On a Strictly Convex IBM Model 1

Andrei Simion
Columbia University
New York, NY, 10027
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

IBM Model 1 is a classical alignment model. Of the first generation word-based SMT models, it was the only such model with a concave objective function. For concave optimization problems like IBM Model 1, we have guarantees on the convergence of optimization algorithms such as Expectation Maximization (EM). However, as was pointed out recently, the objective of IBM Model 1 is not strictly concave and there is quite a bit of alignment quality variance within the optimal solution set. In this work we detail a strictly concave version of IBM Model 1 whose EM algorithm is a simple modification of the original EM algorithm of Model 1 and does not require the tuning of a learning rate or the insertion of an l_2 penalty. Moreover, by addressing Model 1's shortcomings, we achieve AER and F-Measure improvements over the classical Model 1 by over 30%.

1 Introduction

The IBM translation models were introduced in (Brown et al., 1993) and were the first-generation Statistical Machine Translation (SMT) systems. In the current pipeline, these word-based models are the seeds for more sophisticated models which need alignment tableaus to start their optimization procedure. Among the original IBM Models, only IBM Model 1 can be formulated as a concave optimization problem. Recently, there has been some research on IBM Model 2 which addresses either the model's non-concavity (Simion et al., 2015) or over parametrization (Dyer et al., 2013). We make the following contributions in this paper:

- We utilize and expand the mechanism introduced in (Simion et al., 2015) to construct *strictly* concave versions of IBM Model 1[1]. As was shown in (Toutanova and Galley, 2011), IBM Model 1 is not a strictly concave optimization problem. What this means in practice is that although we can initialize the model with random parameters and get to the same objective cost via the EM algorithm, there is quite a bit of alignment quality variance within the model's optimal solution set and ambiguity persists on which optimal solution truly is the best. Typically, the easiest way to make a concave model strictly concave is to append an l_2 regularizer. However, this method does not allow for seamless EM training: we have to either use a learning-rate dependent gradient based algorithm directly or use a gradient method within the M step of EM training. In this paper we show how to get via a simple technique an infinite supply of models that still allows a straightforward application of the EM algorithm.

- As a concrete application of the above, we detail a very simple strictly concave version of IBM Model 1 and study the performance of different members within this class. Our strictly concave models combine some of the elements of word association and positional dependance as in IBM Model 2 to yield a significant model improvement. Furthermore,

*Currently on leave at Google Inc. New York.

[1]Please refer as needed to the Appendix for examples and definitions of convexity/concavity and strict convexity/concavity.

we now have guarantees that the solution we find is unique.

- We detail an EM algorithm for a subclass of strictly concave IBM Model 1 variants. The EM algorithm is a small change to the original EM algorithm introduced in (Brown et al., 1993).

Notation. Throughout this paper, for any positive integer N, we use $[N]$ to denote $\{1 \dots N\}$ and $[N]_0$ to denote $\{0 \dots N\}$. We denote by $\mathbb{R}_+^n$ the set of nonnegative n dimensional vectors. We denote by $[0,1]^n$ the $n-$dimensional unit cube.

2 IBM Model 1

We begin by reviewing IBM Model 1 and introducing the necessary notation. To this end, 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). 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 $e_1^{(k)} \dots e_{l_k}^{(k)}$ where l_k is the length of the k'th English sentence, and each $e_i^{(k)} \in E$; similarly the k'th French sentence is a sequence $f_1^{(k)} \dots f_{m_k}^{(k)}$ where each $f_j^{(k)} \in F$. We define $e_0^{(k)}$ 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 1 optimization problem is given in Fig. 1 and, for example, (Koehn, 2008). The parameters in this problem are $t(f|e)$. The $t(f|e)$ parameters are *translation* parameters specifying the probability of English word e being translated as French word f. The objective function is then the log-likelihood of the training data (see Eq. 3):

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

where $\log p(f_j^{(k)}|e^{(k)})$ is

$$\log \sum_{i=0}^{l_k} \frac{t(f_j^{(k)}|e_i^{(k)})}{1+l_k} = C + \log \sum_{i=0}^{l_k} t(f_j^{(k)}|e_i^{(k)}) \,,$$

and C is a constant that can be ignored.

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

Parameters:

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

Constraints:

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

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

Objective: Maximize

$$\frac{1}{n}\sum_{k=1}^{n}\sum_{j=1}^{m_k}\log \sum_{i=0}^{l_k} t(f_j^{(k)}|e_i^{(k)}) \tag{3}$$

with respect to the $t(f|e)$ parameters.

