for Arabic-English alignment ($\alpha = 0$).

		word classes?	
direction	system	no	yes
	baseline	49.0	52.1
$P(f \mid e)$	ℓ_0-norm	**63.9**	**65.9**
	difference	+14.9	+13.8
	baseline	64.3	65.2
$P(e \mid f)$	ℓ_0-norm	**69.2**	**70.3**
	difference	+4.9	+5.1

Table 3: Adding word classes improves the F-score in both directions for Arabic-English alignment by a little, for the baseline system more so than ours.

setting		align F1 (%)	B (%)	
$t(f \mid e)$	$t(e \mid f)$		2008	2009
1st	1st	70.8	41.1	43.7
1st	2nd	70.7	41.1	43.8
2nd	1st	70.7	40.7	44.1
2nd	2nd	70.9	41.1	44.2

Table 4: Optimizing hyperparameters on alignment F1 score does not necessarily lead to optimal B . The first two columns indicate whether we used the first- or second-best alignments in each direction (according to F1); the third column shows the F1 of the symmetrized alignments, whose corresponding B scores are shown in the last two columns.

Figure 2 shows four examples of Chinese-English alignment, comparing the baseline with our smoothed-ℓ_0 method. In all four cases, the baseline produces incorrect extra alignments that prevent good translation rules from being extracted while the smoothed-ℓ_0 results are correct. In particular, the baseline system demonstrates typical "garbage collection" behavior (Moore, 2004) in all four examples.

3.3 Translation

We then tested the effect of word alignments on translation quality using the hierarchical phrase-based translation system Hiero (Chiang, 2007). We used a fairly standard set of features: seven inherited from Pharaoh (Koehn et al., 2003), a second language model, and penalties for the glue rule, identity rules, unknown-word rules, and two kinds of number/name rules. The feature weights were discriminatively trained using MIRA (Chiang et al., 2008). We used two 5-gram language models, one on the combined English sides of the NIST 2009 Arabic-English and Chinese-English constrained tracks (385M words), and another on 2 billion words of English.

For each language pair, we extracted grammar rules from the same data that were used for word alignment. The development data that were used for discriminative training were: for Chinese-English and Arabic-English, data from the NIST 2004 and NIST 2006 test sets, plus newsgroup data from the

GALE program (LDC2006E92); for Urdu-English, half of the NIST 2008 test set; for Czech-English, a training set of 2051 sentences provided by the WMT10 translation workshop.

The results are shown in the B column of Table 1. We used case-insensitive IBM B (closest reference length) as our metric. Significance testing was carried out using bootstrap resampling with 1000 samples (Koehn, 2004; Zhang et al., 2004).

All of the tests showed significant improvements ($p < 0.01$), ranging from +0.4 B to +1.4 B . For Urdu, even though we didn't have manual alignments to tune hyperparameters, we got significant gains over a good baseline. This is promising for languages that do not have any manually aligned data.

Ideally, one would want to tune α and β to maximize B . However, this is prohibitively expensive, especially if we must tune them separately in each alignment direction before symmetrization. We ran some contrastive experiments to investigate the impact of hyperparameter tuning on translation quality. For the smaller Arabic-English corpus, we symmetrized all combinations of the two top-scoring alignments (according to F1) in each direction, yielding four sets of alignments. Table 4 shows B scores for translation models learned from these alignments. Unfortunately, we find that optimizing F1 is not optimal for B —using the second-best alignments yields a further improvement of 0.5 B on the NIST 2009 data, which is statistically significant ($p < 0.05$).

4 Related Work

Schoenemann (2011a), taking inspiration from Bodrumlu et al. (2009), uses integer linear programming to optimize IBM Model 1–2 and the HMM with the ℓ_0-norm. This method, however, does not outperform GIZA++. In later work, Schoenemann (2011b) used projected gradient descent for the ℓ_1-norm. Here, we have adopted his use of projected gradient descent, but using a smoothed ℓ_0-norm.

Liang et al. (2006) show how to train IBM models in both directions simultaneously by adding a term to the log-likelihood that measures the agreement between the two directions. Graça et al. (2010) explore modifications to the HMM model that encourage bijectivity and symmetry. The modifications take the form of constraints on the posterior distribution over alignments that is computed during the E-step. Mermer and Saraçlar (2011) explore a Bayesian version of IBM Model 1, applying sparse Dirichlet priors to t. However, because this method requires the use of Monte Carlo methods, it is not clear how well it can scale to larger datasets.

