Concavity and Initialization for Unsupervised Dependency Parsing

Kevin Gimpel and **Noah A. Smith**
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{kgimpel,nasmith}@cs.cmu.edu

Abstract

We investigate models for unsupervised learning with concave log-likelihood functions. We begin with the most well-known example, IBM Model 1 for word alignment (Brown et al., 1993) and analyze its properties, discussing why other models for unsupervised learning are so seldom concave. We then present concave models for dependency grammar induction and validate them experimentally. We find our concave models to be effective initializers for the dependency model of Klein and Manning (2004) and show that we can encode linguistic knowledge in them for improved performance.

1 Introduction

In NLP, unsupervised learning typically implies optimization of a "bumpy" objective function riddled with local maxima. However, one exception is IBM Model 1 (Brown et al., 1993) for word alignment, which is the only model commonly used for unsupervised learning in NLP that has a concave log-likelihood function.[1] For other models, such as those used in unsupervised part-of-speech tagging and grammar induction, and indeed for more sophisticated word alignment models, the log-likelihood function maximized by EM is non-concave. As a result, researchers are obligated to consider initialization in addition to model design (Klein and Manning, 2004; Goldberg et al., 2008).

For example, consider the dependency grammar induction results shown in Table 1 when training the widely used dependency model with valence (DMV; Klein and Manning, 2004). Using uniform distributions for initialization (UNIF) results in an accuracy of 17.6% on the test set, well below the baseline of attaching each word to its right neighbor (ATTACHRIGHT, 31.7%). Furthermore, when using a set of 50 random initializers (RAND), the standard deviation of the accuracy is an alarming 8.3%.

In light of this sensitivity to initialization, it is compelling to consider unsupervised models with concave log-likelihood functions, which may provide stable, data-supported initializers for more complex models. In this paper, we explore the issues involved with such an expedition and elucidate the limitations of such models for unsupervised NLP. We then present simple concave models for dependency grammar induction that are easy to implement and offer efficient optimization. We also show how linguistic knowledge can be encoded without sacrificing concavity. Using our models to initialize the DMV, we find that they lead to an improvement in average accuracy across 18 languages.

2 IBM Model 1 and Concavity

IBM Model 1 is a conditional model of a target-language sentence $\boldsymbol{e}$ of length m and an alignment $\boldsymbol{a}$ given a source-language sentence $\boldsymbol{f}$ of length l. The generation of m is assumed to occur with some (inconsequential) uniform probability ϵ. The alignment vector $\boldsymbol{a}$, a hidden variable, has an entry for each element of $\boldsymbol{e}$ that contains the index in $\boldsymbol{f}$ of the aligned word. These entries are used to define which translation parameters $t(e_j \mid f_{a_j})$ are active. Model 1 assumes that the probability of the ith ele-

[1] It is not *strictly* concave (Toutanova and Galley, 2011).

2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 577–581,
Montréal, Canada, June 3-8, 2012.

ment in $\boldsymbol{a}$, denoted $a(i \mid j, l, m)$, is simply a uniform distribution over all l source words plus the null word. These assumptions result in the following log-likelihood for a sentence pair $\langle \boldsymbol{f}, \boldsymbol{e} \rangle$ under Model 1 (marginalizing $\boldsymbol{a}$):

$$\log p(\boldsymbol{e} \mid \boldsymbol{f}) = \log \tfrac{\epsilon}{(l+1)^m} + \sum_{j=1}^{m} \log \sum_{i=0}^{l} t(e_j \mid f_i) \tag{1}$$

The only parameters to be learned in the model are $\boldsymbol{t} = \{t(e \mid f)\}_{e,f}$. Since a parameter is concave in itself, the sum of concave functions is concave, and the log of a concave function is concave, Eq. 1 is concave in $\boldsymbol{t}$ (Brown et al., 1993).

IBM Model 2 involves a slight change to Model 1 in which the probability of a word link depends on the word positions. However, this change renders it no longer concave. Consider the log-likelihood function for Model 2:

$$\log \epsilon + \sum_{j=1}^{m} \log \sum_{i=0}^{l} t(e_j \mid f_i) \cdot a(i \mid j, l, m) \tag{2}$$

Eq. 2 is not concave in the parameters $t(e_j \mid f_i)$ and $a(i \mid j, l, m)$ because a product is neither convex nor concave in its vector of operands. This can be shown by computing the Hessian matrix of $f(x, y) = xy$ and showing that it is indefinite.

