Lateen EM: Unsupervised Training with Multiple Objectives, Applied to Dependency Grammar Induction

Valentin I. Spitkovsky
Computer Science Department
Stanford University and Google Inc.
valentin@cs.stanford.edu

Hiyan Alshawi
Google Inc.
Mountain View, CA, 94043, USA
hiyan@google.com

Daniel Jurafsky
Departments of Linguistics and Computer Science
Stanford University, Stanford, CA, 94305, USA
jurafsky@stanford.edu

Abstract

We present new training methods that aim to mitigate local optima and slow convergence in unsupervised training by using additional imperfect objectives. In its simplest form, *lateen EM* alternates between the two objectives of ordinary "soft" and "hard" expectation maximization (EM) algorithms. Switching objectives when stuck can help escape local optima. We find that applying a single such alternation already yields state-of-the-art results for English dependency grammar induction. More elaborate lateen strategies track *both* objectives, with each validating the moves proposed by the other. Disagreements can signal earlier opportunities to switch or terminate, saving iterations. De-emphasizing fixed points in these ways eliminates some guesswork from tuning EM. An evaluation against a suite of unsupervised dependency parsing tasks, for a variety of languages, showed that lateen strategies significantly speed up training of both EM algorithms, and improve accuracy for hard EM.

1 Introduction

Expectation maximization (EM) algorithms (Dempster et al., 1977) play important roles in learning latent linguistic structure. Unsupervised techniques from this family excel at core natural language processing (NLP) tasks, including segmentation, alignment, tagging and parsing. Typical implementations specify a probabilistic framework, pick an initial model instance, and iteratively improve parameters using EM. A key guarantee is that subsequent model instances are no worse than the previous, according to training data likelihood in the given framework.

Another attractive feature that helped make EM instrumental (Meng, 2007) is its initial efficiency: Training tends to begin with large steps in a parameter space, sometimes bypassing many local optima at once. After a modest number of such iterations, however, EM lands close to an attractor. Next, its convergence rate necessarily suffers: Disproportionately many (and ever-smaller) steps are needed to finally approach this fixed point, which is almost invariably a local optimum. Deciding when to terminate EM often involves guesswork; and finding ways out of local optima requires trial and error. We propose several strategies that address both limitations.

Unsupervised objectives are, at best, loosely correlated with extrinsic performance (Pereira and Schabes, 1992; Merialdo, 1994; Liang and Klein, 2008, *inter alia*). This fact justifies (occasionally) deviating from a prescribed training course. For example, since *multiple* equi-plausible objectives are usually available, a learner could cycle through them, optimizing alternatives when the primary objective function gets stuck; or, instead of trying to escape, it could aim to avoid local optima in the first place, by halting search early if an improvement to one objective would come at the expense of harming another.

We test these general ideas by focusing on non-convex likelihood optimization using EM. This setting is standard and has natural and well-understood objectives: the classic, "soft" EM; and Viterbi, or "hard" EM (Kearns et al., 1997). The name "lateen" comes from the sea — triangular *lateen* sails can take wind on either side, enabling sailing vessels to *tack* (see Figure 1). As a captain can't count on favorable winds, so an unsupervised learner can't rely on co-operative gradients: soft EM maximizes

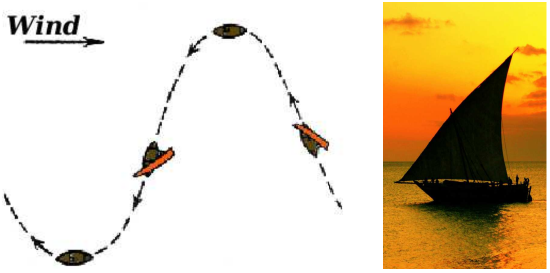

Figure 1: A triangular sail atop a traditional Arab sailing vessel, the *dhow* (right). Older square sails permitted sailing only before the wind. But the efficient *lateen* sail worked like a wing (with high pressure on one side and low pressure on the other), allowing a ship to go almost directly into a headwind. By *tacking*, in a zig-zag pattern, it became possible to sail in any direction, provided there was some wind at all (left). For centuries seafarers expertly combined both sails to traverse extensive distances, greatly increasing the reach of medieval navigation.[1]

likelihoods of observed data across assignments to hidden variables, whereas hard EM focuses on most likely completions.[2] These objectives are plausible, yet both can be provably "wrong" (Spitkovsky et al., 2010a, §7.3). Thus, it is permissible for lateen EM to maneuver between their gradients, for example by tacking around local attractors, in a zig-zag fashion.

2 The Lateen Family of Algorithms

We propose several strategies that use a secondary objective to improve over standard EM training. For hard EM, the secondary objective is that of soft EM; and vice versa if soft EM is the primary algorithm.

2.1 Algorithm #1: Simple Lateen EM

Simple lateen EM begins by running standard EM to convergence, using a user-supplied initial model, primary objective and definition of convergence. Next, the algorithm alternates. A single lateen alternation involves two phases: (i) retraining using the secondary objective, starting from the previous converged solution (once again iterating until convergence, but now of the secondary objective); and (ii) retraining using the primary objective again, starting from the latest converged solution (once more to convergence of the primary objective). The algorithm stops upon failing to sufficiently improve the primary objective across alternations (applying the standard convergence criterion end-to-end) and returns the best of all models re-estimated during training (as judged by the primary objective).

2.2 Algorithm #2: Shallow Lateen EM

Same as algorithm #1, but switches back to optimizing the primary objective after a *single* step with the secondary, during phase (i) of all lateen alternations. Thus, the algorithm alternates between optimizing a primary objective to convergence, then stepping away, using one iteration of the secondary optimizer.

2.3 Algorithm #3: Early-Stopping Lateen EM

This variant runs standard EM but quits early if the secondary objective suffers. We redefine convergence by "or"-ing the user-supplied termination criterion (i.e., a "small-enough" change in the primary objective) with *any* adverse change of the secondary (i.e., an increase in its cross-entropy). Early-stopping lateen EM does *not* alternate objectives.

2.4 Algorithm #4: Early-Switching Lateen EM

Same as algorithm #1, but with the new definition of convergence, as in algorithm #3. Early-switching lateen EM halts primary optimizers as soon as they hurt the secondary objective and stops secondary optimizers once they harm the primary objective. This algorithm terminates when it fails to sufficiently improve the primary objective across a full alternation.

