Improved Iterative Scaling can yield multiple globally optimal models with radically differing performance levels

Iain Bancarz and Miles Osborne
{iainrb,osborne}@cogsci.ed.ac.uk
Division of Informatics
University of Edinburgh
2 Buccleuch Place
Edinburgh EH8 9LW
Scotland

Abstract

Log-linear models can be efficiently estimated using algorithms such as Improved Iterative Scaling (IIS)(Lafferty et al., 1997). Under certain conditions and for a particular class of problems, IIS is guaranteed to approach both the maximum-likelihood and maximum entropy solution. This solution, in likelihood space, is unique. Unfortunately, in realistic situations, multiple solutions may exist, all of which are equivalent to each other in terms of likelihood, but radically different from each other in terms of performance. We show that this behaviour can occur when a model contains *overlapping features* and the training material is sparse. Experimental results, from the domain of parse selection for stochastic attribute value grammars, shows the wide variation in performance that can be found when estimating models using IIS. Further results show that the influence of the initial model can be diminished by selecting either uniform weights, or else by model averaging.

1 Background

When statistically modelling linguistic phenomena of one sort or another, researchers typically fit log-linear models to the data (for example (Johnson et al., 1999)). There are (at least) three reasons for the popularity of such models: they do not make unwarranted independence assumptions, the maximum likelihood solution of such models coincides with the maximum entropy solution, and finally, they can be efficiently estimated (using algorithms such as Improved Iterative Scaling (IIS) (Lafferty et al., 1997)).

Now, the solution found by IIS is guaranteed to approach a *global* maximum for both likelihood and entropy under certain conditions. Although this is appealing, in realistic situations it turns out that multiple models exist, all of which are equivalent in terms of likelihood but different from each other in terms of their performance at some task. In particular, the *initial* weight settings can influence the quality of the final model, even though this final model is the maximum entropy solution (as found by IIS).

At first glance, this seems very strange. The IIS algorithm is guaranteed to converge to a globally optimal solution regardless of the initial parameters. If the initial weights assigned to some of the features are wildly inappropriate then the algorithm may take longer to converge, but one would expect the final destination to remain the same. However, as we show later, what is unique in terms of likelihood need not be unique in terms of performance, and so IIS can be sensitive to the initial weight settings.

Some of the reason for this behaviour *may* lie in a relatively subtle effect which we call *overlapping features*.[1] If some features behave identically in the training set (but not necessarily in future samples), IIS cannot distinguish between different sets of weights for those features unless the *sum* of all weights in each set is different. In fact, the final weights assigned to such features will be dependent on their initial values. Under these conditions, there will be a family of models, all of which are identical as far as IIS is concerned, but distinguishable from each other in terms of performance. This means that in terms of performance space, the landscape will contain local maxima. In terms of likelihood space, the landscape will continue to contain just the single (global) maximum.

This indeterminacy is clearly undesirable. However, there are (at least) two practical ways of dealing with it: one either initialises IIS with weights that are identical (0 is a good choice), or else one takes a Bayesian approach, and averages over the space of models. In this paper, we show that the performance of IIS is sensitive to the initial weight settings. We show that setting the weights to zero yields performance that is better than a large set of randomly initialised models. Also, we show that model averaging can also reduce indeterminacies introduced through the choice of initial weights.

The rest of this paper is as follows. Section 2 is a restatement of the theory behind IIS. Section 3 shows how sparse training material, coupled

[1] We do not claim that overlapping features are the *sole* reason for our observed effects. As an anonymous reviewer noted, sparse statistics may also yield similar findings.

with large models can result in situations whereby IIS produces suboptimal results. We then move on to experimental support for our theoretical results. Our domain is parse selection for broad-coverage stochastic attribute-value grammars. Section 4 shows how we model the broad coverage attribute-value grammar (mentioned in section 5 used in our experiments), and then present our results. The first set of experiments (section 7) deals with how well IIS, with uniform initial settings. outperforms models with randomised initial settings. The second set of experiments shows how model averging can deal with the problem of initialising the model. We conclude the paper with some comments on our work.

