Why Initialization Matters for IBM Model 1: Multiple Optima and Non-Strict Convexity

Kristina Toutanova
Microsoft Research
Redmond, WA 98005, USA
kristout@microsoft.com

Michel Galley
Microsoft Research
Redmond, WA 98005, USA
mgalley@microsoft.com

Abstract

Contrary to popular belief, we show that the optimal parameters for IBM Model 1 are not unique. We demonstrate that, for a large class of words, IBM Model 1 is indifferent among a continuum of ways to allocate probability mass to their translations. We study the magnitude of the variance in optimal model parameters using a linear programming approach as well as multiple random trials, and demonstrate that it results in variance in test set log-likelihood and alignment error rate.

1 Introduction

Statistical alignment models have become widely used in machine translation, question answering, textual entailment, and non-NLP application areas such as information retrieval (Berger and Lafferty, 1999) and object recognition (Duygulu et al., 2002).

The complexity of the probabilistic models needed to explain the hidden correspondence among words has necessitated the development of highly non-convex and difficult to optimize models, such as HMMs (Vogel et al., 1996) and IBM Models 3 and higher (Brown et al., 1993). To reduce the impact of getting stuck in bad local optima the original IBM paper (Brown et al., 1993) proposed the idea of training a sequence of models from simpler to complex, and using the simpler models to initialize the more complex ones. IBM Model 1 was the first model in this sequence and was considered a reliable initializer due to its convexity.

In this paper we show that although IBM Model 1 is convex, it is not strictly convex, and there is a large space of parameter values that achieve the same optimal value of the objective.

We study the magnitude of this problem by formulating the space of optimal parameters as solutions to a set of linear equalities and seek maximally different parameter values that reach the same objective, using a linear programming approach. This lets us quantify the percentage of model parameters that are not uniquely defined, as well as the number of word types that have uncertain translation probabilities. We additionally study the achieved variance in parameters resulting from different random initialization in EM, and the impact of initialization on test set log-likelihood and alignment error rate. These experiments suggest that initialization does matter in practice, contrary to what is suggested in (Brown et al., 1993, p. 273).[1]

2 Preliminaries

In Appendix A we define convexity and strict convexity of functions following (Boyd and Vandenberghe, 2004). In this section we detail the generative model for Model 1.

2.1 IBM Model 1

IBM Model 1 (Brown et al., 1993) defines a generative process for a source sentences $\mathbf{f} = f_1 \dots f_m$ and alignments $\mathbf{a} = a_1 \dots a_m$ given a corresponding target translation $\mathbf{e} = e_0 \dots e_l$. The generative process is as follows: (*i*) pick a length m using a uniform distribution with mass function proportional to ϵ; (*ii*) for each source word position j, pick an alignment

[1] When referring to Model 1, Brown et al. (1993) state that "details of our initial guesses for $t(f|e)$ are unimportant".

position in the target sentence $a_j \in 0, 1, \ldots, l$ from a uniform distribution; and (*iii*) generate a source word using the translation probability distribution $t(f_j|e_{a_j})$. A special empty word (NULL) is assumed to be part of the target vocabulary and to occupy the first position in each target language sentence (e_0=NULL).

The trainable parameters of Model 1 are the lexical translation probabilities $t(f|e)$, where f and e range over the source and target vocabularies, respectively. The log-probability of a single source sentence $\mathbf{f}$ given its corresponding target sentence $\mathbf{e}$ and values for the translation parameters $t(f|e)$ can be written as follows (Brown et al., 1993):

$$\sum_{j=1}^{m} \log \sum_{i=0}^{l} t(f_j|e_i) - m \log(l+1) + \log \epsilon$$

The parameters of IBM Model 1 are usually derived via maximum likelihood estimation from a corpus, which is equivalent to negative log-likelihood minimization. The negative log-likelihood for a parallel corpus D is:

$$L_D(T) = -\sum_{\mathbf{f},\mathbf{e}} \sum_{j=1}^{m} \log \sum_{i=0}^{l} t(f_j|e_i) + B \qquad (1)$$

where T is the matrix of translation probabilities and B represents the other terms of Model 1 (string length probability and alignment probability), which are constant with respect to the translation parameters $t(f|e)$.

We can define the optimization problem as the one of minimizing negative log-likelihood $L_D(T)$ subject to constraints ensuring that the parameters are well-formed probabilities, i.e., that they are non-negative and summing to one. It is well-known that the EM algorithm for this problem converges to a local optimum of the objective function (Dempster et al., 1977).