2.5 Algorithm #5: Partly-Switching Lateen EM

Same as algorithm #4, but again iterating primary objectives to convergence, as in algorithm #1; secondary optimizers still continue to terminate early.

3 The Task and Study #1

We chose to test the impact of these five lateen algorithms on unsupervised dependency parsing — a task in which EM plays an important role (Paskin, 2001; Klein and Manning, 2004; Gillenwater et al., 2010, *inter alia*). This entailed two sets of experiments: In study #1, we tested whether single alternations of simple lateen EM (as defined in §2.1,

[1] Partially adapted from `http://www.britannica.com/EBchecked/topic/331395`, `http://allitera.tive.org/archives/004922.html` and `http://landscapedvd.com/desktops/images/ship1280x1024.jpg`.

[2] See Brown et al.'s (1993, §6.2) definition of *Viterbi training* for a succinct justification of hard EM; in our case, the corresponding objective is Spitkovsky et al.'s (2010a, §7.1) $\hat{\theta}_{\text{VIT}}$.

System	*DDA (%)*
(Blunsom and Cohn, 2010)	**55.7**
(Gillenwater et al., 2010)	53.3
(Spitkovsky et al., 2010b)	50.4
+ soft EM + hard EM	52.8 (+**2.4**)
lexicalized, using hard EM	54.3 (+1.5)
+ soft EM + hard EM	**55.6** (+**1.3**)

Table 1: Directed dependency accuracies (DDA) on Section 23 of WSJ (all sentences) for recent state-of-the-art systems and our two experiments (one unlexicalized and one lexicalized) with a single alternation of lateen EM.

Algorithm #1) improve our recent publicly-available system for English dependency grammar induction. In study #2, we introduced a more sophisticated methodology that uses factorial designs and regressions to evaluate lateen strategies with unsupervised dependency parsing in many languages, after also controlling for other important sources of variation.

For study #1, our base system (Spitkovsky et al., 2010b) is an instance of the popular (unlexicalized) Dependency Model with Valence (Klein and Manning, 2004). This model was trained using hard EM on WSJ45 (WSJ sentences up to length 45) until successive changes in per-token cross-entropy fell below 2^{-20} bits (Spitkovsky et al., 2010b; 2010a, §4).[3]

We confirmed that the base model had indeed converged, by running 10 steps of hard EM on WSJ45 and verifying that its objective did not change much. Next, we applied a single alternation of simple lateen EM: first running soft EM (this took 101 steps, using the same termination criterion), followed by hard EM (again to convergence — another 23 iterations). The result was a decrease in hard EM's cross-entropy, from 3.69 to 3.59 bits per token (bpt), accompanied by a 2.4% jump in accuracy, from 50.4 to 52.8%, on Section 23 of WSJ (see Table 1).[4]

Our first experiment showed that lateen EM holds promise for simple models. Next, we tested it in a more realistic setting, by re-estimating *lexicalized* models,[5] starting from the unlexicalized model's parses; this took 24 steps with hard EM. We then applied another single lateen alternation: This time, soft EM ran for 37 steps, hard EM took another 14, and the new model again improved, by 1.3%, from 54.3 to 55.6% (see Table 1); the corresponding drop in (lexicalized) cross-entropy was from 6.10 to 6.09 bpt. This last model is competitive with the state-of-the-art; moreover, gains from single applications of simple lateen alternations (2.4 and 1.3%) are on par with the increase due to lexicalization alone (1.5%).

4 Methodology for Study #2

Study #1 suggests that lateen EM can improve grammar induction in English. To establish statistical significance, however, it is important to test a hypothesis in many settings (Ioannidis, 2005). We therefore use a factorial experimental design and regression analyses with a variety of lateen strategies. Two regressions — one predicting accuracy, the other, the number of iterations — capture the effects that lateen algorithms have on performance and efficiency, relative to standard EM training. We controlled for important dimensions of variation, such as the underlying language: to make sure that our results are not English-specific, we induced grammars in 19 languages. We also explored the impact from the quality of an initial model (using both uniform and ad hoc initializers), the choice of a primary objective (i.e., soft or hard EM), and the quantity and complexity of training data (shorter versus both short and long sentences). Appendix A gives the full details.

4.1 Data Sets

We use all 23 train/test splits from the 2006/7 CoNLL shared tasks (Buchholz and Marsi, 2006; Nivre et al., 2007),[6] which cover 19 different languages.[7] We splice out all punctuation labeled in the data, as is standard practice (Paskin, 2001; Klein and Manning, 2004), introducing new arcs from grandmothers to grand-daughters where necessary, both in train- and test-sets. Evaluation is always against the

[3] `http://nlp.stanford.edu/pubs/markup-data.tar.bz2`: `dp.model.dmv`

[4] It is standard practice to convert gold labeled constituents from Penn English Treebank's Wall Street Journal (WSJ) portion (Marcus et al., 1993) into unlabeled reference dependency parses using deterministic "head-percolation" rules (Collins, 1999); sentence root symbols (but not punctuation) arcs count towards accuracies (Paskin, 2001; Klein and Manning, 2004).

[5] We used Headden et al.'s (2009) method (also the approach taken by the two stronger state-of-the-art systems): for words seen at least 100 times in the training corpus, gold part-of-speech tags are augmented with lexical items.

[6] These disjoint splits require smoothing; in the WSJ setting, training and test sets overlapped (Klein and Manning, 2004).

[7] We down-weigh languages appearing in both years — Arabic, Chinese, Czech and Turkish — by 50% in all our analyses.

entire resulting test sets (i.e., all sentence lengths).[8]

4.2 Grammar Models

In all remaining experiments we model grammars via the original DMV, which ignores punctuation; all models are unlexicalized, with gold part-of-speech tags for word classes (Klein and Manning, 2004).

4.3 Smoothing Mechanism

All unsmoothed models are smoothed immediately prior to evaluation; some of the baseline models are also smoothed during training. In both cases, we use the “add-one” (a.k.a. Laplace) smoothing algorithm.

4.4 Standard Convergence

We always halt an optimizer once a change in its objective’s consecutive cross-entropy values falls below 2^{-20} bpt (at which point we consider it “stuck”).

4.5 Scoring Function

We report directed accuracies — fractions of correctly guessed (unlabeled) dependency arcs, including arcs from sentence root symbols, as is standard practice (Paskin, 2001; Klein and Manning, 2004). Punctuation does not affect scoring, as it had been removed from all parse trees in our data (see §4.1).