2 The Duality Lemma

How can we be certain that IIS seeks out a global maximum for both likelihood and entropy? The answer is that for the class of problems under consideration, there is only a single maximum - which is then necessarily the global one. The IIS algorithm simply 'hill-climbs' and seeks to increase the likelihood of the model being trained. Its success is guaranteed because of a result which we refer to as the Duality Lemma[2].

The proof of the Duality Lemma is contained in (Lafferty et al., 1997) but will be omitted here. In order to state the lemma, we first define the setting and establish some notation.

Suppose that we have a probability measure space $(\Omega, \mathcal{F}, \mathcal{P})$, where as usual Ω is a set made up of elements ω, $\mathcal{F}$ is a sigma-field on Ω, and P is a probability measure on $\mathcal{F}$. Suppose further that X is a simple random variable on Ω - that is to say, it is a real-valued function on Ω, having finite range, and such that $[\omega : X(\omega) = x] \in \mathcal{F}$. As usual, we will omit the argument ω, so that X indicates a general value of the function as well as the function itself, and x denotes $[\omega : X(\omega) = x]$. We will also abuse notation by letting X denote the set of possible values of x; the meaning should be clear from context.

We now consider a stochastic process on $(\Omega, \mathcal{F}, P)$. We are given a sample of past outputs (data points) from this process which make up the training data. We again abuse notation by using $\tilde{p}$ to refer to both the set of data points and the distribution defined by that set. Let Δ denote the set of all possible probability distributions over X; we seek a model $q_* \in \Delta$ which is in some sense the best possible probability distribution over future outputs. We specifically examine generalized Gibbs distributions, which are of the form:

$$q_h(x) = \frac{1}{(Z_q(h))} \exp h(x) q(x) \tag{1}$$

[2] This result was called Proposition 4 (Lafferty et al., 1997).

In this case, h is a real-valued function on X, q is an initial probability distribution over X (which may be the uniform distribution), and $Z_q(h)$ is a normalising constant, taking a value such that $\sum_{x \in X} q_h(x) = 1$.

The function h here takes the form:

$$h(x) = \sum_{i=1}^{n} \lambda_i f_i(x) = (\lambda \cdot f)(x) \tag{2}$$

where the $f_i(x)$ are integer feature functions. The real numbers λ_i are adjustable parameters.

Suppose that we are given the data $\tilde{p}$, an initial model q_0, and a set of features f. Lafferty et al. (1997) describe two natural sets of models. The first is the set $\mathcal{P}(f, \tilde{p})$ of all distributions that agree with $\tilde{p}$ as to the expected value of the feature function f:

$$\mathcal{P}(f, \tilde{p}) = [p \in \Delta : p[f] = \tilde{p}[f]] \tag{3}$$

where, as usual, $p[f]$ denotes the expectation of the function f under the distribution p.

The second is $\mathcal{Q}(f, q_0)$, the set of generalized Gibbs distributions based on q_0 and with feature set f:

$$\mathcal{Q}(f, q_0) = [(\lambda \cdot f) \circ q_0] \tag{4}$$

Let $\overline{\mathcal{Q}}$ denote the closure of $\mathcal{Q}$ in Δ, with respect to the topology Δ inherits as a subset of Euclidean space.