In general, concavity is lost when the log-likelihood function contains a product of model parameters enclosed within a $\log \sum$. If the sum is not present, the $\log$ can be used to separate the product of parameters, making the function concave. It can also be shown that a "featurized" version (Berg-Kirkpatrick et al., 2010) of Model 1 is not concave. More generally, any non-concave function enclosed within $\log \sum$ will cause the log-likelihood function to be non-concave, though there are few other non-concave functions with a probabilistic semantics than those just discussed.

3 Concave, Unsupervised Models

Nearly every other model used for unsupervised learning in NLP has a non-concave log-likelihood function. We now proceed to describe the conditions necessary to develop concave models for two tasks.

3.1 Part-of-Speech Tagging

Consider a standard first-order hidden Markov model for POS tagging. Letting $\boldsymbol{y}$ denote the tag sequence for a sentence $\boldsymbol{e}$ with m tokens, the single-example log-likelihood is:

$$\log \sum_{\boldsymbol{y}} p(\text{stop} \mid y_m) \prod_{j=1}^{m} p(y_j \mid y_{j-1}) \cdot p(e_j \mid y_j) \tag{3}$$

where y_0 is a designated "start" symbol. Unlike IBM Models 1 and 2, we cannot reverse the order of the summation and product here because the transition parameters $p(y_j \mid y_{j-1})$ cause each tag decision to affect its neighbors. Therefore, Eq. 3 is non-concave due to the presence of a product within a $\log \sum$.

However, if the tag transition probabilities $p(y_j \mid y_{j-1})$ are all constants and also do not depend on the previous tag y_{j-1}, then we can rewrite Eq. 3 as the following concave log-likelihood function (using $C(y)$ to denote a constant function of tag y, e.g., a fixed tag prior distribution):

$$\log C(\text{stop}) + \log \prod_{j=1}^{m} \sum_{y_j} C(y_j) \cdot p(e_j \mid y_j)$$

Lacking any transition modeling power, this model appears weak for POS tagging. However, we note that we can add additional conditioning information to the $p(e_j \mid y_j)$ distributions and retain concavity, such as nearby words and tag dictionary information. We speculate that such a model might learn useful patterns about local contexts and provide an initializer for unsupervised part-of-speech tagging.

3.2 Dependency Grammar Induction

To develop dependency grammar induction models, we begin with a version of Model 1 in which a sentence $\boldsymbol{e}$ is generated from a copy of itself (denoted $\boldsymbol{e}'$): $\log p(\boldsymbol{e} \mid \boldsymbol{e}')$

$$= \log \tfrac{\epsilon}{(m+1)^m} + \sum_{j=1}^{m} \log \sum_{i=0, i \neq j}^{m} c(e_j \mid e'_i) \tag{4}$$

If a word e_j is "aligned" to e'_0, e_j is a root. This is a simple child-generation model with no tree constraint. In order to preserve concavity, we are forbidden from conditioning on other parent-child assignments or including any sort of larger constraints.

However, we can condition the child distributions on additional information about $\boldsymbol{e}'$ since it is fully observed. This conditioning information may include the direction of the edge, its distance, and any properties about the words in the sentence. We found that conditioning on direction improved performance: we rewrite the c distributions as $c(e_j \mid e'_i, sign(j - i))$ and denote this model by CCV1.

We note that we can also include constraints in the sum over possible parents and still preserve concavity. Naseem et al. (2010) found that adding parent-child constraints to a grammar induction system can improve performance dramatically. We employ one simple rule: roots are likely to be verbs.[2] We modify Ccv1 to restrict the summation over parents to exclude e'_0 if the child word is not a verb.[3] We only employ this restriction during EM learning for sentences containing at least one verb. For sentences without verbs, we allow all words to be the root. We denote this model by Ccv2.

In related work, Brody (2010) also developed grammar induction models based on the IBM word alignment models. However, while our goal is to develop concave models, Brody employed Bayesian nonparametrics in his version of Model 1, which makes the model non-concave.

4 Experiments

We ran experiments to determine how well our concave grammar induction models Ccv1 and Ccv2 can perform on their own and when used as initializers for the DMV (Klein and Manning, 2004). The DMV is a generative model of POS tag sequences and projective dependency trees over them. It is the foundation of most state-of-the-art unsupervised grammar induction models (several of which are listed in Tab. 1). The model includes multinomial distributions for generating each POS tag given its parent and the direction of generation: where e_i is the parent POS tag and e_j the child tag, these distributions take the form $c(e_j \mid e_i, sign(j-i))$, analogous to the distributions used in our concave models. The DMV also has multinomial distributions for deciding whether to stop or continue generating children in each direction considering whether any children have already been generated in that direction.