5 Experiments

We now summarize our baseline models and briefly review the proposed lateen algorithms. For details of the default systems (standard soft and hard EM), all control variables and both regressions (against final accuracies and iteration counts) see Appendix A.

5.1 Baseline Models

We tested a total of six baseline models, experimenting with two types of alternatives: (i) strategies that perturb stuck models directly, by *smoothing*, ignoring secondary objectives; and (ii) *shallow* applications of a single EM step, ignoring convergence.

Baseline $B1$ alternates running standard EM to convergence and smoothing. A second baseline, $B2$, smooths after every step of EM instead. Another shallow baseline, $B3$, alternates single steps of soft and hard EM.[9] Three such baselines begin with hard EM (marked with the subscript h); and three more start with soft EM (marked with the subscript s).

5.2 Lateen Models

Ten models, $A\{1,2,3,4,5\}_{\{h,s\}}$, correspond to our lateen algorithms #1–5 (§2), starting with either hard or soft EM’s objective, to be used as the primary.

6 Results

		Soft EM		Hard EM	
	Model	Δa	Δi	Δa	Δi
Baselines	$B3$	**-2.7**	×**0.2**	**-2.0**	×**0.3**
	$B2$	+0.6	×**0.7**	+0.6	×1.2
	$B1$	0.0	×**2.0**	+0.8	×**3.7**
Algorithms	$A1$	0.0	×**1.3**	**+5.5**	×**6.5**
	$A2$	-0.0	×**1.3**	+1.5	×**3.6**
	$A3$	0.0	×**0.7**	-0.1	×**0.7**
	$A4$	0.0	×**0.8**	**+3.0**	×**2.1**
	$A5$	0.0	×**1.2**	**+2.9**	×**3.8**

Table 2: Estimated additive changes in directed dependency accuracy (Δa) and multiplicative changes in the number of iterations before terminating (Δi) for all baseline models and lateen algorithms, relative to standard training: soft EM (left) and hard EM (right). Bold entries are statistically different ($p < 0.01$) from zero, for Δa, and one, for Δi (details in Table 4 and Appendix A).

Not one baseline attained a statistically significant performance improvement. Shallow models $B3_{\{h,s\}}$, in fact, significantly lowered accuracy: by 2.0%, on average ($p \approx 7.8 \times 10^{-4}$), for $B3_h$, which began with hard EM; and down 2.7% on average ($p \approx 6.4 \times 10^{-7}$), for $B3_s$, started with soft EM. They were, however, 3–5x faster than standard training, on average (see Table 4 for all estimates and associated p-values; above, Table 2 shows a preview of the full results).

6.1 $A1_{\{h,s\}}$ — Simple Lateen EM

$A1_h$ runs 6.5x slower, but scores 5.5% higher, on average, compared to standard Viterbi training; $A1_s$ is only 30% slower than standard soft EM, but does not impact its accuracy at all, on average.

Figure 2 depicts a sample training run: Italian ’07 with $A1_h$. Viterbi EM converges after 47 iterations,

[8] With the exception of Arabic ’07, from which we discarded a single sentence containing 145 non-punctuation tokens.

[9] It approximates a mixture (the average of soft and hard objectives) — a natural comparison, computable via gradients and standard optimization algorithms, such as L-BFGS (Liu and Nocedal, 1989). We did not explore exact interpolations, however, because replacing EM is itself a significant confounder, even with unchanged objectives (Berg-Kirkpatrick et al., 2010).

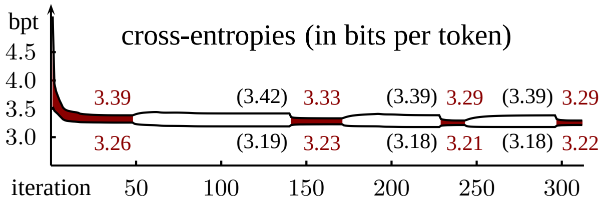

Figure 2: Cross-entropies for Italian '07, initialized uniformly and trained on sentences up to length 45. The two curves are primary and secondary objectives (soft EM's lies below, as sentence yields are at least as likely as parse trees): shaded regions indicate iterations of hard EM (primary); and annotated values are measurements upon each optimizer's convergence (soft EM's are parenthesized).

reducing the primary objective to 3.39 bpt (the secondary is then at 3.26); accuracy on the held-out set is 41.8%. Three alternations of lateen EM (totaling 265 iterations) further decrease the primary objective to 3.29 bpt (the secondary also declines, to 3.22) and accuracy increases to 56.2% (14.4% higher).

6.2 $A2_{\{h,s\}}$ — Shallow Lateen EM

$A2_h$ runs 3.6x slower, but scores only 1.5% higher, on average, compared to standard Viterbi training; $A2_s$ is again 30% slower than standard soft EM and also has no measurable impact on parsing accuracy.

6.3 $A3_{\{h,s\}}$ — Early-Stopping Lateen EM

Both $A3_h$ and $A3_s$ run 30% faster, on average, than standard training with hard or soft EM; and neither heuristic causes a statistical change to accuracy.

Table 3 shows accuracies and iteration counts for 10 (of 23) train/test splits that terminate early with $A3_s$ (in one particular, example setting). These runs are nearly twice as fast, and only two score (slightly) lower, compared to standard training using soft EM.

6.4 $A4_{\{h,s\}}$ — Early-Switching Lateen EM

$A4_h$ runs only 2.1x slower, but scores only 3.0% higher, on average, compared to standard Viterbi training; $A4_s$ is, in fact, 20% faster than standard soft EM, but still has no measurable impact on accuracy.

6.5 $A5_{\{h,s\}}$ — Partly-Switching Lateen EM

$A5_h$ runs 3.8x slower, scoring 2.9% higher, on average, compared to standard Viterbi training; $A5_s$ is 20% slower than soft EM, but, again, no more accurate. Indeed, $A4$ strictly dominates both $A5$ variants.

CoNLL Year		Soft EM		$A3_s$	
& Language		*DDA*	*iters*	*DDA*	*iters*
Arabic	2006	28.4	180	28.4	118
Bulgarian	'06	39.1	253	**39.6**	131
Chinese	'06	49.4	268	49.4	204
Dutch	'06	21.3	246	**27.8**	35
Hungarian	'07	17.1	366	**17.4**	213
Italian	'07	39.6	194	39.6	164
Japanese	'06	56.6	113	56.6	93
Portuguese	'06	37.9	180	*37.5*	102
Slovenian	'06	30.8	234	**31.1**	118
Spanish	'06	33.3	125	*33.1*	73
	Average:	35.4	216	36.1	125

Table 3: Directed dependency accuracies (DDA) and iteration counts for the 10 (of 23) train/test splits affected by early termination (setting: soft EM's primary objective, trained using shorter sentences and ad-hoc initialization).