3 Convexity analysis for IBM Model 1

In this section we show that, contrary to the claim in (Brown et al., 1993), the optimization problem for IBM Model 1 is not strictly convex, which means that there could be multiple parameter settings that achieve the same globally optimal value of the objective.[2]

The function $-\log(x)$ is strictly convex (Boyd and Vandenberghe, 2004). Each term in the negative log-likelihood is a negative logarithm of a sum of parameters. The negative logarithm of a sum is not strictly convex, as illustrated by the following simple counterexample. Let's look at the function $-\log(x_1 + x_2)$. We can express it in vector notation using $-\log(\mathbf{1}^T\mathbf{x})$, where $\mathbf{1}$ is a vector with all elements equal to 1. We will come up with two parameter settings $\mathbf{x}$,$\mathbf{y}$ and a value θ that violate the definition of strict convexity. Take $\mathbf{x} = [x_1, x_2] = [.1, .2]$, $\mathbf{y} = [y_1, y_2] = [.2, .1]$ and $\theta = .5$. We have $\mathbf{z} = \theta\mathbf{x} + (1-\theta)\mathbf{y} = [z_1, z_2] = [.15, .15]$. Also $-\log(\mathbf{1}^T(\theta\mathbf{x} + (1-\theta)\mathbf{y})) = -\log(z_1 + z_2) = -\log(.3)$. On the other hand, $-\theta\log(x_1 + x_2) - (1-\theta)\log(y_1+y_2) = -\log(.3)$. Strict convexity requires that the former expression be strictly smaller than the latter, but we have equality. Therefore, this function is not strictly convex. It is however convex as stated in (Brown et al., 1993), because it is a composition of $\log$ and a linear function.

We thus showed that every term in the negative log-likelihood objective is convex but not strictly convex and thus the overall objective is convex, but not strictly convex. Because the objective is convex, the inequality constraints are convex, and the equality constraints are affine, the IBM Model 1 optimization problem is a convex optimization problem. Therefore every local optimum is a global optimum. But since the objective is not strictly convex, there might be multiple distinct parameter values achieving the same optimal value. In the next section we study the actual space of optima for small and realistically-sized parallel corpora.

[2] Brown et al. (1993, p. 303) claim the following about the log-likelihood function (Eq. 51 and 74 in their paper, and Eq. 1 in ours): "The objective function (51) for this model is a strictly concave function of the parameters", which is equivalent to claiming that the negative log-likelihood function is strictly convex. In this section, we will theoretically demonstrate that Brown et al.'s claim is in fact incorrect. Furthermore, we will empirically show in Sections 4 and 5 that multiple distinct parameter values can achieve the global optimum of the objective function, which also disproves Brown et al.'s claim about the strict convexity of the objective function. Indeed, if a function is strictly convex, it admits a *unique* globally optimum solution (Boyd and Vandenberghe, 2004, p. 151), so our experiments prove by *modus tollens* that Brown et al.'s claim is wrong.

4 Solution Space

In this section, we characterize the set of parameters that achieve the maximum of the log-likelihood of IBM Model 1. As illustrated with the following simple example, it is relatively easy to establish cases where the set of optimal parameters $t(f|e)$ is not unique:

$\mathbf{e}$: short sentence $\mathbf{f}$: phrase courte

If the above sentence pair represents the entire training data, Model 1 likelihood (ignoring NULL words) is proportional to

$$\big[t(\text{phrase}|\text{short}) + t(\text{phrase}|\text{sentence})\big] \cdot \big[t(\text{courte}|\text{short}) + t(\text{courte}|\text{sentence})\big]$$

which can be maximized in infinitely many different ways. For instance, setting $t(\text{phrase}|\text{sentence}) = t(\text{courte}|\text{short}) = 1$ yields the maximum likelihood value with $(0+1)(1+0) = 1$, but the most divergent set of parameters ($t(\text{courte}|\text{sentence}) = t(\text{phrase}|\text{sentence}) = 1$) also reaches the same optimum: $(1+0)(0+1) = 1$. While this example may not seem representative given the small size of this data, the laxity of Model 1 that we observe in this example also surfaces in real and much larger training sets. Indeed, it suffices that a given pair of target words (e_1,e_2) systematically co-occurs in the data (as with $e_1 = \text{short}$ $e_2 = \text{sentence}$) to cause Model 1 to fail to distinguish the two.[3]

To characterize the solution space, we use the definition of IBM Model 1 log-likelihood from Eq. 1 in Section 2.1. We ask whether distinct sets of parameters yield the same minimum negative log-likelihood value of Eq. 1, i.e., whether we can find distinct models $t(f|e)$ and $t'(f|e)$ so that:

$$\sum_{\mathbf{f},\mathbf{e}} \sum_{j=1}^{m} \log \sum_{i=0}^{l} t(f_j|e_i) = \sum_{\mathbf{f},\mathbf{e}} \sum_{j=1}^{m} \log \sum_{i=0}^{l} t'(f_j|e_i)$$