The majority of researchers use the original initializer from Klein and Manning (2004), denoted here K&M. K&M is a deterministic harmonic initializer that sets parent-child token affinities inversely proportional to their distances, then normalizes to obtain probability distributions. K&M is often described as corresponding to an initial E step for an unspecified model that favors short attachments.

		Train ≤ 10		Train ≤ 20	
		Test		Test	
Model	Init.	≤ 10	$\leq \infty$	≤ 10	$\leq \infty$
AttRight	N/A	38.4	31.7	38.4	31.7
Ccv1	Unif	31.4	25.6	31.0	23.7
Ccv2	Unif	43.1	28.6	43.9	27.1
DMV	Unif	21.3	17.6	21.3	16.4
	Rand*	41.0	31.8	-	-
	K&M	44.1	32.9	51.9	37.8
	Ccv1	45.3	30.9	53.9	36.7
	Ccv2	54.3	43.0	**64.3**	**53.1**
Shared LN	K&M	61.3	41.4		
L-EVG	Rand†	**68.8**	-		
Feature DMV	K&M	63.0	-		
LexTSG-DMV	K&M	67.7	**55.7**		
Posterior Reg.	K&M	64.3	53.3		
Punc/UTags	K&M′			-	59.1‡

Table 1: English attachment accuracies on Section 23, for short sentences ($\leq$10 words) and all ($\leq\infty$). We include selected results on this same test set: Shared LN = Cohen and Smith (2009), L-EVG = Headden III et al. (2009), Feature DMV = Berg-Kirkpatrick et al. (2010), LexTSG-DMV = Blunsom and Cohn (2010), Posterior Reg. = Gillenwater et al. (2010), Punc/UTags = Spitkovsky et al. (2011a). K&M′ is from Spitkovsky et al. (2011b). *Accuracies are averages over 50 random initializers; $\sigma = 10.9$ for test sentences ≤ 10 and 8.3 for all. †Used many random initializers with unsupervised run selection. ‡Used staged training with sentences ≤ 45 words.

Procedure We run EM for our concave models for 100 iterations. We evaluate the learned models directly as parsers on the test data and also use them to initialize the DMV. When using them directly as parsers, we use dynamic programming to ensure that a valid tree is recovered. When using the concave models as initializers for the DMV, we copy the c parameters over directly since they appear in both models. We do not have the stop/continue parameters in our concave models, so we simply initialize them uniformly for the DMV. We train each DMV for 200 iterations and use minimum Bayes risk decoding with the final model on the test data. We use several initializers for training the DMV, including the uniform initializer (Unif), K&M, and our trained concave models Ccv1 and Ccv2.

[2] This is similar to the rule used by Mareček and Žabokrtský (2011) with empirical success.

[3] As verbs, we take all tags that map to V in the universal tag mappings from Petrov et al. (2012). Thus, to apply this constraint to a new language, one would have to produce a similar tag mapping or identify verb tags through manual inspection.

Init.	eu	bg	ca	zh	cs	da	nl	en	de	el	hu
UNIF	24/21	32/26	27/29	44/40	32/30	24/19	21/21	21/18	31/24	37/32	23/18
K&M	**32/26**	**48/40**	24/25	38/33	31/29	**34/23**	39/33	44/33	47/**37**	50/41	23/20
CCV1	22/21	34/27	**44/51**	**46/45**	33/31	19/14	24/24	45/31	46/31	**51/45**	32/28
CCV2	26/25	34/26	29/35	**46**/44	**50/40**	29/18	**50/43**	**54/43**	**49**/33	50/**45**	**60/46**
	it	ja	pt	sl	es	sv	tr	avg. accuracy		avg. log-likelihood	
UNIF	31/24	35/30	49/36	20/20	29/24	26/22	33/30	29.8 / 25.7		-15.05	
K&M	32/24	39/**31**	44/28	**33/27**	19/11	**46/33**	**39/36**	36.7 / 29.4		-14.84	
CCV1	34/25	42/27	**50/38**	30/25	41/33	45/**33**	37/29	37.5 / 30.9		-14.93	
CCV2	**55/48**	**49/31**	**50/38**	22/21	**57/50**	**46**/32	31/22	**43.7 / 35.5**		**-14.45**	

Table 2: Test set attachment accuracies for 18 languages; first number in each cell is accuracy for sentences ≤ 10 words and second is for all sentences. For training, sentences ≤ 10 words from each treebank were used. In order, languages are Basque, Bulgarian, Catalan, Chinese, Czech, Danish, Dutch, English, German, Greek, Hungarian, Italian, Japanese, Portuguese, Slovenian, Spanish, Swedish, and Turkish.