These in turn determine two natural candidates for the 'best' model q_*. Let $D(p\|q)$ denote the Kullback-Leibler divergence between two distributions p and q. The suitable models are:

- *Maximum Likelihood Gibbs Distribution.* A distribution in $\overline{\mathcal{Q}}$ with maximum likelihood with respect to $\tilde{p}$: $q_*^{ML} = \text{argmin}_{q \in \overline{\mathcal{Q}}} D(\tilde{p}\|q)$.
- *Maximum Entropy Constrained Distribution.* A distribution in $\mathcal{P}$ with maximum entropy relative to q_0: $q_*^{ME} = \text{argmin}_{q \in \mathcal{P}(f,\tilde{p})} D(p\|q_0)$

The key result of Lafferty et al. (1997) is that there is a unique q_* satisfying $q_* = q_*^{ML} = q_*^{ME}$. In Appendix 1 of that paper, the following result is proved:

The Duality Lemma. Suppose that $D(\tilde{p}\|q_0) < \infty$. Then there exists a unique $q_* \in \Delta$ satisfying:

1. $q_* \in \mathcal{P} \cap \overline{\mathcal{Q}}$
2. $D(p\|q) = D(p\|q_*) + D(q_*\|q)$ $\forall p \in \mathcal{P}, q \in \overline{\mathcal{Q}}$
3. $q_* = \text{argmin}_{q \in \overline{\mathcal{Q}}} D(\tilde{p}\|q)$
4. $q^* = \text{argmin}_{q \in \mathcal{P}(f,\tilde{p})} D(p\|q_0)$

The Duality Lemma is a very useful result, fundamental to the ability of IIS to converge upon the single maximum likelihood, maximum entropy solution. The IIS algorithm itself uses a fairly straightforward technique to look for any maximum of the likelihood function; because of the lemma, IIS is guaranteed to approach the solution q_*.

3 Limitations With Sparse Training Data

3.1 Overlapping Features

In this paper, all models have the same training data and a uniform initial distribution q_0. This ensures that $D(p||q_0) \leq \infty$ and that, for all models, the IIS algorithm approaches the same optimal distribution q_*. However, our experiments show that not all distributions obtained by running IIS (to convergence) are equally good at modelling a set of test data. Performance appears to depend (at least) on the starting values for the weights λ_i. This may be a result of the following situation:

Consider two features, f_i and f_j with $i < j$, with weights λ_i and λ_j respectively. Suppose that $f_i(x) = f_j(x)$ for all values of x in $\tilde{p}$, but there exist values of x outside $\tilde{p}$ such that $f_i(x) \neq f_j(x)$; that is, the two functions take exactly the same values on the set of training data but differ outside of it. This phenomenon can be called *overlapping.*[3] Overlapping features are commonly found in maximum entropy models, and are one of the main reasons for their popularity. In fact, one could argue that *all* maximum entropy models found in natural language applications contain overlapping features. Overlapping features may be present by explicit design (for example when emulating backing-off smoothing), or else naturally, for example when using features to treat words co-occurring with each other.

We assign initial weights $\lambda_i^{(0)}$ and $\lambda_j^{(0)}$ to the features, and after n iterations of the algorithm, they have been adjusted to $\lambda_i^{(n)}$ and $\lambda_j^{(n)}$ respectively.

Now consider the target solution q_*, as determined by q_0 and p. Let us assume for convenience that q_* is of the form:

$$q_* = \frac{1}{Z^*} \exp\left(\sum_{k=1}^{n} \lambda_k^* f_k\right) \tag{5}$$

where Z^* is the usual normalising constant. Notice that q_* may not belong to the family of exponential models $\mathcal{Q}$, but instead may be part of the larger set $\overline{\mathcal{Q}}$. Indeed, della Pietra et al. state that $\mathcal{P} \cap \mathcal{Q}$ may be empty. This is not a serious limitation as one can come arbitrarily close to any element of $\overline{\mathcal{Q}}$ while remaining inside $\mathcal{Q}$. Thus, if $q_* \notin \mathcal{Q}$, we may consider the above expression to be a very close approximation to q_*, such as could be obtained by running IIS to convergence.

Because the ith and jth features are equal on the training data, the exact values of λ_i^* and λ_j^* have no effect on the likelihood of the model as long as their sum remains the same. In this instance, q_* is not a single model at all, but rather a family of models satisfying the condition $\lambda_i^* + \lambda_j^* = \lambda_{ij}^*$ for a particular λ_{ij}^*.[4] All models in this family assign equal likelihood to the training data, and so the IIS algorithm is unable to distinguish between them.