Since the negative logarithm is strictly convex, the above equation can be satisfied for optimal parameters only if the following holds for each $\mathbf{f}$, $\mathbf{e}$ pair:

$$\sum_{i=0}^{l} t(f_j|e_i) = \sum_{i=0}^{l} t'(f_j|e_i), j = 1 \ldots m \quad (2)$$

We can further simplify the above equation if we recall that both $t(f|e)$ and $t'(f|e)$ are maximum log-likelihood parameters, and noting it is generally easy to obtain one such set of parameters, e.g., by running the EM algorithm until convergence. Using these EM parameters (θ) in the right hand side of the equation, we replace these right hand sides with EM's estimate $t_\theta(f_j|\mathbf{e})$. This finally gives us the following linear program (LP), which characterizes the solution space of the maximum log-likelihood:[4]

$$\sum_{i=0}^{l} t(f_j|e_i) = t_\theta(f_j|\mathbf{e}),\ j = 1 \ldots m \quad \forall \mathbf{f}, \mathbf{e} \quad (3)$$

$$\sum_{f} t(f|e) = 1,\ \forall e \quad (4)$$

$$t(f|e) \geq 0,\ \forall e, f \quad (5)$$

The two conditions in Eq. 4-5 are added to ensure that $t(f|e)$ is well-formed. To solve this LP, we use the interior-point method of (Karmarkar, 1984).

To measure the maximum divergence in optimal model parameters, we solve the LP of Eq. 3-5 by minimizing the linear objective function $\mathbf{x}_{k-1}^T \mathbf{x}_k$, where $\mathbf{x}_k$ is the column-vector representing all parameters of the model $t(f|e)$ currently optimized, and where $\mathbf{x}_{k-1}$ is a pre-existing set of maximum log-likelihood parameters. Starting with $\mathbf{x}_0$ defined using EM parameters, we are effectively searching for the vector $\mathbf{x}_1$ with lowest cosine similarity to $\mathbf{x}_0$. We repeat with $k > 1$ until $\mathbf{x}_k$ doesn't reduce the cosine similarity with any of the previous parameter vectors $\mathbf{x}_0 \ldots \mathbf{x}_{k-1}$ (which generally happens with $k = 3$).[5]

[3] Since e_1 and e_2 co-occur with exactly the same source words, one can redistribute the probability mass between $t(f|e_1)$ and $t(f|e_2)$ without affecting the log-likelihood. This is true if (a) the two distributions remain well-formed: $\sum_j t(f_j|e_i) = 1$ for $i \in \{1, 2\}$; (b) any adjustments to parameters of f_j leave each estimate $t(f_j|e_1) + t(f_j|e_2)$ unchanged.

[4] In general, an LP admits either (a) an infinity of solutions, when the system is underconstrained; (b) exactly one solution; (c) zero solutions, when it is ill-posed. The latter case never occurs in our case, since the system was explicitly constructed to allow at least one solution: the parameter set returned by EM.

[5] Note that this greedy procedure is not guaranteed to find the two points of the feasible region (a convex polytope) with minimum cosine similarity. This problem is related to finding the diameter of this polytope, which is known to be NP-hard when the number of variables is unrestricted (Kaibel et al., 2002). Nevertheless, divergences found by this procedure are fairly substantial, as shown in Section 5.

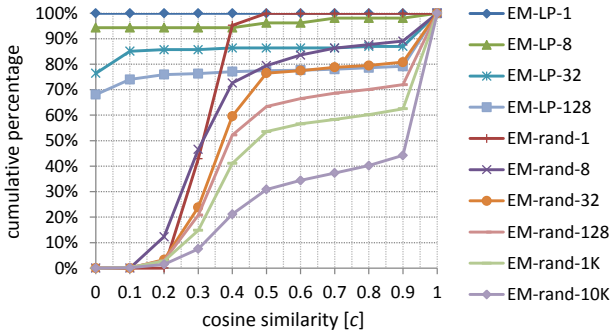

Figure 1: Percentage of target words for which we found pairs of distributions $t(f|e)$ and $t'(f|e)$ whose cosine similarity drops below a given threshold c (x-axis).

5 Experiments

In this section, we show that the solution space defined by the LP of Eq. 3-5 can be fairly large. We demonstrate this with Bulgarian-English parallel data drawn from the JRC-AQUIS corpus (Steinberger et al., 2006). Our training data consists of up to 10,000 sentence pairs, which is representative of the amount of data used to train SMT systems for language pairs that are relatively resource-poor.

