Towards a Convex HMM Surrogate for Word Alignment

Andrei Arsene Simion
Columbia University *
New York, NY, 10011
aas2148@columbia.edu

Michael Collins
Columbia University†
Computer Science
New York, NY, 10027
mc3354@columbia.edu

Clifford Stein
Columbia University
IEOR Department
New York, NY, 10027
cs2035@columbia.edu

Abstract

Among the alignment models used in statistical machine translation (SMT), the hidden Markov model (HMM) is arguably the most elegant: it performs consistently better than IBM Model 3 and is very close in performance to the much more complex IBM Model 4. In this paper we discuss a model which combines the structure of the HMM and IBM Model 2. Using this surrogate, our experiments show that we can attain a similar level of alignment quality as the HMM model implemented in GIZA++ (Och and Ney, 2003). For this model, we derive its convex relaxation and show that it too has strong performance despite not having the local optima problems of non-convex objectives. In particular, the word alignment quality of this new convex model is significantly above that of the standard IBM Models 2 and 3, as well as the popular (and still non-convex) IBM Model 2 variant of (Dyer et al., 2013).

1 Introduction

The IBM translation models are widely used in modern statistical translation systems. Typically, one seeds more complex models with simpler models, and the parameters of each model are estimated through an Expectation Maximization (EM) procedure. Among the IBM Models, perhaps the most elegant is the HMM model (Vogel et al., 1996). The HMM is the last model whose expectation step is both exact and simple, and it attains a level of accuracy that is very close to the results achieved by much more complex models. In particular, experiments have shown that IBM Models 1, 2, and 3 all perform worse than the HMM and Model 4 benefits greatly from being seeded by the HMM (Och and Ney, 2003).

In this paper we make the following contributions:

- We derive a new alignment model which combines the structure of the HMM and IBM Model 2 and show that its performance is very close to that of the HMM. There are several reasons why such a result would be of value (for more on this, see (Simion et al., 2013) and (Simion et al., 2015a), for example).
- *The main goal of this work is not to eliminate highly non-convex models such as the HMM entirely but, rather, to develop a new, powerful, convex alignment model and thus push the boundary of these theoretically justified techniques further.* Building on the work of (Simion et al., 2015a), we derive a convex relaxation for the new model and show that its performance is close to that of the HMM. Although it does not beat the HMM, the new convex model improves upon the standard IBM Model 2 significantly. Moreover, the convex relaxation also performs better than the strong IBM 2 variant FastAlign (Dyer et al., 2013), IBM Model 3, and the other available convex alignment models detailed in (Simion et al., 2015a) and (Simion et al., 2013).
- We derive a parameter estimation algorithm for

*Currently at Google.
†Currently on leave at Google.

Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 531–540,
Austin, Texas, November 1-5, 2016.

new model and its convex relaxation based on the EM algorithm. Our model has both HMM emission probabilities and IBM Model 2's distortions, so we can use Model 2 to seed both the model's lexical and distortion parameters. For the convex model, we need not use any initialization heuristics since the EM algorithm we derive is guaranteed to converge to a local optima that is also global.

The goal of our work is to present a model which is convex and has state of the art empirical performance. Although one step of this task was achieved for IBM Model 2 (Simion et al., 2015a), our target goal deals with a much more local-optima-laden, non-convex objective. Finally, whereas IBM 2 in some ways leads to a clear method of attack, we will discuss why the HMM presents challenges that require the insertion of this new surrogate.

Notation. We adopt the notation introduced in (Och and Ney, 2003) of having $1^m 2^n$ denote the training scheme of m IBM Model 1 EM iterations followed by initializing Model 2 with these parameters and running n IBM Model 2 EM iterations. We denote by H the HMM and note that it too can be seeded by running Model 1 followed by Model 2. Additionally, we denote our model as 2_{H}, and note that it has distortion parameters like IBM Model 2 and emission parameters like that of the HMM. Under this notation, we let $1^m 2^n 2^o_{\text{H}}$ denote running Model 1 for m iterations, then Model 2 for n iteration, and then finally our Model for o iterations. As before, we are seeding from the more basic to the more complex model in turn. We denote the convex relaxation of 2_{H} by 2_{HC}. Throughout this paper, for any integer N, we use $[N]$ to denote $\{1 \dots N\}$ and $[N]_0$ to denote $\{0 \dots N\}$. Finally, in our presentation, "convex function" means a function for which a local maxima also global, for example, $f(x) = -x^2$.

2 IBM Models 1 and 2 and the HMM

In this section we give a brief review of IBM Models 1, 2, and the HMM, as well as the the optimization problems arising from these models. The standard approach for optimization within these latent variable models is the EM algorithm.

Throughout this section, and the remainder of the paper, we assume that our set of training examples is $(e^{(k)}, f^{(k)})$ for $k = 1 \dots n$, where $e^{(k)}$ is the k'th English sentence and $f^{(k)}$ is the k'th French sentence. Following standard convention, we assume the task is to translate from *French* (the "source" language) into *English* (the "target" language) [1]. We use E to denote the English vocabulary (set of possible English words), and F to denote the French vocabulary. The k'th English sentence is a sequence of words $e^{(k)}_1 \dots e^{(k)}_{l_k}$ where l_k is the length of the k'th English sentence, and each $e^{(k)}_i \in E$; similarly the k'th French sentence is a sequence $f^{(k)}_1 \dots f^{(k)}_{m_k}$, where m_k is the length of the k'th French sentence, and each $f^{(k)}_j \in F$. We define $e^{(k)}_0$ for $k = 1 \dots n$ to be a special NULL word (note that E contains the NULL word).