Under these circumstances we should ensure that $\lambda_i^* = \lambda_j^*$. Since we have no way to distinguish between the two features, our model should assign them equal importance. However, this is not guaranteed by the IIS algorithm. In particular, it is less likely to occur if the initial weights $\lambda_i^{(0)}$ and $\lambda_j^{(0)}$ are not equal.

3.2 IIS and Overlapping Features - A Simple Example

Suppose that our model has a vector of parameters $\lambda = [\lambda_i : i = 1 \ldots n]$ and we wish to change it to $\lambda + \delta$, where $\delta = [\delta_i : i = 1 \ldots n]$ The IIS algorithm considers each λ_i in turn. It chooses δ_i to maximize an auxiliary function $B(\delta|\lambda)$, which provides a lower bound on the change in log-likelihood of the model. Each adjustment $\lambda_i \leftarrow \lambda_i + \delta_i$ is guaranteed to increase the likelihood of the model and thus approach our ideal solution q_*. This process is repeated until convergence. There are no inherent restrictions on the value of δ_i; it may be positive or negative, and large or small compared to the values taken for different features.

Now, suppose that the features f_i and f_j overlap and we halt the algorithm after t iterations. Clearly, the algorithm cannot guarantee that the final weights λ_i^t and λ_j^t will be equal. The following highly simplified example should demonstrate why this is the case.

Example 1: Imagine that our model has two overlapping features f_1 and f_2, and q_* is the family of models in which $\lambda_1 + \lambda_2 = 5$. Suppose that the initial weights are $\lambda_1^{(0)} = 5, \lambda_2^0 = 0$. Recall that at the ith step of an iteration, the IIS algorithm considers the change in log-likelihood of the model which can be made by only adjusting the ith parameter. In this case no change is possible as the sum $\lambda_1 + \lambda_2$ is already at its optimum value, so the algorithm terminates with $\lambda_1^t = 5$ and $\lambda_2^t = 0$. We have assigned far greater importance to f_1 than to f_2 with no justification for doing so. Clearly, this assumption might not be warranted.

3.3 IIS and Overlapping Features in Practice

The situation will obviously be much more complicated in practice. Most importantly, there is no such

[3] We can relax this definition of overlapping and allow features to largely co-occur together. Depending upon the degree of co-occurrence, we would expect to continue to find our results.

[4] The number λ_{ij} is not a 'constant' as such, since any vector of weights is in a sense unique only up to multiplication by a positive constant. This will be examined in greater detail when we consider overlapping features in practice.

thing as an absolute "optimum" vector of weights λ_*. As a result, no one parameter can be regarded as fixed until the algorithm has converged for all parameters. In the above example, our "true" set of target solutions is those for which $\lambda_1 + \lambda_2$ is a constant. Since there are no other features, IIS would in fact terminate at once for *any* pair of initial weights.

Suppose that we added a third feature f_3 which did not overlap the first two. Our set of target solutions would then be of the form $\lambda_1 + \lambda_2 = \lambda^*_{12}k, \lambda_3 = \lambda^*_3 k$ for some fixed λ_{12} and λ_3 and for any $k > 0$. The adjustments made to λ_1 and λ_2 will depend on those made to λ_3, and the algorithm will not necessarily terminate after the first pass.

The following example will describe what happens in this more realistic situation. It will also explain the possible benefits of setting all initial weights to the same value.

Example 2: Let the model have the three features described above, with initial weights $\lambda^0_1 = 5, \lambda^0_2 = 0, \lambda^0_3 = 1$ and a family of target solutions q_* defined by $\lambda_1 + \lambda_2 = 5k, \lambda_3 = k$ for any $k > 0$. IIS again terminates at once, because the initial model is already part of the family q_*. Again there is an unjustified difference between the final weights for f_1 and f_2.