Figure 1 relies on two methods for determining to what extent the model $t(f|e)$ can vary while remaining optimal. The EM-LP-N method consists of applying the method described at the end of Section 4 with N training sentence pairs. For EM-rand-N, we instead run EM 100 times (also on N sentence pairs) until convergence using different random starting points, and then use cosine similarity to compare the resulting models.[6] Figure 1 shows some surprising results: First, EM-LP-128 finds that, for about 68% of target token types, cosine similarity between contrastive models is equal to 0. A cosine of zero essentially means that we can turn 1's into 0's without affecting log-likelihood, as in the *short sentence* example in Section 4. Second, with a much larger training set, EM-rand-10K finds a cosine similarity lower or equal to 0.5 for 30% of word types, which is a large portion of the vocabulary.

[6]While the first method is better at finding divergent optimal model parameters, it needs to construct large linear programs that do not scale with large training sets (linear systems quickly reach millions of entries, even with 128 sentence pairs). We use EM-rand to assess the model space on larger training set, while we use EM-LP mainly to illustrate that divergence between optimal models can be much larger than suggested by EM-rand.

train	coupled	non-unique			log-lik	
		all	c.	non-c.	stdev	unif
1	100	100	100	-	2.9K	-4.9K
8	83.6	89.0	100	33.3	2.3K	-2.3K
32	77.8	81.8	100	17.9	874	74.4
128	67.8	73.3	99.7	17.7	270	272
1K	52.6	64.1	99.8	24.0	220	281
10K	30.3	47.33	99.9	24.4	150	300

Table 1: Results using 100 random initialization trials.

In Table 1 we show additional statistics computed from the EM-rand-N experiments. Every row represents statistics for a given training set size (in number of sent. pairs, first column); the second column shows the percent of target word types that always co-occur with another word type (we term these words *coupled*); the third, fourth, and fifth columns show the percent of word types whose translation distributions were found to be non-unique, where we define the non-unique types to be ones where the minimum cosine between any two different optimal parameter vectors was less than .95. The percent of non-unique types are reported overall, as well as only among coupled words (c.) and non-coupled words (non-c.). The last two columns show the standard deviation in test set log-likelihood across different random trials, as well as the difference between the log-likelihood of the uniformly initialized model and the best model from the random trials.

We can see that as the training set size increases, the percentage of words that have non-unique translation probabilities goes down but is still very large. The coupled words almost always end up having varying translation parameters at convergence (more than 99.5% of these words). This also happens for a sizable portion of the non-coupled words, which suggests that there are additional patterns of co-occurrence that result in non-determinism.[7] We also computed the percent of word types that are coupled for two more-realistically sized data-sets: we found that in a 1.6 million sent pair English-Bulgarian corpus 15% of Bulgarian word types were coupled and in a 1.9 million English-German corpus from the WMT workshop (Callison-Burch et al., 2010), 13% of the German word types were coupled.

The log-likelihood statistics show that although

[7]We did not perform such experiments for larger data-sets, since EM takes thousands of iterations to converge.

the standard deviation goes down with training set size, it is still large at reasonable data sizes. Interestingly, the uniformly initialized model performs worse for a very small data size, but it catches up and surpasses the random models at data sizes greater than 100 sentence pairs.

To further evaluate the impact of initialization for IBM Model 1, we report on a set of experiments looking at alignment error rate achieved by different models. We report the performance of Model 1, as well as the performance of the more competitive HMM alignment model (Vogel et al., 1996), initialized from IBM-1 parameters. The dataset for these experiments is English-French parallel data from Hansards. The manually aligned data for evaluation consists of 137 sentences (a development set from (Och and Ney, 2000)).

We look at two different training set sizes, a small set consisting of 1000 sentence pairs, and a reasonably-sized dataset containing 100,000 sentence pairs. In each data size condition, we report on the performance achieved by IBM-1, and the performance achieved by HMM initialized from the IBM-1 parameters. For IBM Model 1 training, we either perform only 5 EM iterations (the standard setting in GIZA++), or run it to convergence. For each of these two settings, we either start training from uniform $t(f|e)$ parameters, or random parameters. Table 2 details the results of these experiments.

Each row in the table represents an experimental condition, indicating the training data size (1K in the first four rows and 100K in the next four rows), the type of initialization (uniform versus random) and the number of iterations EM was run for Model 1 (5 iterations versus unlimited (to convergence, denoted ∞)). The numbers in the table are alignment error rates, achieved at the end of Model 1 training, and at 5 iterations of HMM. When random initialization is used, we run 20 random trials with different initialization, and report the min, max, and mean AER achieved in each setting.