7 Discussion

Lateen strategies improve dependency grammar induction in several ways. Early stopping offers a clear benefit: 30% higher efficiency yet same performance as standard training. This technique could be used to (more) fairly compare learners with radically different objectives (e.g., lexicalized and unlexicalized), requiring quite different numbers of steps (or magnitude changes in cross-entropy) to converge.

The second benefit is improved performance, but only starting with hard EM. Initial local optima discovered by soft EM are such that the impact on accuracy of all subsequent heuristics is indistinguishable from noise (it's not even negative). But for hard EM, lateen strategies consistently improve accuracy — by 1.5, 3.0 or 5.5% — as an algorithm follows the secondary objective longer (a single step, until the primary objective gets worse, or to convergence).

Our results suggest that soft EM should use early termination to improve efficiency. Hard EM, by contrast, could use any lateen strategy to improve either efficiency or performance, or to strike a balance.

8 Related Work

8.1 Avoiding and/or Escaping Local Attractors

Simple lateen EM is similar to Dhillon et al.'s (2002) refinement algorithm for text clustering with spherical k-means. Their "ping-pong" strategy alternates batch and incremental EM, exploits the strong points of each, and improves a *shared* objective at every step. Unlike generalized (GEM) variants (Neal and Hinton, 1999), lateen EM uses multiple objectives: it sacrifices the primary in the short run, to escape local optima; in the long run, it also does no harm, by construction (as it returns the best model seen).

Of the meta-heuristics that use more than a standard, scalar objective, deterministic annealing (DA) (Rose, 1998) is closest to lateen EM. DA perturbs objective functions, instead of manipulating solutions directly. As other continuation methods (Allgower and Georg, 1990), it optimizes an easy (e.g., convex) function first, then "rides" that optimum by gradually morphing functions towards the difficult objective; each step reoptimizes from the previous approximate solution. Smith and Eisner (2004) employed DA to improve part-of-speech disambiguation, but found that objectives had to be further "skewed," using domain knowledge, before it helped (constituent) grammar induction. (For this reason, we did not experiment with DA, despite its strong similarities to lateen EM.) Smith and Eisner (2004) used a "temperature" β to anneal a flat uniform distribution ($\beta = 0$) into soft EM's non-convex objective ($\beta = 1$). In their framework, hard EM corresponds to $\beta \longrightarrow \infty$, so the algorithms differ only in their β-schedule: DA's is continuous, from 0 to 1; lateen EM's is a discrete alternation, of 1 and $+\infty$.[10]

8.2 Terminating Early, Before Convergence

EM is rarely run to (even numerical) convergence. Fixing a modest number of iterations a priori (Klein, 2005, §5.3.4), running until successive likelihood ratios become small (Spitkovsky et al., 2009, §4.1) or using a combination of the two (Ravi and Knight, 2009, §4, Footnote 5) is standard practice in NLP. Elworthy's (1994, §5, Figure 1) analysis of part-of-speech tagging showed that, in most cases, a small number of iterations is actually preferable to convergence, in terms of final accuracies: "regularization by early termination" had been suggested for image deblurring algorithms in statistical astronomy (Lucy, 1974, §2); and validation against held-out data — a strategy proposed much earlier, in psychology (Larson, 1931), has also been used as a halting criterion in NLP (Yessenalina et al., 2010, §4.2, 5.2). Early-stopping lateen EM tethers termination to a *sign* change in the direction of a secondary objective, similarly to (cross-)validation (Stone, 1974; Geisser, 1975; Arlot and Celisse, 2010), but without splitting data — it trains using all examples, at all times.[11,12]

8.3 Training with Multiple Views

Lateen strategies may seem conceptually related to co-training (Blum and Mitchell, 1998). However, bootstrapping methods generally begin with some labeled data and gradually label the rest (discriminatively) as they grow more confident, but do not optimize an explicit objective function; EM, on the other hand, can be fully unsupervised, relabels all examples on each iteration (generatively), and guarantees not to hurt a well-defined objective, at every step.[13] Co-training classically relies on two views of the data — redundant feature sets that allow different algorithms to label examples for each other, yielding "probably approximately correct" (PAC)-style guarantees under certain (strong) assumptions. In contrast, lateen EM uses the same data, features, model and essentially the same algorithms, changing only their objective functions: it makes no assumptions, but guarantees not to harm the primary objective.

Some of these distinctions have become blurred with time: Collins and Singer (1999) introduced an objective function (also based on agreement) into co-training; Goldman and Zhou (2000), Ng and Cardie (2003) and Chan et al. (2004) made do without redundant views; Balcan et al. (2004) relaxed other strong assumptions; and Zhou and Goldman (2004) generalized co-training to accommodate three and more algorithms. Several such methods have been applied to dependency parsing (Søgaard and Rishøj, 2010), constituent parsing (Sarkar,

[10] One can think of this as a kind of "beam search" (Lowerre, 1976), with soft EM expanding and hard EM pruning a frontier.

[11] We see in it a milder contrastive estimation (Smith and Eisner, 2005a; 2005b), agnostic to implicit negative evidence, but caring *whence* learners push probability mass towards training examples: when most likely parse trees begin to benefit at the expense of their sentence yields (or vice versa), optimizers halt.

[12] For a recently proposed instance of EM that uses cross-validation (CV) to optimize *smoothed* data likelihoods (in learning synchronous PCFGs, for phrase-based machine translation), see Mylonakis and Sima'an's (2010, §3.1) CV-EM algorithm.

[13] Some authors (Nigam and Ghani, 2000; Ng and Cardie, 2003; Smith and Eisner, 2005a, §5.2, 7; §2; §6) draw a hard line between bootstrapping algorithms, such as self- and co-training, and probabilistic modeling using EM; others (Dasgupta et al., 2001; Chang et al., 2007, §1; §5) tend to lump them together.

2001) and parser reranking (Crim, 2002). Fundamentally, co-training exploits redundancies in unlabeled data and/or learning algorithms. Lateen strategies *also* exploit redundancies: in noisy objectives. Both approaches use a second vantage point to improve their perception of difficult training terrains.