5 Conclusion

We have extended the IBM models and HMM model by the addition of an ℓ_0 prior to the word-to-word translation model, which compacts the word-to-word translation table, reducing overfitting, and, in particular, the "garbage collection" effect. We have shown how to perform MAP-EM with this prior efficiently, even for large datasets. The method is implemented as a modification to the open-source toolkit GIZA++, and we have shown that it significantly improves translation quality across four different language pairs. Even though we have used a small set of gold-standard alignments to tune our hyperparameters, we found that performance was fairly robust to variation in the hyperparameters, and translation performance was good even when gold-standard alignments were unavailable. We hope that our method, due to its simplicity, generality, and effectiveness, will find wide application for training better statistical translation systems.

Acknowledgments

We are indebted to Thomas Schoenemann for initial discussions and pilot experiments that led to this work, and to the anonymous reviewers for their valuable comments. We thank Jason Riesa for providing the Arabic-English and Chinese-English hand-aligned data and the alignment visualization tool, and Chris Dyer for the Czech-English hand-aligned data. This research was supported in part by DARPA under contract DOI-NBC D11AP00244 and a Google Faculty Research Award to L. H.

References

Andrew Barron, Jorma Rissanen, and Bin Yu. 1998. The minimum description length principle in coding and modeling. *IEEE Transactions on Information Theory*, 44(6):2743–2760.

Dimitri P. Bertsekas. 1999. *Nonlinear Programming*. Athena Scientific.

Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer.

Tugba Bodrumlu, Kevin Knight, and Sujith Ravi. 2009. A new objective function for word alignment. In *Proceedings of the NAACL HLT Workshop on Integer Linear Programming for Natural Language Processing*.

Ondřej Bojar and Magdalena Prokopová. 2006. Czech-English word alignment. In *Proceedings of LREC*.

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*, 19:263–311.

David Chiang, Yuval Marton, and Philip Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *Proceedings of EMNLP*.

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–208.

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Computational Linguistics*, 39(4):1–38.

John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. 2008. Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *Proceedings of ICML*.

Chris Dyer, Jonathan H. Clark, Alon Lavie, and Noah A. Smith. 2011. Unsupervised word alignment with arbitrary features. In *Proceedings of ACL*.

Alexander Fraser and Daniel Marcu. 2007. Measuring word alignment quality for statistical machine translation. *Computational Linguistics*, 33(3):293–303.

João V. Graça, Kuzman Ganchev, and Ben Taskar. 2010. Learning tractable word alignment models with complex constraints. *Computational Linguistics*, 36(3):481–504.

Philipp Koehn, Franz Joseph Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of NAACL*.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP*.

Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *Proceedings of HLT-NAACL*.

Coşkun Mermer and Murat Saraçlar. 2011. Bayesian word alignment for statistical machine translation. In *Proceedings of ACL HLT*.

Robert C. Moore. 2004. Improving IBM word-alignment Model 1. In *Proceedings of ACL*.

Robert Moore. 2005. A discriminative framework for bilingual word alignment. In *Proceedings of HLT-EMNLP*.

Franz Joseph Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30:417–449.

Jason Riesa and Daniel Marcu. 2010. Hierarchical search for word alignment. In *Proceedings of ACL*.

Thomas Schoenemann. 2011a. Probabilistic word alignment under the L_0-norm. In *Proceedings of CoNLL*.

Thomas Schoenemann. 2011b. Regularizing mono- and bi-word models for word alignment. In *Proceedings of IJCNLP*.

Ben Taskar, Lacoste-Julien Simon, and Klein Dan. 2005. A discriminative matching approach to word alignment. In *Proceedings of HLT-EMNLP*.

Ashish Vaswani, Adam Pauls, and David Chiang. 2010. Efficient optimization of an MDL-inspired objective function for unsupervised part-of-speech tagging. In *Proceedings of ACL*.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *Proceedings of COLING*.

Ying Zhang, Stephan Vogel, and Alex Waibel. 2004. Interpreting BLEU/NIST scores: How much improvement do we need to have a better system? In *Proceedings of LREC*.