Now suppose that all three initial weights are set to zero. Imagine for simplicity that we have restricted the algorithm to adjust weights only by zero or ± 1. On the first pass, all three weights are changed to 1. On the second, λ_1 and λ_2 are increased to 2; λ_3 remains at 1. In the third iteration, λ_1 is set to 3, λ_2 remains at 2 and λ_3 at 1, and the algorithm terminates.[5]

Notice that, although there is still a difference between the final weights for f_1 and f_2, it is much less than before. This more closely approaches the ideal situation in which the weights are equal.

This concludes our theoretical treatment of IIS. We now show experimentally the influence that the initial weight settings have, and how it can be minimised. Our strategy is to use plausible features, as found in a realistic domain (parse selection for stochastic attribute-value grammars), and firstly show what happens when the initial weight settings are set uniformly (to zero). We then show what happens when these initial settings are randomly set, and finally, what happens when we average over randomly initialised maximum likelihood solutions.

[5] The exact behaviour of the algorithm will depend on the training data, so this is not the only imaginable outcome, but it is certainly a plausible one.

4 Log-linear Modelling of Attribute-Value Grammars

Here we show how attribute-value grammars may be modelled using log-linear models. Abney gives fuller details (Abney, 1997).

Let G be an attribute-value grammar, and D a set of sentences within the string-set defined by $L(G)$. A log-linear model, M, consist of two components: a set of *features*, F and a set of *weights*, Λ.

The (unnormalised) total weight of a parse x, $\psi(x)$, is a function of the k features that are 'active' on a parse:

$$\psi(x) = \exp(\sum_{i=1}^{k} \lambda_i f_i(x)) \tag{6}$$

The probability of a parse, $P(x \mid M)$, is simply the result of normalising the total weight associated with that parse:

$$P(x \mid M) = \frac{1}{Z}\psi(x) \tag{7}$$

$$Z = \sum_{y \in \Omega} \psi(y) \tag{8}$$

Ω is the union of the set of parses assigned to each sentence in D by the grammar G, such that each parse in Ω is unique in terms of the features that are active on it. Normally a parse can be viewed as the set of features that are active on it.

The interpretation of this probability (equation 7) depends upon the application of the model. Here, we use parse probabilities to reflect preferences for parses.

5 The Grammar

The grammar we model with log-linear models (called the *Tag Sequence Grammar* (Briscoe and Carroll, 1996), or TSG for short) was manually developed with regard to coverage, and when compiled consists of 455 Definite Clause Grammar (DCG) rules. It does not parse sequences of words directly, but instead assigns derivations to sequences of part-of-speech tags (using the CLAWS2 tagset). The grammar is relatively shallow (for example, it does not fully analyse unbounded dependencies).

6 Modelling the Grammar

Modelling the TSG with respect to the parsed Wall Street Journal consists of two steps: creation of a feature set and definition of a reference distribution (the target model, $\tilde{p}$).

6.1 Feature Set

Modelling the TSG with respect to the parsed Wall Street Journal consists of two steps: creation of a

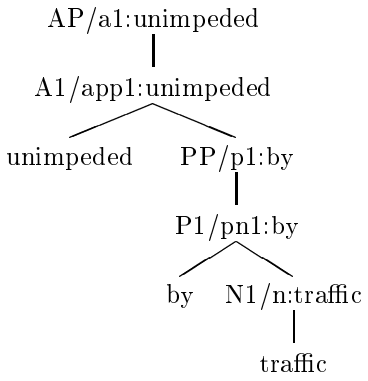


Figure 1: TSG Parse Fragment

feature set and definition of the reference distribution.

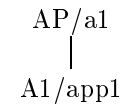
Our feature set is created by parsing sentences in the training set, and using each parse to instantiate *templates*. Each template defines a family of features. Our templates are motivated by the observations that linguistically-stipulated units (DCG rules) are informative, and that many DCG applications in preferred parses can be predicted using lexical information.