Figure 1: The IBM Model 1 Optimization Problem.

While IBM Model 1 is concave optimization problem, it is not strictly concave (Toutanova and Galley, 2011). Therefore, optimization methods for IBM Model 1 (specifically, the EM algorithm) are typically only guaranteed to reach a global maximum of the objective function (see the Appendix for a simple example contrasting convex and strictly convex functions). In particular, although the objective cost is the same for any optimal solution, the translation quality of the solutions is not fixed and will still depend on the initialization of the model (Toutanova and Galley, 2011).

3 A Strictly Concave IBM Model 1

We now detail a very simple method to make IBM Model 1 strictly concave with a unique optimal solution without the need for appending an l_2 loss.

Theorem 1. *Consider IBM Model 1 and modify its objective to be*

$$\frac{1}{n}\sum_{k=1}^{n}\sum_{j=1}^{m_k}\log \sum_{i=0}^{l_k} h_{i,j,k}(t(f_j^{(k)}|e_i^{(k)})) \tag{4}$$

where $h_{i,j,k} : \mathbb{R}_+ \to \mathbb{R}_+$ is strictly concave. With the new objective and the same constraints as IBM Model 1, this new optimization problem is strictly concave.

Proof. To prove concavity, we now show that the new likelihood function

$$L(\mathbf{t}) = \frac{1}{n} \sum_{k=1}^{n} \sum_{j=1}^{m_k} \log \sum_{i=0}^{l_k} h_{i,j,k}(t(f_j^{(k)}|e_i^{(k)})) \,,$$

is strictly concave (concavity follows in the same way trivially). Suppose by way of contradiction that there is $(\mathrm{t}) \neq (\mathrm{t}')$ and $\theta \in (0,1)$ such that equality hold for Jensen's inequality. Since $h_{i,j,k}$ is strictly concave and $(\mathrm{t}) \neq (\mathrm{t}')$ we must have that there must be some (k,j,i) such that $t(f_j^{(k)}|e_i^{(k)}) \neq t'(f_j^{(k)}|e_i^{(k)})$ so that Jensen's inequality is strict for $h_{i,j,k}$ and we have

$$\begin{aligned} &h_{i,j,k}(\theta t(f_j^{(k)}|e_i^{(k)}) + (1-\theta)t'(f_j^{(k)}|e_i^{(k)})) \\ &\quad > \theta h_{i,j,k}(t(f_j^{(k)}|e_i^{(k)})) + (1-\theta)h_{i,j,k}(t'(f_j^{(k)}|e_i^{(k)})) \end{aligned}$$

Using Jensen's inequality, the monotonicity of the log, and the above strict inequality we have

$$\begin{aligned} &L(\theta\mathbf{t} + (1-\theta)\mathbf{t}') \\ &= \sum_{k=1}^{n} \sum_{j=1}^{m_k} \log \sum_{i=0}^{l_k} h_{i,j,k}(\theta t(f_j^{(k)}|e_i^{(k)}) + (1-\theta)t'(f_j^{(k)}|e_i^{(k)})) \\ &> \sum_{k=1}^{n} \sum_{j=1}^{m_k} \log \sum_{i=0}^{l_k} \theta h_{i,j,k}(t(f_j^{(k)}|e_i^{(k)})) + (1-\theta)h_{i,j,k}(t'(f_j^{(k)}|e_i^{(k)})) \\ &\geq \theta \sum_{k=1}^{n} \sum_{j=1}^{m_k} \log \sum_{i=0}^{l_k} h_{i,j,k}(t(f_j^{(k)}|e_i^{(k)})) \\ &+ (1-\theta) \sum_{k=1}^{n} \sum_{j=1}^{m_k} \log \sum_{i=0}^{l_k} h_{i,j,k}(t'(f_j^{(k)}|e_i^{(k)})) \\ &= \theta L(\mathbf{t}) + (1-\theta) L(\mathbf{t}') \end{aligned}$$

□

The IBM Model 1 strictly concave optimization problem is presented in Fig. 2. In (7) it is crucial that each $h_{i,j,k}$ be strictly concave within $\sum_{i=0}^{l_k} h_{i,j,k}(t(f_j^{(k)}|e_i^{(k)}))$. For example, we have that $\sqrt{x_1} + x_2$ is concave but not strictly concave and the proof of Theorem 1 would break down. To see this, we can consider $(x_1, x_2) \neq (x_1, x_3)$ and note that equality holds in Jensen's inequality. We should be clear: the main reason why Theorem 1 works is that we have $h_{i,j,k}$ are strictly concave (on $\mathbb{R}_+$) and all the lexical probabilities that are arguments to L are present within the log-likelihood.