For each English word $e \in E$, we will assume that $D(e)$ is a *dictionary* specifying the set of possible French words that can be translations of e. The set $D(e)$ is a subset of F. In practice, $D(e)$ can be derived in various ways; in our experiments we simply define $D(e)$ to include all French words f such that e and f are seen in a translation pair.

Given these definitions, the IBM Model 2 optimization problem is presented in several sources, for example, (Simion et al., 2013). The parameters in this problem are $t(f|e)$ and $d(i|j,l,m)$. The objective function for IBM Model 2 is then the log-likelihood of the training data; we can simplify the log-likelihood (Koehn, 2008) as

$$\frac{1}{n}\sum_{k=1}^{n}\sum_{j=1}^{m_k}\log p(f^{(k)}_j|e^{(k)}) \,,$$

where

$$p(f^{(k)}_j|e^{(k)}) = \sum_{i=0}^{l_k} t(f^{(k)}_j|e^{(k)}_i)d(i|j,l_k,m_k) \,.$$

[1] Technically, in most standard sources (Koehn, 2008), this goes as follows: when we want to translate from *French* to *English* we note that $p(e|f) \propto p(f|e)p(e)$ by Bayes's Theorem. When translating, the alignment models we consider are concerned with modeling $p(f|e)$ while the rest of the translation is handled by the language model $p(e)$. Therefore, in the context of the original task, we have that English is the target language while French is the source. However, for the sake of clarity, we emphasize that the alignment models we study are concerned with the development of $p(f|e)$.

This last simplification is crucial as it allows for a simple multinomial EM implementation, and can be done for IBM Model 1 as well (Koehn, 2008). Furthermore, the ability to write out the marginal likelihood per sentence in this manner has seen other applications: it was crucial, for example, in deriving a convex relaxation of IBM Model 2 and solving the new problem using subgradient methods (Simion et al., 2013).

An improvement on IBM Model 2, called the HMM alignment model, was introduced by Vogel et al (Vogel et al., 1996). For this model, the distortion parameters are replaced by *emission* parameters $d(a_j|a_{j-1}, l)$. These emission parameters specify the probability of the next alignment variable for the j^{th} target word is a_j, given that the previous source word was aligned to a target word whose position was a_{j-1} in a target sentence with length of l. The objective of the HMM is given by

$$\frac{1}{n}\sum_{k=1}^{n}\sum_{a_1^{(k)}\ldots a_{m_k}^{(k)}}\log\prod_{j=1}^{m_k} t(f_j^{(k)}|e_{a_j^k}^{(k)})d(a_j^{(k)}|a_{j-1}^{(k)}, l_k)$$

and we present this in Fig 1. We note that unlike IBM Model 2, we cannot simplify the exponential sum within the log-likelihood of the HMM, and so EM training for this model requires the use of a special EM implementation knows as the Baum-Welch algorithm (Rabiner and Juang., 1986).

Once these models are trained, each model's highest probability (Viterbi) alignment is computed. For IBM Models 1 and 2, the Viterbi alignment splits easily (Koehn, 2008). For the HMM, dynamic programming is used (Vogel et al., 1996). Although it is non-convex and thus its initialization is important, the HMM is the last alignment model in the classical setting that has an exact EM procedure (Och and Ney, 2003): from IBM Model 3 onwards heuristics are used within the expectation and maximization steps of each model's associated EM procedure.

3 Distortion and emission parameter structure

The structure of IBM Model 2's distortion parameters and the HMM's emission parameters is important and is used in our model as well, so we detail this here. We are using the roughly same structure as (Liang et al., 2006) and (Dyer et al., 2013): the distortions and emissions of our model are parametrized by forcing the model to concentrate its alignments on the diagonal.

Input: Define E, F, $(e^{(k)}, f^{(k)}, l_k, m_k)$ for $k = 1 \ldots n$, $D(e)$ for $e \in E$ as in Section 2.

Parameters:

- A parameter $t(f|e)$ for each $e \in E, f \in D(e)$.
- A parameter $d(i|i, l_k)$ for each $i \in [l_k]_0$, $i' \in [l_k]_0$.

Constraints:

$$\forall e \in E, f \in D(e), t(f|e) \geq 0 \quad (1)$$

$$\forall e \in E, \sum_{f \in D(e)} t(f|e) = 1 \quad (2)$$

$$\forall i, i' \in [l_k]_0, d(i'|i, l_k) \geq 0 \quad (3)$$

$$\forall i \in [l_k]_0, \sum_{i' \in [l_k]_0} d(i'|i, l_k) = 1 \quad (4)$$

Objective: Maximize

$$\frac{1}{n}\sum_{k=1}^{n}\sum_{a_1^{(k)}\ldots a_{m_k}^{(k)}}\log\prod_{j=1}^{m_k} t(f_j^{(k)}|e_{a_j^k}^{(k)})d(a_j^{(k)}|a_{j-1}^{(k)}, l_k)$$

with respect to the $t(f|e)$ parameters $d(i'|i, l)$.