The first template creates features that count the number of times a DCG instantiationis present within a parse.[6] For example, suppose we parsed the Wall Street Journal AP:

1 unimpeded by traffic

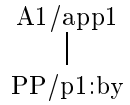
A parse tree generated by TSG might be as shown in figure 1. Here, to save on space, we have labelled each interior node in the parse tree with TSG rule names, and not attribute-value bundles. Furthermore, we have annotated each node with the head word of the phrase in question. Within our grammar, heads are (usually) explicitly marked. This means we do not have to make any guesses when identifying the head of a local tree. With head information, we are able to lexicalise models. We have suppressed tagging information.

For example, a feature defined using this template might count the number of times the we saw:



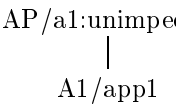
in a parse. Such features record some of the context of the rule application, in that rule applications that differ in terms of how attributes are bound will be modelled by different features.

Our second template creates features that are partially lexicalised. For each local tree (of depth one) that has a PP daughter, we create a feature that counts the number of times that local tree, decorated with the head-word of the PP, was seen in a parse. An example of such a lexicalised feature would be:



These features are designed to model PP attachments that can be resolved using the head of the PP.

The third and final template creates features that are again partially lexicalised. This time, we create local trees of depth one that are decorated with the head word. For example, here is one such feature:



Note the second and third templates result in features that overlap with features resulting from applications of the first template.

6.2 Reference Distribution

We create the reference distribution R (an association of probabilities with TSG parses of sentences, such that the probabilities reflect parse preferences) using the following process:

1. Take the training set of parses and for each parse, compare the structural differences between it and a reference treebank parse.
2. Map these tree similarity scores into probabilities (where the sum of all reference probabilities for all parses sums to one).

Again, see Anon for more details.

This concludes our discussion of how we model grammars. We now go on to present our experimental investigation of the influence of the initial weight settings.

7 Experiments

Here we present three sets of experiments. The first set shows the performance of maximum entropy when the initial weight setting are zero. The second set show the effects of randomised initial setting, and so establishes (an estimate of) the variation in performance space. The third set of experiments showed how the influence of the initial weight settings could be minimised by averaging over manyh models.

[6] Note, all our features suppress any terminals that appear in a local tree. Lexical information is included when we decide to lexicalise features.

Throughout, we used the same training set. This consisted of a sample of 53795 parses (produced from sentences at most 15 tokens long, with at most 15 parses per sentence). The sentences were drawn from the parsed Wall Street Journal, and all could be parsed using our grammar. The motivation for this choice of training set came from the fact that when the sample of sentences is too small, the resulting model will tend to underfit. Likewise, when the training set is too large, the model will tend to overfit. A sample of appropriate size (which can be found using a simple search, as Osborne (2000) demonstrated) will therefore neither significantly underfit nor overfit. Quite apart from estimation issues related to sample size, because we repeatedly estimate models, using a sample that is just sufficiently large (and no larger) allows us to make significant computational savings.

We used a disjoint development set and testing set. The development set consisted of 2620 parses, derived from parsing sentences at most 30 tokens long, with at most 100 parses per sentence. The testing set was randomly sampled from the Wall Street Journal, and consisted of 469 sentences, with each sentence at most 30 tokens long, with at most 100 parses per sentence. Each sentence has on average 60.0 parses per sentence.

The model we used throughout contained 75171 features.

For all experiments, we ran IIS for the same number of iterations (75). This number was selected as the number of iterations that produced the best performance for maximum entropy (our yardstick).

Evaluation was in terms of exact match: for each sentence in the test set, we awarded ourselves a point if the estimated model ranked highest the same parse that was ranked highest using the reference probabilities.[7] Ties were randomly broken.

7.1 Maximum Entropy Results using Uniform Initialisation

A model trained using weights all initialised to zero yielded an exact match score of 52.0%.

7.2 Randomised Models

A pool of 10,000 models was created by randomly setting the initial weights to values in the range [-0.3,0.3] and then estimating the final weights using the training set.