Input: Define E, F, L, M, $(e^{(k)}, f^{(k)}, l_k, m_k)$ for $k = 1 \ldots n$, $D(e)$ for $e \in E$ as in Section 2. A set of strictly concave functions $h_{i,j,k} : \mathbb{R}_+ \to \mathbb{R}_+$.

Parameters:

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

Constraints:

$$\forall e \in E, f \in D(e), \quad t(f|e) \geq 0 \tag{5}$$

$$\forall e \in E, \quad \sum_{f \in D(e)} t(f|e) = 1 \tag{6}$$

Objective: Maximize

$$\frac{1}{n} \sum_{k=1}^{n} \sum_{j=1}^{m_k} \log \sum_{i=0}^{l_k} h_{i,j,k}(t(f_j^{(k)}|e_i^{(k)})) \tag{7}$$

with respect to the $t(f|e)$ parameters.

Figure 2: The IBM Model 1 strictly concave optimization problem.

4 Parameter Estimation via EM

For the IBM Model 1 strictly concave optimization problem, we can derive a clean EM Algorithm if we base our relaxation of

$$h_{i,j,k}(t(f_j^{(k)}|e_i^{(k)})) = \alpha(e_i^{(k)}, f_j^{(k)})(t(f_j^{(k)}|e_i^{(k)}))^{\beta(e_i^{(k)}, f_j^{(k)})}$$

with $\beta(e_i^{(k)}, f_j^{(k)}) < 1$. To justify this, we first need the following:

Lemma 1. *Consider $h : \mathbb{R}_+ \to \mathbb{R}_+$ given by $h(x) = x^\beta$ where $\beta \in (0,1)$. Then h is strictly concave.*

Proof. The proof of this lemma is elementary and follows since the second derivative given by $h''(x) = \beta(\beta-1)x^{\beta-2}$ is strictly negative. □

For our concrete experiments, we picked a model based on Lemma 1 and used $h(x) = \alpha x^\beta$ with $\alpha, \beta \in (0,1)$ so that

$$h_{i,j,k}(t(f_j^{(k)}|e_i^{(k)})) = \alpha(f_j^{(k)}, e_i^{(k)})(t(f_j^{(k)}|e_i^{(k)}))^{\beta(f_j^{(k)}, e_i^{(k)})} \,.$$

Using this setup, parameter estimation for the new model can be accomplished via a slight modification of the EM algorithm for IBM Model 1. In particular, we have that the posterior probabilities of this model factor just as those of the standard Model 1 and we have an M step that requires optimizing

$$\sum_{a^{(k)}} q(a^{(k)}|e^{(k)}, f^{(k)}) \log p(f^{(k)}, a^{(k)}|e^{(k)})$$

Input: Define E, F, L, M, $(e^{(k)}, f^{(k)}, l_k, m_k)$ for $k = 1 \ldots n$, $D(e)$ for $e \in E$ as in Section 2. An integer T specifying the number of passes over the data. A set of weighting parameter $\alpha(e, f), \beta(e, f) \in (0, 1)$ for each $e \in E$, $f \in D(e)$. A tuning parameter $\lambda > 0$.

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 $t(f|e) = 1/|D(e)|$.

EM Algorithm:

for all $t = 1 \ldots T$ **do**
  $\forall e \in E,\ f \in D(e),\ count(f, e) = 0$
  $\forall e \in E,\ count(e) = 0$
  EM Algorithm: Expectation
  for all $k = 1 \ldots n$ **do**
    for all $j = 1 \ldots m_k$ **do**
      $\delta_1[i] = 0\ \forall i \in [l_k]_0$
      $\Delta_1 = 0$
      for all $i = 0 \ldots l_k$ **do**
        $\delta_1[i] = \alpha(f_j^{(k)}, e_i^{(k)})(t(f_j^{(k)}|e_i^{(k)}))^{\beta(f_j^{(k)}, e_i^{(k)})}$
        Δ_1 += $\delta_1[i]$
      for all $i = 0 \ldots l_k$ **do**
        $\delta_1[i] = \frac{\delta_1[i]}{\Delta_1}$
        $count(f_j^{(k)}, e_i^{(k)})$ += $\beta(f_j^{(k)}, e_i^{(k)})\delta_1[i]$
        $count(e_i^{(k)})$ += $\beta(f_j^{(k)}, e_i^{(k)})\delta_1[i]$
  EM Algorithm: Maximization
  for all $e \in E$ **do**
    for all $f \in D(e)$ **do**
      $t(f|e) = \frac{count(e,f)}{count(e)}$