Figure 1: The HMM Optimization Problem

3.1 Distortion Parameters for IBM2

Let $\lambda > 0$. For the IBM Model 2 distortions we set the NULL word probability as $d(0|j, l, m) = p_0$, where $p_0 = \frac{1}{l+1}$ and note that this will generally depend on the target sentence length within a bitext training pair that we are considering. For $i \neq 0$ which satisfies we set

$$d(i|j, l, m) = \frac{(1 - p_0)e^{-\lambda|\frac{i}{l} - \frac{j}{m}|}}{Z_\lambda(j, l, m)},$$

where $Z_\lambda(j, l, m)$ is a normalization constant as in (Dyer et al., 2013).

3.2 Emission Parameters for HMM

Let $\theta > 0$. For the HMM emissions we first set the NULL word generation to $d(0|i, l) = p_0$, with

$p_0 = \frac{1}{l+1}$. For target word position $i, i' \neq 0$, we set

$$d(i'|i,l) = \frac{(1-p_0)e^{-\theta|\frac{i'}{l}-\frac{i}{l}|}}{Z_\theta(i,l,m)} ,$$

where $Z_\theta(i,l,m)$ is a suitable normalization constant. Lastly, if $i = 0$ so that we are jumping from the NULL word onto a possibly different word, we set $d(i'|0,l) = p_0$. Aside from making the NULL word have uniform jump probability, the above emission parameters are modeled to favor a jumping to an adjacent English word.

4 Combining IBM Model 2 and the HMM

In deriving the new HMM surrogate, our main goal was to allow the current alignment to know as much as possible about the previous alignment variable and still have a likelihood that factors as that of IBM Model 2 (Simion et al., 2013; Koehn, 2008). We combine IBM Model 2 and the HMM by incorporating the generation of words using the structure of both models. The model we introduce, IBM2-HMM, is displayed in Fig 2.

Consider a target-source sentence pair (e, f) with $|e| = l$ and $|f| = m$. For source sentence positions j and $j+1$ we have source words f_j and f_{j+1} and we assign a joint probability involving the alignments a_j and a_{j+1} as:

$$q(j, a_j, a_{j+1}, l, m) \quad = \tag{5}$$
$$t(f_j|e_{a_j})d(a_j|j,l,m)t(f_{j+1}|e_{a_{j+1}})d(a_{j+1}|a_j,l) . \tag{6}$$

From the equation above, we note that we use the IBM Model 2's word generation method for position j and the HMM generative structure for position $j+1$. The generative nature of the above procedure introduces dependency between adjacent words two at a time. Since we want to mimic the HMM's structure as much as possible, we devise our likelihood function to mimic the HMM's dependency between alignments using q. Essentially, we move the source word position j from 1 to m and allow for overlapping terms when $j \in \{2, \ldots, m-1\}$. In what follows, we describe this representation in detail.

The likelihood in Eq. 16 is actually the sum of two likelihoods which use equations Eq. 5 and 6 repeatedly. To this end, we will discuss how our objective is actually

$$\frac{1}{n}\sum_{k=1}^{n} \log \sum_{a^{(k)}, b^{(k)}} p(f^{(k)}, a^{(k)}, b^{(k)}|e^{(k)}) , \tag{7}$$

where $a^{(k)}$ and $b^{(k)}$ both are alignment vectors whose components are independent and can take on any values in $[l_k]_0$. To see how $p(f, a, b|e)$ comes about, note that we could generate the sentence f by generating pairs $(1,2), (3,4), (5,6), \ldots$ using equations Eqs. 5 and 6 for each pair. Taking all this together, the upshot of our discussion is that generating the pair (e, f) in this way gives us that the likelihood for an alignment a would be given by:

$$p_1(f, a|e) = \prod_{j \text{ odd}}^{m-1} q(j, a_j, a_{j+1}, l, m) . \tag{8}$$

Using the same idea as above, we could also skip the first target word position and generate pairs $(2,3), (4,5), \ldots$ using Eqs. 5 and 6. Under this second generative method, the joint probability for f and alignment b is:

$$p_2(f, b|e) = \prod_{j \text{ even}}^{m-1} q(j, b_j, b_{j+1}, l, m) , \tag{9}$$

Finally, we note that if m is even we do not generate f_1 and f_m under p_2 but we do generate these words under p_1. Similarly, if m is odd we do not generate f_1 under p_2 and we do not generate f_m under p_1; however in this case as in the first, we still generate these missing words under the other generative method. Using $p(f, a, b|e) = p_1(f, a|e)p_2(f, b|e)$ and factoring the log-likelihood as in IBM Model 1 and 2 (Koehn, 2008), we get the log-likelihood in Fig 2. Finally, we note that our model's log-likelihood could be viewed as the sum of the log-likelihoods of a model which generates (e, f) using p_1 and another model which generates sentences using p_2. These models share parameters but generate words using different recipes, as discussed above.

5 The parameter estimation for IBM2-HMM

To fully optimize our new model (over t, λ, and θ), we can use an EM algorithm in the same fashion as

Input: Define E, F, $(e^{(k)}, f^{(k)}, l_k, m_k)$ for $k = 1 \ldots n$, $D(e)$ for $e \in E$ as in Section 2.

Parameters:

- A parameter $t(f|e)$ for each $e \in E, f \in D(e)$.
- A parameter $\lambda > 0$ for distortion centering.
- A parameter $\theta > 0$ for emission centering.