9 Conclusions and Future Work

Lateen strategies can improve performance and efficiency for dependency grammar induction with the DMV. Early-stopping lateen EM is 30% faster than standard training, without affecting accuracy — it reduces guesswork in terminating EM. At the other extreme, simple lateen EM is slower, but significantly improves accuracy — by 5.5%, on average — for hard EM, escaping some of its local optima.

It would be interesting to apply lateen algorithms to advanced parsing models (Blunsom and Cohn, 2010; Headden et al., 2009, *inter alia*) and learning algorithms (Gillenwater et al., 2010; Cohen and Smith, 2009, *inter alia*). Future work could explore other NLP tasks — such as clustering, sequence labeling, segmentation and alignment — that often employ EM. Our meta-heuristics are multi-faceted, featuring aspects of iterated local search, deterministic annealing, cross-validation, contrastive estimation and co-training. They may be generally useful in machine learning and non-convex optimization.

Appendix A. Experimental Design

Statistical techniques are vital to many aspects of computational linguistics (Johnson, 2009; Charniak, 1997; Abney, 1996, *inter alia*). We used factorial designs,[14] which are standard throughout the natural and social sciences, to assist with experimental design and statistical analyses. Combined with ordinary regressions, these methods provide succinct and interpretable summaries that explain which settings meaningfully contribute to changes in dependent variables, such as running time and accuracy.

9.1 Dependent Variables

We constructed two regressions, for two types of dependent variables: to summarize performance, we predict accuracies; and to summarize efficiency, we predict (logarithms of) iterations before termination.

In the performance regression, we used four different scores for the dependent variable. These include both directed accuracies and *undirected* accuracies, each computed in two ways: (i) using a best parse tree; and (ii) using all parse trees. These four types of scores provide different kinds of information. Undirected scores ignore polarity of parent-child relations (Paskin, 2001; Klein and Manning, 2004; Schwartz et al., 2011), partially correcting for some effects of alternate analyses (e.g., systematic choices between modals and main verbs for heads of sentences, determiners for noun phrases, etc.). And *integrated* scoring, using the inside-outside algorithm (Baker, 1979) to compute expected accuracy across all — not just best — parse trees, has the advantage of incorporating probabilities assigned to individual arcs: This metric is more sensitive to the margins that separate best from next-best parse trees, and is not affected by tie-breaking. We tag scores using two binary predictors in a simple (first order, multi-linear) regression, where having multiple relevant quality assessments improves goodness-of-fit.

In the efficiency regression, dependent variables are logarithms of the numbers of iterations. Wrapping EM in an inner loop of a heuristic has a multiplicative effect on the total number of models re-estimated prior to termination. Consequently, logarithms of the final counts better fit the observed data.

9.2 Independent Predictors

All of our predictors are binary indicators (a.k.a. "dummy" variables). The *undirected* and *integrated* factors only affect the regression for accuracies (see Table 4, left); remaining factors participate also in the running times regression (see Table 4, right). In a default run, all factors are zero, corresponding to the intercept estimated by a regression; other estimates reflect changes in the dependent variable associated with having that factor "on" instead of "off."

- *adhoc* — This setting controls initialization. By default, we use the uninformed uniform initializer (Spitkovsky et al., 2010a); when it is

[14] We used *full* factorial designs for clarity of exposition. But many fewer experiments would suffice, especially in regression models without interaction terms: for the more efficient *fractional* factorial designs, as well as for randomized block designs and full factorial designs, see Montgomery (2005, Ch. 4–9).

		Regression for **Accuracies**		Regression for $\ln$(**Iterations**)		
	Goodness-of-Fit:	($R^2_{adj} \approx 76.2\%$)		($R^2_{adj} \approx 82.4\%$)		
Model	*Indicator Factors*	*coeff.* $\hat{\beta}$	*adj. p-value*	*coeff.* $\hat{\beta}$	*mult.* $e^{\hat{\beta}}$	*adj. p-value*
	undirected	**18.1**	$< 2.0 \times 10^{-16}$			
	integrated	**-0.9**	$\approx 7.0 \times 10^{-7}$			
	(intercept)	**30.9**	$< 2.0 \times 10^{-16}$	5.5	**255.8**	$< 2.0 \times 10^{-16}$
	adhoc	**1.2**	$\approx 3.1 \times 10^{-13}$	-0.0	1.0	≈ 1.0
	sweet	**1.0**	$\approx 3.1 \times 10^{-9}$	-0.2	**0.8**	$< 2.0 \times 10^{-16}$
$B3_s$	*shallow (soft-first)*	**-2.7**	$\approx 6.4 \times 10^{-7}$	-1.5	**0.2**	$< 2.0 \times 10^{-16}$
$B3_h$	*shallow (hard-first)*	**-2.0**	$\approx 7.8 \times 10^{-4}$	-1.2	**0.3**	$< 2.0 \times 10^{-16}$
$B2_s$	*shallow smooth*	0.6	≈ 1.0	-0.4	**0.7**	$\approx 1.4 \times 10^{-12}$
$B1_s$	*smooth*	0.0	≈ 1.0	0.7	**2.0**	$< 2.0 \times 10^{-16}$
$A1_s$	*simple lateen*	0.0	≈ 1.0	0.2	**1.3**	$\approx 4.1 \times 10^{-4}$
$A2_s$	*shallow lateen*	-0.0	≈ 1.0	0.2	**1.3**	$\approx 5.8 \times 10^{-4}$
$A3_s$	*early-stopping lateen*	0.0	≈ 1.0	-0.3	**0.7**	$\approx 2.6 \times 10^{-7}$
$A4_s$	*early-switching lateen*	0.0	≈ 1.0	-0.3	**0.8**	$\approx 2.6 \times 10^{-7}$
$A5_s$	*partly-switching lateen*	0.0	≈ 1.0	0.2	**1.2**	$\approx 4.2 \times 10^{-3}$
	viterbi	**-4.0**	$\approx 5.7 \times 10^{-16}$	-1.7	**0.2**	$< 2.0 \times 10^{-16}$
$B2_h$	*shallow smooth*	0.6	≈ 1.0	0.2	1.2	$\approx 5.6 \times 10^{-2}$
$B1_h$	*smooth*	0.8	≈ 1.0	1.3	**3.7**	$< 2.0 \times 10^{-16}$
$A1_h$	*simple lateen*	**5.5**	$< 2.0 \times 10^{-16}$	1.9	**6.5**	$< 2.0 \times 10^{-16}$
$A2_h$	*shallow lateen*	1.5	$\approx 5.0 \times 10^{-2}$	1.3	**3.6**	$< 2.0 \times 10^{-16}$
$A3_h$	*early-stopping lateen*	-0.1	≈ 1.0	-0.4	**0.7**	$\approx 1.7 \times 10^{-11}$
$A4_h$	*early-switching lateen*	**3.0**	$\approx 1.0 \times 10^{-8}$	0.7	**2.1**	$< 2.0 \times 10^{-16}$
$A5_h$	*partly-switching lateen*	**2.9**	$\approx 7.6 \times 10^{-8}$	1.3	**3.8**	$< 2.0 \times 10^{-16}$