Data We use data prepared for the CoNLL 2006/07 shared tasks (Buchholz and Marsi, 2006; Nivre et al., 2007).[4] We follow standard practice in removing punctuation and using short sentences (≤ 10 or ≤ 20 words) for training. For all experiments, we train on separate data from that used for testing and use gold POS tags for both training and testing. We report accuracy on (i) test set sentences ≤ 10 words and (ii) all sentences from the test set.

Results Results for English are shown in Tab. 1. We train on §2–21 and test on §23 in the Penn Treebank. The constraint on sentence roots helps a great deal, as CCV2 by itself is competitive with the DMV when testing on short sentences. The true benefit of the concave models, however, appears when using them as initializers. The DMV initialized with CCV2 achieves a substantial improvement over all others. When training on sentences of length ≤ 20 words (bold), the performance even rivals that of several more sophisticated models shown in the table, despite only using the DMV with a different initializer.

Tab. 2 shows results for 18 languages. On average, CCV2 performs best and CCV1 does at least as well as K&M. This shows that a simple, concave model can be as effective as a state-of-the-art hand-designed initializer (K&M), and that concave models can encode linguistic knowledge to further improve performance.

Average log-likelihoods (micro-averaged across sentences) achieved by EM training are shown in the final column of Tab. 2. CCV2 leads to substantially-higher likelihoods than the other initializers, suggesting that the verb-root constraint is helping EM to find better local optima.[5]

5 Discussion

Staged training has been shown to help unsupervised learning in the past, from early work in grammar induction (Lari and Young, 1990) and word alignment (Brown et al., 1993) to more recent work in dependency grammar induction (Spitkovsky et al., 2010). While we do not yet offer a generic procedure for extracting a concave approximation from any model for unsupervised learning, our results contribute evidence in favor of the general methodology of staged training in unsupervised learning, and provide a simple and powerful initialization method for dependency grammar induction.

Acknowledgments

We thank Shay Cohen, Dipanjan Das, Val Spitkovsky, and members of the ARK research group for helpful comments that improved this paper. This research was supported in part by the NSF through grant IIS-0915187, the U. S. Army Research Laboratory and the U. S. Army Research Office under contract/grant number W911NF-10-1-0533, and Sandia National Laboratories (fellowship to K. Gimpel).

[4] In some cases, we did not use official CoNLL test sets but instead took the training data and reserved the first 80% of the sentences for training, the next 10% for development, and the final 10% as our test set; dataset details are omitted for space but are the same as those given by Cohen (2011).

[5] However, while CCV1 leads to a higher average accuracy than K&M, the latter reaches slightly higher likelihood, suggesting that the success of the concave initializers is only partially due to reaching high training likelihood.

References

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

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

S. Brody. 2010. It depends on the translation: Unsupervised dependency parsing via word alignment. In *Proc. of EMNLP*.

P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

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

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

S. Cohen. 2011. *Computational Learning of Probabilistic Grammars in the Unsupervised Setting*. Ph.D. thesis, Carnegie Mellon University.

J. Gillenwater, K. Ganchev, J. Graça, F. Pereira, , and B. Taskar. 2010. Posterior sparsity in unsupervised dependency parsing. *Journal of Machine Learning Research*.

Y. Goldberg, M. Adler, and M. Elhadad. 2008. EM can find pretty good HMM POS-taggers (when given a good start). In *Proc. of ACL*.

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

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

K. Lari and S. J. Young. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56.

D. Mareček and Z. Žabokrtský. 2011. Gibbs sampling with treeness constraint in unsupervised dependency parsing. In *Proc. of Workshop on Robust Unsupervised and Semisupervised Methods in Natural Language Processing*.

T. Naseem, H. Chen, R. Barzilay, and M. Johnson. 2010. Using universal linguistic knowledge to guide grammar induction. In *Proc. of EMNLP*.

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 *Proc. of CoNLL*.

S. Petrov, D. Das, and R. McDonald. 2012. A universal part-of-speech tagset. In *Proc. of LREC*.

V. I. Spitkovsky, H. Alshawi, and D. Jurafsky. 2010. From Baby Steps to Leapfrog: How “Less is More” in unsupervised dependency parsing. In *Proc. of NAACL-HLT*.

V. I. Spitkovsky, H. Alshawi, A. X. Chang, and D. Jurafsky. 2011a. Unsupervised dependency parsing without gold part-of-speech tags. In *Proc. of EMNLP*.

V. I. Spitkovsky, H. Alshawi, and D. Jurafsky. 2011b. Punctuation: Making a point in unsupervised dependency parsing. In *Proc. of CoNLL*.

K. Toutanova and M. Galley. 2011. Why initialization matters for IBM Model 1: Multiple optima and non-strict convexity. In *Proc. of ACL*.