Constraints:

$$\forall e \in E, f \in D(e), t(f|e) \geq 0 \quad (10)$$

$$\forall e \in E, \sum_{f \in D(e)} t(f|e) = 1 \quad (11)$$

$$\forall i \in [l_k]_0, j \in [m_k], d(i|j, l_k, m_k) \geq 0 \quad (12)$$

$$\forall j \in [m_k], \sum_{i \in [l_k]_0} d(i|j, l_k, m_k) = 1 \quad (13)$$

$$\forall i, i' \in [l_k]_0, d(i'|i, l_k) \geq 0 \quad (14)$$

$$\forall i \in [l_k]_0, \sum_{i' \in [l_k]_0} d(i'|i, l_k) = 1 \quad (15)$$

Objective: Maximize

$$\frac{1}{n} \sum_{k=1}^{n} \sum_{j=1}^{m_k - 1} \log \sum_{i=0}^{l_k} \sum_{i'=0}^{l_k} q(j, i, i', l_k, m_k) \quad (16)$$

with respect to the parameters $t(f|e)$, $d(i'|i, l)$ $d(i|j, l, m)$, and $q(j, i, i', l_k, m_k)$ set as

$$t(f_j^{(k)}|e_i^{(k)}) d(i|j, l, m) t(f_{j+1}^{(k)}|e_{i'}) d(i'|i, l) \quad (17)$$

Figure 2: The IBM2-HMM Optimization Problem. We use equation (5) within the likelihood definition.

(Dyer et al., 2013). Specifically, for the model in question the EM algorithm still applies but we have to use a gradient-based algorithm within the learning step. On the other hand, since such a gradient-based method introduces the necessary complication of a learning rate, we could also optimize the objective by picking θ and λ via cross-validation and using a multinomial EM algorithm for the learning of the lexical t terms. For our experiments, we opted for this simpler choice: we derived a multinomial EM algorithm and cross-validated the centering parameters for the distortion and emission terms. With λ and θ fixed, the derivation of this algorithm is very similar to the one used for IBM2-HMM's convex relaxation and this uses the path discussed in (Simion et al., 2015a) and (Simion et al., 2015b). We detail the EM algorithm for the convex relaxation below.

6 A Convex HMM Surrogate

In this section we detail a procedure to get a convex relaxation for IBM2-HMM. Let $(\mathbf{t}, \mathbf{d})$ be all the parameters of the HMM. As a first step in getting a convex HMM, one could follow the path developed in (Simion et al., 2015a) and directly replace the HMM's objective terms

$$\prod_{j=1}^{m_k} t(f_j^{(k)}|e_{a_j^k}^{(k)}) d(a_j^{(k)}|a_{j-1}^{(k)}, l_k)$$

by

$$\left(\prod_{j=1}^{m_k} t(f_j^{(k)}|e_{a_j^k}^{(k)}) d(a_j^{(k)}|a_{j-1}^{(k)}, l_k)\right)^{\frac{1}{2m_k}} .$$

In particular, the geometric mean function $h(x_1, \ldots, h_{2m_k}) = (\prod_{j=1}^{2m_k} x_j)^{\frac{1}{2m_k}}$ is convex ((Boyd and Vandenberghe, 2004)) and, for a given sentence pair $(e^{(k)}, f^{(k)})$ with alignment $a^{(k)}$ we can find a projection matrix $\mathbf{P}$ so that $\mathbf{P}(\mathbf{t}, \mathbf{d}) = (\tilde{\mathbf{t}}, \tilde{\mathbf{d}})$ where $\tilde{\mathbf{t}} = \{t(f_j^{(k)}|e_{a_j^k}^{(k)})\}_{j=1}^{m_k}$ and $\tilde{\mathbf{d}} = \{d(a_j^{(k)}|a_{j-1}^{(k)}, l_k)\}_{j=1}^{m_k}$ are exactly the parameters used in the term above (in particular, $\mathbf{t}, \mathbf{d}$ are the set of all parameters while $\tilde{\mathbf{t}}, \tilde{\mathbf{d}}$ are the set of parameters for the specific training pair k; $\mathbf{P}$ projects from the full space onto only the parameters used for training pair k). Given this, we then have that $g(\mathbf{t}, \mathbf{d}) = h(\mathbf{P}(\mathbf{t}, \mathbf{d})) = h(\tilde{\mathbf{t}}, \tilde{\mathbf{d}})$ is convex and, by composition, so is $\log g(\mathbf{t}, \mathbf{d})$ (see (Simion et al., 2015a; Boyd and Vandenberghe, 2004) for details; the main idea lies in the fact that as linear transformations preserve convexity, so do compositions of convex functions with increasing convex functions such as $\log$). Finally, if we run this plan for all terms in the objective, the new objective is convex since it is the sum of convex functions (the new optimization problem is convex as it has linear constraints). Although this gives a convex program, we observed that the powers being so small made the optimized probabilities very uninformative (i.e. uniform). The above makes sense: no matter what the parameters are, we will easily get the 1 we seek

for each term in the objective since all terms are taken to a low ($\frac{1}{2m_k}$) power .

Since this direct relaxation does not yield fruit, we next could turn to our model. Developing its relaxation in the vein of (Simion et al., 2015a), we could be to let $d(i|j,l,m)$ and $d(i'|i,l)$ be multinomial probabilities (that is, these parameters would not have centering parameters λ and θ and would be just standard probabilities as in the GIZA++ versions of the HMM and IBM Model 2 (Och and Ney, 2003)) and replace all the terms $q(j,i',i,l,m)$ in (16) by $(q(j,i',i,l,m))^{\frac{1}{4}}$. Although this method is feasible, experiments showed that the relaxation is not very competitive and performs on par with IBM Model 2; this relaxation is far in performance from the HMM even though we are relaxing (only) the product of 4 terms (lastly, we mention that we tried other variants were we replaced $d(i|j,l,m)d(i'|i,l)$ by $d(i,i'|j,l,m)$ so that we would have only three terms; unfortunately, this last attempt also produced parameters that were "too uniform").