Table 4: Regressions for accuracies and natural-log-iterations, using 86 binary predictors (all p-values jointly adjusted for simultaneous hypothesis testing; $\{$*langyear*$\}$ indicators not shown). Accuracies' estimated coefficients $\hat{\beta}$ that are statistically different from 0 — and iteration counts' multipliers $e^{\hat{\beta}}$ significantly different from 1 — are shown in bold.

on, we use Klein and Manning's (2004) "ad-hoc" harmonic heuristic, bootstrapped using sentences up to length 10, from the training set.

- *sweet* — This setting controls the length cutoff. By default, we train with all sentences containing up to 45 tokens; when it is on, we use Spitkovsky et al.'s (2009) "sweet spot" cutoff of 15 tokens (recommended for English, WSJ).
- *viterbi* — This setting controls the primary objective of the learning algorithm. By default, we run soft EM; when it is on, we use hard EM.
- $\{$*langyear*$_i\}_{i=1}^{22}$ — This is a set of 22 mutually-exclusive selectors for the language/year of a train/test split; default (all zeros) is English '07.

Due to space limitations, we exclude *langyear* predictors from Table 4. Further, we do not explore (even two-way) interactions between predictors.[15]

9.3 Statistical Significance

Our statistical analyses relied on the R package (R Development Core Team, 2011), which does not, by default, adjust statistical significance (p-values) for multiple hypotheses testing.[16] We corrected this using the Holm-Bonferroni method (Holm, 1979), which is uniformly more powerful than the older (Dunn-)Bonferroni procedure; since we tested many fewer hypotheses ($44 + 42$ — one per intercept/coefficient $\hat{\beta}$) than settings combinations, its adjustments to the p-values are small (see Table 4).[17]

[15] This approach may miss some interesting facts, e.g., that the *adhoc* initializer is exceptionally good for English, with soft EM. Instead it yields coarse summaries of regularities supported by overwhelming evidence across data and training regimes.

[16] Since we would *expect* $p\%$ of randomly chosen hypotheses to appear significant at the $p\%$ level simply by *chance*, we must take precautions against these and other "data-snooping" biases.

[17] We adjusted the p-values for all 86 hypotheses jointly, using `http://rss.acs.unt.edu/Rdoc/library/multtest/html/mt.rawp2adjp.html`.

CoNLL Year	$A3_s$		Soft EM		$A3_h$		Hard EM		$A1_h$	
& Language	*DDA*	*iters*	*DDA*	*iters*	*DDA*	*iters*	*DDA*	*iters*	*DDA*	*iters*
Arabic 2006	28.4	118	28.4	162	21.6	19	21.6	21	**32.1**	200
’7	–	–	**26.9**	171	24.7	17	24.8	24	22.0	239
Basque ’7	–	–	39.9	180	32.0	16	32.2	20	**43.6**	128
Bulgarian ’6	39.6	131	39.1	253	41.6	22	41.5	25	**44.3**	140
Catalan ’7	–	–	58.5	135	50.1	48	50.1	54	**63.8**	279
Chinese ’6	**49.4**	204	49.4	268	31.3	24	31.6	55	37.9	378
’7	–	–	**46.0**	262	30.0	25	30.2	64	34.5	307
Czech ’6	–	–	**50.5**	294	27.8	27	27.7	33	35.2	445
’7	–	–	**49.8**	263	29.0	37	29.0	41	31.4	307
Danish ’6	–	–	43.5	116	43.8	31	43.9	45	**44.0**	289
Dutch ’6	27.8	35	21.3	246	24.9	44	24.9	49	**32.5**	241
English ’7	–	–	**38.1**	180	34.0	32	33.9	42	34.9	186
German ’6	–	–	33.3	136	25.4	20	25.4	39	**33.5**	155
Greek ’7	–	–	17.5	230	18.3	18	18.3	21	**21.4**	117
Hungarian ’7	17.4	213	17.1	366	12.3	26	12.4	36	**23.0**	246
Italian ’7	**39.6**	164	39.6	194	32.6	25	32.6	27	37.6	273
Japanese ’6	**56.6**	93	56.6	113	49.6	20	49.7	23	53.5	91
Portuguese ’6	37.5	102	**37.9**	180	28.6	27	28.9	41	34.4	134
Slovenian ’6	31.1	118	30.8	234	–	–	23.4	22	**33.6**	255
Spanish ’6	33.1	73	**33.3**	125	18.2	29	18.4	36	33.3	235
Swedish ’6	–	–	41.8	242	36.0	24	36.1	29	**42.5**	296
Turkish ’6	–	–	29.8	303	17.8	19	22.2	38	**31.9**	134
’7	–	–	28.3	227	14.0	9	10.7	31	**33.4**	242
Average:	**37.4**	162	37.0	206	30.0	26	30.0	35	37.1	221

Table 5: Performance (directed dependency accuracies measured against all sentences in the evaluation sets) and efficiency (numbers of iterations) for standard training (soft and hard EM), early-stopping lateen EM ($A3$) and simple lateen EM with hard EM's primary objective ($A1_h$), for all 23 train/test splits, with *adhoc* and *sweet* settings on.

9.4 Interpretation

Table 4 shows the estimated coefficients and their (adjusted) p-values for both intercepts and most predictors (excluding the language/year of the data sets) for all 1,840 experiments. The default (English) system uses soft EM, trains with both short and long sentences, and starts from an uninformed uniform initializer. It is estimated to score 30.9%, converging after approximately 256 iterations (both intercepts are statistically different from zero: $p < 2.0 \times 10^{-16}$).