Output: t parameters

Figure 3: Pseudocode for T iterations of the EM Algorithm for the IBM Model 1 strictly concave optimization problem.

where

$$q(a^{(k)}|e^{(k)}, f^{(k)}) \propto \prod_{j=1}^{m_k} h_{a_j^{(k)},j,k}(t(f_j^{(k)}|e^{(k)}_{a_j^{(k)}}))$$

are constants gotten in the E step. This optimization step is very similar to the regular Model 1 M step since the β drops down using $\log t^\beta = \beta \log t$; the exact same count-based method can be applied. The details of this algorithm are in Fig. 3.

5 Choosing α and β

The performance of our new model will rely heavily on the choice of $\alpha(e_i^{(k)}, f_j^{(k)})$, $\beta(e_i^{(k)}, f_j^{(k)}) \in (0, 1)$ we use. In particular, we could make β depend on the association between the words, or the words' positions, or both. One classical measure of word association is the dice coefficient (Och and Ney, 2003) given by

$$dice(e, f) = \frac{2c(e, f)}{c(e) + c(f)}\ .$$

In the above, the count terms c are the number of training sentences that have either a particular word or a pair of of words (e, f). As with the other choices we explore, the dice coefficient is a fraction between 0 and 1, with 0 and 1 implying less and more association, respectively. Additionally, we make use of positional constants like those of the IBM Model 2 distortions given by

$$d(i|j, l, m) = \begin{cases} \frac{1}{(l+1)Z(j,l,m)} & : i = 0 \\ \frac{le^{-\lambda|\frac{i}{l} - \frac{j}{m}|}}{(l+1)Z(j,l,m)} & : i \neq 0 \end{cases}$$

In the above, $Z(j, l, m)$ is the partition function discussed in (Dyer et al., 2013). The previous measures all lead to potential candidates for $\beta(e, f)$, we have $t(f|e) \in (0, 1)$, and we want to enlarge competing values when decoding (we use αt^β instead of t when getting the Viterbi alignment). The above then implies that we will have the word association measures inversely proportional to β, and so we set $\beta(e, f) = 1 - dice(e, f)$ or $\beta(e, f) = 1 - d(i|j, l, m)$. In our experiments we picked $\alpha(f_j^{(k)}, e_i^{(k)}) = d(i|j, l_k, m_k)$ or 1; we hold λ to a constant of either 16 or 0 and do not estimate this variable ($\lambda = 16$ can be chosen by cross validation on a small trial data set).

6 Experiments

6.1 Data Sets

For our alignment experiments, we used a subset of the Canadian Hansards bilingual corpus with 247,878 English-French sentence pairs as training data, 37 sentences of development data, and 447 sentences of test data (Michalcea and Pederson, 2003). As a second validation corpus, we considered a training set of 48,706 Romanian-English sentence-pairs, a development set of 17 sentence pairs, and a test set of 248 sentence pairs (Michalcea and Pederson, 2003).

6.2 Methodology

Below we report results in both AER (lower is better) and F-Measure (higher is better) (Och and Ney, 2003) for the English $\rightarrow$ French translation direction. To declare a better model we have to settle on an alignment measure. Although the relationship between AER/F-Measure and translation quality varies (Dyer et al., 2013), there are some positive experiments (Fraser and Marcu, 2004) showing that F-Measure may be more useful, so perhaps a comparison based on F-Measure is ideal.

Table 1 contains our results for the Hansards data. For the smaller Romanian data, we obtained similar behavior, but we leave out these results due