The above analysis motivates why we defined our model as we did: we now have only two terms to relax. In particular, to rectify the above, we left in place the structure discussed in Section 3 and made λ and θ be tuning parameters which we can cross-validate for on a small held-out data set. This last constraint effectively removes the distortion and emission parameters from the model but we still maintain the structural property of these parameters: we maintain their favoring the diagonal or adjacent alignment. To get the relaxation, we replace $q(j,i,i',l,m)$ by

$$p(j,i,i',l,m) \propto \sqrt{t(f_j^{(k)}|e_i^{(k)})t(f_{j+1}^{(k)}|e_{i'})}$$

and set the proportionality constant to be $d(i|j,l,m)d(i'|i,l)$. Using this setup we now have a convex objective to optimize over. In particular, we've formulated a convex relaxation of the IBM2-HMM problem which, like the Support Vector Machine, includes parameters that can be cross-validated over (Boyd and Vandenberghe, 2004).

Input: Define E, F, $(e^{(k)}, f^{(k)}, l_k, m_k)$ for $k = 1 \ldots n$, $D(e)$ for $e \in E$ as in Section 2. Pick $\lambda, \theta > 0$ as in Section 3 via cross-validation.

Parameters:

• A parameter $t(f|e)$ for each $e \in E, f \in D(e)$.

Constraints:

$$\forall e \in E, f \in D(e), t(f|e) \geq 0 \quad (18)$$

$$\forall e \in E, \sum_{f \in D(e)} t(f|e) = 1 \quad (19)$$

Objective: Maximize

$$\frac{1}{n}\sum_{k=1}^{n}\sum_{j=1}^{m_k-1}\log\sum_{i=0}^{l_k}\sum_{i'=0}^{l_k} p(j,i,i',l_k,m_k) \quad (20)$$

with respect to the parameters $t(f|e)$ and $p(j,i,i',l_k,m_k)$ set as

$$\sqrt{t(f_j^{(k)}|e_i^{(k)})}d(i|j,l,m)\sqrt{t(f_{j+1}^{(k)}|e_{i'})}d(i'|i,l)$$

Figure 3: The IBM2-HMM convex relaxation optimization problem. Note that the distortions $d(i|j,l,,m)$ and emissions $d(i'|i,l)$ are constants held fixed and parameterized by cross-validated parameters λ and θ as in Section 3.

7 An EM algorithm for the convex surrogate

The EM algorithm for the convex relaxation of our surrogate is given in Fig 4. As the model's objective is the sum of the objectives of two models generated by a multinomial rule, we can get a very succinct EM algorithm. For more details on this and a similar derivation, please refer to (Simion et al., 2015a), (Koehn, 2008) or (Simion et al., 2015b). For this algorithm, we again note that the distortion and emission parameters are constants so that the only estimation that needs to be conducted is on the lexical t terms.

To be specific, we have that the M step requires optimizing

$$\frac{1}{n}\sum_{k=1}^{n}\log\sum_{a^{(k)},b^{(k)}} q(a^{(k)},b^{(k)}|e^{(k)},f^{(k)})p(f^{(k)},a^{(k)},b^{(k)}|e^{(k)}).$$

In the above, we have that

$$q(a^{(k)}, b^{(k)} | e^{(k)}, f^{(k)})$$

are constants proportional to

$$\prod_{j=1}^{m_k - 1} \sqrt{t(f_j^{(k)} | e^{(k)}_{a_j^{(k)}}) t(f_{j+1}^{(k)} | e^{(k)}_{a_{j+1}^{(k)}})} \prod_{j=2}^{m_k} \sqrt{t(f_j^{(k)} | e^{(k)}_{b_j^{(k)}}) t(f_{j+1}^{(k)} | e^{(k)}_{b_{j+1}^{(k)}})}$$

and gotten through the E step. This optimization step is very similar to the regular Model 2 M step since the β drops down using $\log t^\beta = \beta \log t$; the exact same count-based method can be applied. The upshot of this is given in Fig 4; similar to the logic above for 2_{HC}, we can get the EM algorithm for 2_{H}.

8 Decoding methods for IBM2-HMM

When computing the optimal alignment we wanted to compare our model with the HMM as closely as possible. Because of this, the most natural method of evaluating the quality of the parameters would be to use the same rule as the HMM. Specifically, for a sentence pair (e, f) with $|e| = l$ and $|f| = m$, in HMM decoding we aim to find $(a_1 \dots a_m)$ which maximizes

$$\max_{a_1,\dots,a_m} \prod_{j=1}^{m} t(f_j | e_{a_j}) d(a_j | a_{j-1}, l).$$

As is standard, dynamic programming can now be used to find the Viterbi alignment. Although there are a number of ways we could define the optimal alignment, we felt that the above would be the best since it tests dependance between alignment variables and allows for easy comparison with the GIZA++ HMM. Finding the optimal alignment under the HMM setting is labelled "HMM" in Table 1.