As had to be the case, we detect a gain from *undirected* scoring; *integrated* scoring is slightly (but significantly: $p \approx 7.0 \times 10^{-7}$) negative, which is reassuring: best parses are scoring higher than the rest and may be standing out by large margins. The *adhoc* initializer boosts accuracy by 1.2%, overall (also significant: $p \approx 3.1 \times 10^{-13}$), without a measurable impact on running time ($p \approx 1.0$). Training with fewer, shorter sentences, at the *sweet* spot gradation, adds 1.0% and shaves 20% off the total number of iterations, on average (both estimates are significant).

We find the *viterbi* objective harmful — by 4.0%, on average ($p \approx 5.7 \times 10^{-16}$) — for the CoNLL sets. Spitkovsky et al. (2010a) reported that it helps on WSJ, at least with long sentences and uniform initializers. Half of our experiments are with shorter sentences, and half use ad hoc initializers (i.e., three quarters of settings are not ideal for Viterbi EM), which may have contributed to this negative result; still, our estimates do confirm that hard EM is significantly (80%, $p < 2.0 \times 10^{-16}$) faster than soft EM.

9.5 More on Viterbi Training

The overall negative impact of Viterbi objectives is a cause for concern: On average, $A1_h$'s estimated gain of 5.5% should more than offset the expected 4.0% loss from starting with hard EM. But it is, nevertheless, important to make sure that simple lateen EM with hard EM's primary objective is in fact an improvement over *both* standard EM algorithms.

Table 5 shows performance and efficiency numbers for $A1_h$, $A3_{\{h,s\}}$, as well as standard soft and hard EM, using settings that are least favorable for

CoNLL Year	$A3_s$		Soft EM		$A3_h$		Hard EM		$A1_h$	
& Language	*DDA*	*iters*	*DDA*	*iters*	*DDA*	*iters*	*DDA*	*iters*	*DDA*	*iters*
Arabic 2006	–	–	**33.4**	317	20.8	8	20.2	32	16.6	269
’7	18.6	60	8.7	252	26.5	9	26.4	14	**49.5**	171
Basque ’7	–	–	18.3	245	23.2	16	23.0	23	**24.0**	162
Bulgarian ’6	27.0	242	27.1	293	40.6	33	40.5	34	**43.9**	276
Catalan ’7	15.0	74	13.8	159	53.2	30	53.1	31	**59.8**	176
Chinese ’6	63.5	131	**63.6**	261	36.8	45	36.8	47	44.5	213
’7	**58.5**	130	58.5	258	35.2	20	35.0	48	43.2	372
Czech ’6	29.5	125	**29.7**	224	23.6	18	23.8	41	27.7	179
’7	–	–	25.9	215	27.1	37	27.2	64	**28.4**	767
Danish ’6	–	–	16.6	155	28.7	30	28.7	30	**38.3**	241
Dutch ’6	20.4	51	21.2	174	25.5	30	25.6	38	**27.8**	243
English ’7	–	–	18.0	162	–	–	38.7	35	**45.2**	366
German ’6	–	–	24.4	148	30.1	39	30.1	44	**30.4**	185
Greek ’7	**25.5**	133	25.3	156	–	–	13.2	27	13.2	252
Hungarian ’7	–	–	18.9	310	28.9	34	28.9	44	**34.7**	414
Italian ’7	25.4	127	25.3	165	–	–	**52.3**	36	52.3	81
Japanese ’6	–	–	39.3	143	42.2	38	42.4	48	**50.2**	199
Portuguese ’6	35.2	48	35.6	224	–	–	34.5	21	**36.7**	143
Slovenian ’6	24.8	182	25.3	397	28.8	17	28.8	20	**32.2**	121
Spanish ’6	–	–	27.7	252	–	–	28.3	31	**50.6**	130
Swedish ’6	27.9	49	32.6	287	45.2	22	45.6	52	**50.0**	314
Turkish ’6	–	–	30.5	239	30.2	16	**30.6**	24	29.0	138
’7	–	–	**48.8**	254	34.3	24	33.1	34	35.9	269
Average:	27.3	161	27.3	225	33.2	28	33.2	35	**38.2**	236

Table 6: Performance (directed dependency accuracies measured against all sentences in the evaluation sets) and efficiency (numbers of iterations) for standard training (soft and hard EM), early-stopping lateen EM ($A3$) and simple lateen EM with hard EM’s primary objective ($A1_h$), for all 23 train/test splits, with setting *adhoc* off and *sweet* on.

Viterbi training: *adhoc* and *sweet* on. Although $A1_h$ scores 7.1% higher than hard EM, on average, it is only slightly better than soft EM — up 0.1% (and worse than $A1_s$). Without *adhoc* (i.e., using uniform initializers — see Table 6), however, hard EM still improves, by 3.2%, on average, whereas soft EM drops nearly 10%; here, $A1_h$ further improves over hard EM, scoring 38.2% (up 5.0), higher than soft EM’s accuracies from *both* settings (27.3 and 37.0).

This suggests that $A1_h$ is indeed better than both standard EM algorithms. We suspect that our experimental set-up may be disadvantageous for Viterbi training, since half the settings use ad hoc initializers, and because CoNLL sets are small. (Viterbi EM works best with more data and longer sentences.)

Acknowledgments

Partially funded by the Air Force Research Laboratory (AFRL), under prime contract no. FA8750-09-C-0181, and by NSF, via award #IIS-0811974. We thank Angel X. Chang, Spence Green, David McClosky, Fernando Pereira, Slav Petrov and the anonymous reviewers, for many helpful comments on draft versions of this paper, and Andrew Y. Ng, for a stimulating discussion. First author is grateful to Lynda K. Dunnigan for first introducing him to lateen sails, among other connections, in *Humanities*.

References

S. Abney. 1996. Statistical methods and linguistics. In J. L. Klavans and P. Resnik, editors, *The Balancing Act: Combining Symbolic and Statistical Approaches to Language*. MIT Press.

E. L. Allgower and K. Georg. 1990. *Numerical Continuation Methods: An Introduction*. Springer-Verlag.

S. Arlot and A. Celisse. 2010. A survey of cross-validation procedures for model selection. *Statistics Surveys*, 4.

J. K. Baker. 1979. Trainable grammars for speech recognition. In *Speech Communication Papers for the 97th Meeting of the Acoustical Society of America*.

M.-F. Balcan, A. Blum, and K. Yang. 2004. Co-training and expansion: Towards bridging theory and practice. In *NIPS*.

T. Berg-Kirkpatrick, A. Bouchard-Côté, J. DeNero, and D. Klein. 2010. Painless unsupervised learning with features. In *NAACL-HLT*.