From the table, we can draw several conclusions. First, in agreement with current practice using only 5 iterations of Model 1 training results in better final performance of the HMM model (even though the performance of Model 1 is higher when ran to convergence). Second, the minimum AER achieved by randomly initialized models was always smaller than the AER of the uniform-initialized models. In some cases, even the mean of the random trials was better than the corresponding uniform model. Interestingly, the advantage of the randomly initialized models in AER does not seem to diminish with increased training data size like their advantage in test set perplexity.

setting	IBM-1			HMM		
	min	mean	max	min	mean	max
1K-unif-5	42.99	-	-	22.53	-	-
1K-rand-5	42.90	44.07	45.08	22.26	22.99	24.01
1K-unif-∞	42.10	-	-	28.09	-	-
1K-rand-∞	41.72	42.61	43.63	27.88	28.47	28.89
100K-unif-5	28.98	-	-	12.68	-	-
100K-rand-5	28.63	28.99	30.13	12.25	12.62	12.89
100K-unif-∞	28.18	-	-	16.84	-	-
100K-rand-∞	27.95	28.22	30.13	16.66	16.78	16.85

Table 2: AER results for Model 1 and HMM using uniform and random initialization. We do not report mean and max for uniform, since they are identical to min.

6 Conclusions

Through theoretical analysis and three sets of experiments, we showed that IBM Model 1 is not strictly convex and that there is large variance in the set of optimal parameter values. This variance impacts a significant fraction of word types and results in variance in predictive performance of trained models, as measured by test set log-likelihood and word-alignment error rate. The magnitude of this non-uniqueness further supports the development of models that can use information beyond simple co-occurrence, such as positional and fertility information like higher order alignment models, as well as models that look beyond the surface form of a word and reason about morphological or other properties (Berg-Kirkpatrick et al., 2010).

In future work we would like to study the impact of non-determinism on higher order models in the standard alignment model sequence and to gain more insight into the impact of finer-grained features in alignment.

Acknowledgements

We thank Chris Quirk and Galen Andrew for valuable discussions and suggestions.

References

Taylor Berg-Kirkpatrick, Alexandre Bouchard-Côté, John DeNero, and Dan Klein. 2010. Painless unsupervised learning with features. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics.

Adam Berger and John Lafferty. 1999. Information retrieval as statistical translation. In *Proceedings of the 1999 ACM SIGIR Conference on Research and Development in Information Retrieval*.

Stephen 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.

Chris Callison-Burch, Philipp Koehn, Christof Monz, Kay Peterson, and Omar Zaidan, editors. 2010. *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*.

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).

Pinar Duygulu, Kobus Barnard, Nando de Freitas, P. Duygulu, K. Barnard, and David Forsyth. 2002. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Proceedings of ECCV*.

Volker Kaibel, Marc E. Pfetsch, and TU Berlin. 2002. Some algorithmic problems in polytope theory. In *Dagstuhl Seminars*, pages 23–47.

N. Karmarkar. 1984. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–395, December.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*.

Ralf Steinberger, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaz Erjavec, and Dan Tufis. 2006. The JRC-acquis: A multilingual aligned parallel corpus with 20+ languages. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)*.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *Proceedings of the 16th Int. Conf. on Computational Linguistics (COLING)*. Association for Computational Linguistics.

Appendix A: Convex functions and convex optimization problems

We denote the domain of a function f by $\mathbf{dom}\ f$.

Definition A function $f : \mathbf{R}^n \rightarrow \mathbf{R}$ is convex if and only if $\mathbf{dom}\ f$ is a convex set and for all $x, y \in \mathbf{dom}\ f$ and $\theta \geq 0, \theta \leq 1$:

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y) \qquad (6)$$

Definition A function f is strictly convex iff $\mathbf{dom}\ f$ is a convex set and for all $x \neq y \in \mathbf{dom}\ f$ and $\theta > 0, \theta < 1$:

$$f(\theta x + (1-\theta)y) < \theta f(x) + (1-\theta)f(y) \qquad (7)$$

Definition A convex optimization problem is defined by:

$$\min f_0(x)$$

subject to

$$f_i(x) \leq 0, i = 1 \ldots k$$
$$a_j^T x = b_j, j = 1 \ldots l$$

Where the functions f_0 to f_k are convex and the equality constraints are affine.

It can be shown that the feasible set (the set of points that satisfy the constraints) is convex and that any local optimum for the problem is a global optimum. If f_0 is strictly convex then any local optimum is the unique global optimum.