We can also find the optimal alignment by taking the objective literally (see (Simion et al., 2014) for a similar argument dealing with the convex relaxation of IBM Model 2) and computing

$$\max_{a_1 \dots a_m} p_1(f, a|e) p_2(f, a|e).$$

Above, we are asking for the optimal alignment that yields the highest probability alignment through generating technique p_1 and p_2. This method of decoding is a lot like the HMM style and also relies on dynamic programming. In this case we have the recursion for Q_{Joint} given by

$$Q_{Joint}(1, i) = t(f_1 | e_i) d^2(i | 1, l, m) \ ,$$

$\forall i \in [l]_0$, and

$$Q_{Joint}(j, i') = t^2(f_j | e_{i'}) d(i' | j, l, m) M_{Joint}(j-1, i') \ ,$$

where $M_{Joint}(j-1, i')$ is

$$M_{Joint}(j-1, i') = \max_{i=0}^{l} \{ d(i' | i, l) Q_{Joint}(j-1, i) \} \ ,$$

$\forall\, 2 \le j \le m, \forall\, i' \in [l]_0$. The alignment results gotten by decoding with this method is labelled "Joint" in Table 1.

Input: Define E, F, $(e^{(k)}, f^{(k)}, l_k, m_k)$ for $k = 1 \dots n$, $D(e)$ for $e \in E$ as in Section 2. Two parameters $\lambda, \theta > 0$ picked by cross-validation so that the distortions and emissions are constants obeying the structure in Section 3. An integer T specifying the number of passes over the data.
Parameters:
• A parameter $t(f|e)$ for each $e \in E, f \in D(e)$.
Initialization:
• $\forall e \in E,\ f \in D(e)$, set $t(f|e) = \frac{1}{D(e)}$.
EM Algorithm: Expectation
for all $k = 1 \dots N$ **do**
  for all $j = 1 \dots m_k$ **do**
    $\delta = 0$
    $\Delta = 0$
    for all $i = 0 \dots l_k$ **do**
      for all $i' = 0 \dots l_k$ **do**
        $\delta[i, i'] = p(j, i', i, l_k, m_k)$
        $\Delta += \delta[i, i']$
    for all $i = 0 \dots l_k$ **do**
      for all $i' = 0 \dots l_k$ **do**
        $\delta[i, i'] = \frac{\delta[i, i']}{\Delta}$
        $counts(f_j^{(k)}, e_i^{(k)}) += \delta[i, i']$
        $counts(e_i^{(k)}) += \delta[i, i']$
        $counts(f_{j+1}^{(k)}, e_{i'}^{(k)}) += \delta[i, i']$
        $counts(e_{i'}^{(k)}) += \delta[i, i']$
EM Algorithm: Maximization
for all $e \in E$ **do**
  for all $f \in D(e)$ **do**
    $t(f|e) = \frac{counts(e,f)}{counts(e)}$
Output: t parameters.

Figure 4: Pseudocode for the EM algorithm of the IBM2-HMM's convex relaxation. As the distortion and emission parameters are constants, the algorithm is very similar to that of IBM Model 1.

9 Experiments

In this section we describe experiments using the IBM2-HMM optimization problem combined with the EM algorithm for parameter estimation.

9.1 Data Sets

We use data from the bilingual word alignment workshop held at HLT-NAACL 2003 (Michalcea and Pederson, 2003). We use the Canadian Hansards bilingual corpus, with 743,989 English-French sentence pairs as training data, 37 sentences of development data, and 447 sentences of test data (note that we use a randomly chosen subset of the original training set of 1.1 million sentences, similar to the setting used in (Moore, 2004)). The development and test data have been manually aligned at the word level, annotating alignments between source and target words in the corpus as either "sure" (S) or "possible" (P) alignments, as described in (Och and Ney, 2003). As is standard, we lower-cased all words before giving the data to GIZA++ and we ignored NULL word alignments in our computation of alignment quality scores.

9.2 Methodology

We test several models in our experiments. In particular, we empirically evaluate our models against the GIZA++ IBM Model 3 and HMM, as well as the FastAlign IBM Model 2 implementation of (Dyer et al., 2013) that uses Variational Bayes. For each of our models, we estimated parameters and got alignments in turn using models in the source-target and target-source directions; using the same setup as (Simion et al., 2013), we present the gotten intersected alignments. In training, we employ the standard practice of initializing non-convex alignment models with simpler non-convex models. In particular, we initialize, the GIZA++ HMM with IBM Model 2, IBM Model 2 with IBM Model 1, and IBM2-HMM and IBM Model 3 with IBM Model 2 preceded by Model 1. Lastly, for FastAlign, we initialized all parameters uniformly since this empirically was a more favorable initialization, as discussed in (Dyer et al., 2013).

We measure the performance of the models in terms of *Precision*, *Recall*, *F-Measure*, and *AER* using only sure alignments in the definitions of the first three metrics and sure and possible alignments in the definition of *AER* , as in (Simion et al., 2013) and (Marcu et al., 2006). For our experiments, we report results in both AER (lower is better) and F-Measure (higher is better) (Och and Ney, 2003).

Table 1 shows the alignment summary statistics for the 447 sentences present in the Hansard test data. We present alignments quality scores using either the FastAlign IBM Model 2, the GIZA++ HMM, and our model and its relaxation using either the "HMM" or "Joint" decoding. First, we note that in deciding the decoding style for IBM2-HMM, the HMM method is better than the Joint method. We expected this type of performance since HMM decoding introduces positional dependance among the entire set of words in the sentence, which is shown to be a good modeling assumption (Vogel et al., 1996).