(α, β)	$(1,1)$	$(d,1)$	$(1,1-dice)$	$(1,1-d)$	$(d,1-d)$
Iteration			AER		
0	0.8716	0.6750	0.6240	0.6597	**0.5570**
1	0.4426	0.2917	0.4533	**0.2738**	0.3695
2	0.3383	0.2323	0.4028	**0.2318**	0.3085
3	0.3241	**0.2190**	0.3845	0.2252	0.2881
4	0.3191	**0.2141**	0.3751	0.2228	0.2833
5	0.3175	**0.2118**	0.3590	0.2229	0.2812
6	0.3160	**0.2093**	0.3566	0.2231	0.2793
7	0.3203	**0.2090**	0.3555	0.2236	0.2783
8	0.3198	**0.2075**	0.3546	0.2276	0.2777
9	0.3198	**0.2066**	0.3535	0.2323	0.2769
10	0.3177	**0.2065**	0.3531	0.2352	0.2769
Iteration			F-Measure		
0	0.0427	0.1451	**0.2916**	0.1897	0.2561
1	0.4213	0.5129	0.4401	**0.5453**	0.4427
2	0.5263	0.5726	0.4851	**0.5940**	0.5014
3	0.5413	0.5852	0.5022	**0.6047**	0.5199
4	0.5480	0.5909	0.5111	**0.6085**	0.5255
5	0.5500	0.5939	0.5264	**0.6101**	0.5273
6	0.5505	0.5959	0.5282	**0.6101**	0.5286
7	0.5449	0.5965	0.5298	**0.6096**	0.5296
8	0.5456	0.5977	0.5307	**0.6068**	0.5300
9	0.5451	0.5985	0.5318	**0.6040**	0.5309
10	0.5468	0.5984	0.5322	**0.6024**	0.5311

Table 1: Results on the English-French data for various (α, β) settings as discussed in Section 5. For the d parameters, we use $\lambda = 16$ throughout. The standard IBM Model 1 is column 1 and corresponds to a setting of (1,1). The not necessarily strictly concave model with (d,1) setting gives the best AER, while the strictly concave model given by the $(1, 1-d)$ setting has the highest F-Measure.

to space limitations. Our experiments show that using

$$h_{i,j,k}(t(f_j^{(k)}|e_i^{(k)})) = (t(f_j^{(k)}|e_i^{(k)}))^{1-d(i|j,l_k,m_k)}$$

yields the best F-Measure performance and is not far off in AER from the "fake"[2] IBM Model 2 (gotten by setting $(\alpha, \beta) = (d, 1)$) whose results are in column 2 (the reason why we use this model at all is since it should be better than IBM 1: we want to know how far off we are from this obvious improvement). Moreover, we note that dice does not lead to quality β exponents and that, unfortunately, combining methods as in column 5 ($(\alpha, \beta) = (d, 1-d)$) does not necessarily lead to additive gains in AER and F-Measure performance.

[2]Generally speaking, when using

$$h_{i,j,k}(t(f_j^{(k)}|e_i^{(k)})) = d(i|j,l_k,m_k)t(f_j^{(k)}|e_i^{(k)})$$

with d constant we cannot use Theorem 3 since h is linear. Most likely, the strict concavity of the model will hold because of the asymmetry introduced by the d term; however, there will be a necessary dependency on the data set.

7 Comparison with Previous Work

In this section we take a moment to also compare our work with the classical IBM 1 work of (Moore, 2004). Summarizing (Moore, 2004), we note that this work improves substancially upon the classical IBM Model 1 by introducing a set of heuristics, among which are to (1) modify the lexical parameter dictionaries (2) introduce an initialization heuristic (3) modify the standard IBM 1 EM algorithm by introducing smoothing (4) tune additional parameters. However, we stress that the main concern of this work is not just heuristic-based empirical improvement, but also structured learning. In particular, although using an regularizer l_2 and the methods of (Moore, 2004) would yield a strictly concave version of IBM 1 as well (with improvements), it is not at all obvious how to choose the learning rate or set the penalty on the lexical parameters. The goal of our work was to offer a new, alternate form of regularization. Moreover, since we are changing the original log-likelihood, our method can be thought of as way of *bringing the l_2 regularizer inside the log likelihood*. Like (Moore, 2004), we also achieve appreciable gains but have just one tuning parameter (when $\beta = 1 - d$ we just have the centering λ parameter) and do not break the probabilistic interpretation any more than appending a regularizer would (our method modifies the log-likelihood but the simplex constrains remain).

8 Conclusion

In this paper we showed how IBM Model 1 can be made into a strictly convex optimization problem via functional composition. We looked at a specific member within the studied optimization family that allows for an easy EM algorithm. Finally, we conducted experiments showing how the model performs on some standard data sets and empirically showed 30% important over the standard IBM Model 1 algorithm. For further research, we note that picking the optimal $h_{i,j,k}$ is an open question, so provably finding and justifying the choice is one topic of interest.

Acknowledgments

Andrei Simion was supported by a Google research award. Cliff Stein is supported in part by NSF grants CCF-1349602 and CCF-1421161.

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.

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

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.

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. 2015. A Family of Latent Variable Convex Relaxations for IBM Model 2. *In Proceedings of the AAAI.*

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.*