A. Blum and T. Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *COLT*.

P. Blunsom and T. Cohn. 2010. Unsupervised induction of tree substitution grammars for dependency parsing. In *EMNLP*.

P. F. Brown, V. J. Della Pietra, S. A. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19.

S. Buchholz and E. Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *CoNLL*.

J. Chan, I. Koprinska, and J. Poon. 2004. Co-training with a single natural feature set applied to email classification. In *WI*.

M.-W. Chang, L. Ratinov, and D. Roth. 2007. Guiding semi-supervision with constraint-driven learning. In *ACL*.

E. Charniak. 1997. Statistical techniques for natural language parsing. *AI Magazine*, 18.

S. B. Cohen and N. A. Smith. 2009. Shared logistic normal distributions for soft parameter tying in unsupervised grammar induction. In *NAACL-HLT*.

M. Collins and Y. Singer. 1999. Unsupervised models for named entity classification. In *EMNLP*.

M. Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

J. Crim. 2002. Co-training re-rankers for improved parser accuracy.

S. Dasgupta, M. L. Littman, and D. McAllester. 2001. PAC generalization bounds for co-training. In *NIPS*.

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.

I. S. Dhillon, Y. Guan, and J. Kogan. 2002. Iterative clustering of high dimensional text data augmented by local search. In *ICDM*.

D. Elworthy. 1994. Does Baum-Welch re-estimation help taggers? In *ANLP*.

S. Geisser. 1975. The predictive sample reuse method with applications. *Journal of the American Statistical Association*, 70.

J. Gillenwater, K. Ganchev, J. Graça, F. Pereira, and B. Taskar. 2010. Posterior sparsity in unsupervised dependency parsing. Technical report, University of Pennsylvania.

S. Goldman and Y. Zhou. 2000. Enhancing supervised learning with unlabeled data. In *ICML*.

W. P. Headden, III, M. Johnson, and D. McClosky. 2009. Improving unsupervised dependency parsing with richer contexts and smoothing. In *NAACL-HLT*.

S. Holm. 1979. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6.

J. P. A. Ioannidis. 2005. Why most published research findings are false. *PLoS Medicine*, 2.

M. Johnson. 2009. How the statistical revolution changes (computational) linguistics. In *EACL: Interaction between Linguistics and Computational Linguistics: Virtuous, Vicious or Vacuous?*

M. Kearns, Y. Mansour, and A. Y. Ng. 1997. An information-theoretic analysis of hard and soft assignment methods for clustering. In *UAI*.

D. Klein and C. D. Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *ACL*.

D. Klein. 2005. *The Unsupervised Learning of Natural Language Structure*. Ph.D. thesis, Stanford University.

S. C. Larson. 1931. The shrinkage of the coefficient of multiple correlation. *Journal of Educational Psychology*, 22.

P. Liang and D. Klein. 2008. Analyzing the errors of unsupervised learning. In *HLT-ACL*.

D. C. Liu and J. Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming. Series B*, 45.

B. T. Lowerre. 1976. *The* HARPY *Speech Recognition System*. Ph.D. thesis, CMU.

L. B. Lucy. 1974. An iterative technique for the rectification of observed distributions. *The Astronomical Journal*, 79.

M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19.

X.-L. Meng. 2007. EM and MCMC: Workhorses for scientific computing (thirty years of EM and much more). *Statistica Sinica*, 17.

B. Merialdo. 1994. Tagging English text with a probabilistic model. *Computational Linguistics*, 20.

D. C. Montgomery. 2005. *Design and Analysis of Experiments*. John Wiley & Sons, 6th edition.

M. Mylonakis and K. Sima'an. 2010. Learning probabilistic synchronous CFGs for phrase-based translation. In *CoNLL*.

R. M. Neal and G. E. Hinton. 1999. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*. MIT Press.

V. Ng and C. Cardie. 2003. Weakly supervised natural language learning without redundant views. In *HLT-NAACL*.

K. Nigam and R. Ghani. 2000. Analyzing the effectiveness and applicability of co-training. In *CIKM*.

J. Nivre, J. Hall, S. Kübler, R. McDonald, J. Nilsson, S. Riedel, and D. Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *EMNLP-CoNLL*.

M. A. Paskin. 2001. Grammatical bigrams. In *NIPS*.

F. Pereira and Y. Schabes. 1992. Inside-outside reestimation from partially bracketed corpora. In *ACL*.

R Development Core Team, 2011. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing.

S. Ravi and K. Knight. 2009. Minimized models for unsupervised part-of-speech tagging. In *ACL-IJCNLP*.

K. Rose. 1998. Deterministic annealing for clustering, compression, classification, regression and related optmization problems. *Proceedings of the IEEE*, 86.

A. Sarkar. 2001. Applying co-training methods to statistical parsing. In *NAACL*.

R. Schwartz, O. Abend, R. Reichart, and A. Rappoport. 2011. Neutralizing linguistically problematic annotations in unsupervised dependency parsing evaluation. In *ACL*.

N. A. Smith and J. Eisner. 2004. Annealing techniques for unsupervised statistical language learning. In *ACL*.

N. A. Smith and J. Eisner. 2005a. Contrastive estimation: Training log-linear models on unlabeled data. In *ACL*.

N. A. Smith and J. Eisner. 2005b. Guiding unsupervised grammar induction using contrastive estimation. In *IJCAI: Grammatical Inference Applications*.

A. Søgaard and C. Rishøj. 2010. Semi-supervised dependency parsing using generalized tri-training. In *COLING*.

V. I. Spitkovsky, H. Alshawi, and D. Jurafsky. 2009. Baby Steps: How "Less is More" in unsupervised dependency parsing. In *NIPS: Grammar Induction, Representation of Language and Language Learning*.

V. I. Spitkovsky, H. Alshawi, D. Jurafsky, and C. D. Manning. 2010a. Viterbi training improves unsupervised dependency parsing. In *CoNLL*.

V. I. Spitkovsky, D. Jurafsky, and H. Alshawi. 2010b. Profiting from mark-up: Hyper-text annotations for guided parsing. In *ACL*.

M. Stone. 1974. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society. Series B*, 36.

A. Yessenalina, Y. Yue, and C. Cardie. 2010. Multi-level structured models for document-level sentiment classification. In *EMNLP*.

Y. Zhou and S. Goldman. 2004. Democratic co-learning. In *ICTAI*.