From the results in Table 1 we see that the HMM outperforms all other models, including IBM2-HMM and its convex relaxation. However, IBM2-HMM is not far in AER performance from the HMM and both it and its relaxation do better than FastAlign or IBM Model 3 (the results for IBM Model 3 are not presented; a one-directional English-French run of $1^5 2^5 3^{15}$ gave AER and F-Measure numbers of 0.1768 and 0.6588, respectively, and this was behind both the IBM Model 2 FastAlign and our models).

As a further set of experiments, we also appended an IBM Model 1 or IBM Model 2 objective to our models's original objectives, so that the constraints and parameters are the same but now we are maximizing the average of two log-likelihoods. With regard to the EM optimization, we would only need to add another δ parameter: we'd now have probabilities $\delta_1[i] \propto t(f_j^{(k)}|e_i^{(k)})d(i|j,l_l,m_k)$ (this is for IBM Model 2 smoothing; we have $d = 1$ for IBM 1 smoothing) and $\delta_2[i,i'] \propto p(j,i,i'.l_k,m_k)$ in the EM Algorithm that results (for more, see (Simion et al., 2015a)). We note that the appended IBM Model 2 objective is still convex if we fix the distortions' λ parameter and then optimize for the t parameters via EM (thus, model 2_{HC} is still convex). For us, there were significant gains, especially in the convex model. The results for all these experiments are shown in Table 2, with IBM 2 smoothing for the convex model displayed in the rightmost column.

Finally, we also tested our model in the full

Training	$1^5 2^{10}_{\text{H}}$	$1^5 2^{10}_{\text{H}}$	2^{10}_{HC}	2^{10}_{HC}	FA^{10}	$1^5 2^5 \text{H}^{10}$
Decoding	HMM	Joint	HMM	Joint	IBM2	HMM
Iteration			AER			
1	0.0956	0.1076	0.1538	0.1814	0.5406	0.1761
2	0.0884	0.0943	0.1093	0.1343	0.1625	0.0873
3	0.0844	0.0916	0.1023	0.1234	0.1254	0.0786
4	0.0828	0.0904	0.0996	0.1204	0.1169	0.0753
5	0.0808	0.0907	0.0992	0.1197	0.1131	0.0737
6	0.0804	0.0906	0.0989	0.1199	0.1128	0.0719
7	0.0795	0.0910	0.0986	0.1197	0.1116	0.0717
8	0.0789	0.0900	0.0988	0.1195	0.1086	0.0725
9	0.0793	0.0904	0.0986	0.1195	0.1076	0.0738
10	0.0793	0.0902	0.0986	0.1195	0.1072	0.0734
Iteration			F-Measure			
1	0.7829	0.7797	0.7199	0.6914	0.2951	0.7219
2	0.7854	0.7805	0.7594	0.7330	0.7111	0.8039
3	0.7899	0.7806	0.7651	0.7427	0.7484	0.8112
4	0.7908	0.7813	0.7668	0.7457	0.7589	0.8094
5	0.7928	0.7806	0.7673	0.7461	0.7624	0.8058
6	0.7928	0.7807	0.7678	0.7457	0.7630	0.8056
7	0.7939	0.7817	0.7679	0.7457	0.7633	0.8046
8	0.7942	0.7814	0.7679	0.7458	0.7658	0.8024
9	0.7937	0.7813	0.7680	0.7457	0.7672	0.8007
10	0.7927	0.7816	0.7680	0.7457	0.7679	0.8010

Table 1: Alignment quality results for IBM2-HMM (2_{H}) and its convex relaxation (2_{HC}) using either HMM-style dynamic programming or "Joint" decoding. The first and last columns above are for the GIZA++ HMM initialized either with IBM Model 1 or Model 1 followed by Model 2. FA above refers to the improved IBM Model 2 (FastAlign) of (Dyer et al., 2013).

Training	$1^5 2^{10}_{\text{H}}$	$1^5 2^{10}_{\text{H}}$	2^{10}_{HC}	2^{10}_{HC}
Smoothing	IBM1	IBM2	IBM1	IBM2
Decoding	HMM	HMM	HMM	HMM
Iteration		AER		
1	0.1003	0.0958	0.1703	0.1482
2	0.0949	0.0890	0.1172	0.1057
3	0.0904	0.0840	0.1039	0.0955
4	0.0886	0.0816	0.0984	0.0927
5	0.0866	0.0795	0.0948	0.0894
6	0.0851	0.0794	0.0933	0.0888
7	0.0837	0.0790	0.0922	0.0886
8	0.0825	0.0788	0.0921	0.0880
9	0.0820	0.0785	0.0921	0.0881
10	0.0820	0.0777	0.0920	0.0881
Iteration		F-Measure		
1	0.7791	0.7817	0.7065	0.7251
2	0.7822	0.7839	0.7559	0.7637
3	0.7856	0.7897	0.7689	0.7740
4	0.7873	0.7923	0.7729	0.7760
5	0.7899	0.7938	0.7771	0.7782
6	0.7904	0.7943	0.7789	0.7788
7	0.7917	0.7946	0.7800	0.7791
8	0.7928	0.7944	0.7806	0.7795
9	0.7930	0.7941	0.7806	0.7797
10	0.7925	0.7947	0.7806	0.7796

Table 2: Alignment quality results for IBM2-HMM and its relaxation using IBM 1 and IBM 2 smoothing (in this case, "smoothing" means adding these log-likelihoods to the original objective as in (Simion et al., 2013). For the convex relaxation of IBM2-HMM, we can only smooth by adding in the convex IBM Model 1 objective, or by adding in an IBM Model 2 objective where the distortions are taken to be constants (these distortions are identical to the ones that are used within the relaxation itself and are cross-validated for optimal λ).

SMT pipeline using the cdec system (Dyer et al., 2013). For our experiments, we compared our models' alignments (gotten by training $1^5 2^5_{\text{H}}$ and 2^5_{HC}) against the alignments gotten by the HMM ($1^5 2^5 H^5$), IBM Model 4 ($1^5 H^5 3^3 4^3$), and FastAlign. Unfortunately, we found that all 4 systems led to roughly the same BLEU score of 40 on a Spanish-English training set of size 250000 which was a subset of version 7 of the Europarl dataset (Dyer et al., 2013). For our development and test sets, we used data each of size roughly 1800 and we preprocessed all data by considering only sentences of size less than 80 and filtering out sentences which had a very large (or small) ratio of target and source sentence lengths. Although the SMT results were not a success in that our gains were not significant, we felt that the experiments at least highlight that our model mimics the HMM's alignments even though its structure is much more local. Lastly, we in regards to the new convex model's performance, we observe much better alignment quality than any other convex alignment models in print, for example, (Simion et al., 2015a).

10 Conclusions and Future Work

Our work has explored some of the details of a new model which combines the structure of IBM Model 2 the alignment HMM Model. We've shown that this new model and its convex relaxation performs very close to the standard GIZA++ implementation of the HMM. Bridging the gap between the HMM and convex models proves difficult for a number of reasons (Guo and Schuurmans, 2007). In this paper, we have introduced a new set of ideas aimed at tightening this gap.

Acknowledgments

Andrei Simion was supported by a Google research award. Cliff Stein was partially supported by NSF grant CCF-1421161. We thank the reviewers for their insightful commentary and suggestions.

References

Steven Boyd and Lieven Vandenberghe. 2004. Convex Optimization. Cambridge University Press.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert. L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19:263-311.

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum Likelihood From Incomplete Data via the EM Algorithm. *Journal of the royal statistical society, series B*, 39(1):1-38.

Chris Dyer , Victor Chahuneau, Noah A. Smith. 2013. A Simple, Fast, and Effective Reparameterization of IBM Model 2. *In Proceedings of NAACL.*

Alexander Fraser and Daniel Marcu. 2007. Measuring Word Alignment Quality for Statistical Machine Translation. *Journal Computational Linguistics*, 33(3): 293-303.

Joao V. Graca, Kuzman Ganchev and Ben Taskar. 2007. Expectation Maximization and Posterior Constraints. *In Proceedings of NIPS.*

Yuhong Guo and Dale Schuurmans. 2007. Convex Relaxations of Latent Variable Training. *In Proceedings of NIPS*.

Simon Lacoste-Julien, Ben Taskar, Dan Klein, and Michael Jordan. 2008. Word Alignment via Quadratic Assignment. *In Proceedings of the HLT-NAACL.*

Phillip Koehn. 2004. Statistical Significance Tests for Machine Translation Evaluation. *In Proceedings of the EMNLP*.

Phillip Koehn. 2008. Statistical Machine Translation. Cambridge University Press.

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

Daniel Marcu, Wei Wang, Abdessamad Echihabi, and Kevin Knight. 2006. SPMT: Statistical Machine Translation with Syntactified Target Language Phrases. *In Proceedings of the EMNLP*.

Rada Michalcea and Ted Pederson. 2003. An Evaluation Exercise in Word Alignment. *HLT-NAACL 2003: Workshop in building and using Parallel Texts: Data Driven Machine Translation and Beyond*.

Robert C. Moore. 2004. Improving IBM Word-Alignment Model 1. *In Proceedings of the ACL.*

Stephan Vogel, Hermann Ney and Christoph Tillman. 1996. HMM-Based Word Alignment in Statistical Translation. *In Proceedings of COLING.*

Franz Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational-Linguistics*, 29(1): 19-52.

L.R. Rabiner and B.H. Juang. 1986. An Introduction to Hidden Markov Models. *In IEEE ASSP Magazine*.

Andrei Simion, Michael Collins and Cliff Stein. 2013. A Convex Alternative to IBM Model 2. *In Proceedings of EMNLP*.

Andrei Simion, Michael Collins and Cliff Stein. 2013. Some Experiments with a Convex IBM Model 2. *In Proceedings of EACL.*

Andrei Simion, Michael Collins and Cliff Stein. 2015. A Family of Latent Variable Convex Relaxations for IBM Model 2. *In Proceedings of the AAAI*.

Andrei Simion, Michael Collins and Cliff Stein. 2015. On a Strictly Concave IBM Model 1. *In Proceedings of EMNLP*.

Kristina Toutanova and Michel Galley. 2011. Why Initialization Matters for IBM Model 1: Multiple Optima and Non-Strict Convexity. *In Proceedings of the ACL.*

Ashish Vaswani, Liang Huang and David Chiang. 2012. Smaller Alignment Models for Better Translations: Unsupervised Word Alignment with the $L0$-norm. *In Proceedings of the ACL.*