The histogram in figure 2 shows the number of models that produced the same results on the testing material. As can be seen, performance is roughly normally distributed, with some local minima having a much wider basin of attraction than other minima. Also, note that all of these models underperforms a model crated using uniform initialisation.[8] The performance of the worst model was 24.1%. This is our lower bound, and is less than half that of basic maxent. The best randomly selected model had a performance of 46.5%.

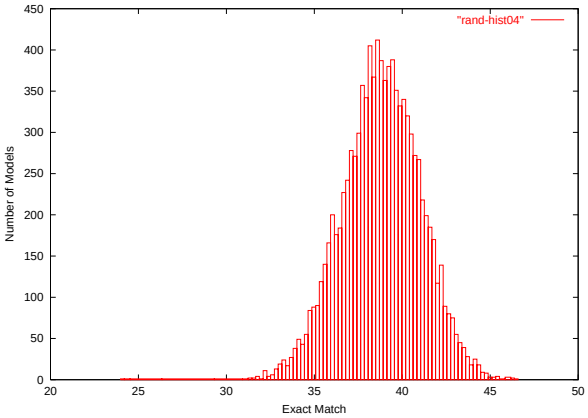


Figure 2: Distribution of models with randomly initialised starting conditions

7.3 Averaging over models

To see whether an ensemble of such randomised models could cancel-out the influence of the initial weight settings, we created a pool of 600 randomised models, and then combined then together using equation 9:

$$P(x \mid M_1 \ldots M_n) = \frac{\prod_{i=1}^{n} P(x \mid M_i)}{\sum_y \prod_{j=1}^{n} P(y \mid M_j)} \qquad (9)$$

Because it is possible that some subset of the models outperforms an ensemble using all models, we uniformly sampled, with replacement, from this pool models for inclusion into the final ensemble. Random selection introduces variation, so we repeated our results ten time and averaged the results.

Figure 3 shows our results (N is the number of models in each ensemble, $\overline{x}$ is the mean performance and σ^2 is the standard deviation). The final entry (marked *all*) shows the performance obtained using an ensemble consisting of all models in the model, equally weighted.

As can be seen, increasing the number of models in the ensemble reduces the influence of the initial weight settings. However, even with a large pool of models, we still marginally underperform uniform initialisation.

[7] We use exact match as an arbitrary evaluation metric. The particular choice of metric is not crucial to the results of this paper.

[8] Running IIS until convergence did not narrow the gap, so our findings cannot be attributed to differential rates of convergence.

N	$\overline{x}$	σ^2	N	$\overline{x}$	σ^2
1	38.78	0.12	50	49.94	1.24
2	41.41	1.04	100	50.55	0.70
3	44.24	1.63	150	50.62	0.88
5	46.57	1.69	200	50.66	0.75
10	47.21	1.71	300	50.81	0.73
20	49.23	1.19	600	51.04	0.36
all	51.39	-			

Figure 3: Model averaging results

Note that we have not carried out the control experiment of repeating our runs using features which do not overlap with each other. Unfortunately, doing this would probably mean having to create models that are unnatural. We therefore cannot be absolutely certain that the sole reason for the existance of local optima is the presence of overlapping features. However, we can be sure that they do exist, and that varying the initial weight settings will reveal them.

8 Comments

We have established that, in the presence of overlapping features, the values of the initial weights can affect the utility of the final model. Our experimental results support this finding. In general, one might encounter a similar problem (overlapping features) with what might be called 'semi-overlapping features': features which are very similar (but not identical) on the training data and very different outside of it.

IIS could be made more robust to the choice of initial parameters in a number of ways:

- The simplest course of action is to set all initial weights to the same value. Although zero is often a convenient initial value, in principle any real number would do, since the IIS algorithm can reach a given optimal solution from any starting point in the space of initial parameters.
- We could also examine all the features to determine which ones overlap, and force it to balance the final weights of these features. For very large models, this may be prohibitively difficult and time-consuming.
- Model averaging can also cancel out variations caused by a particular choice of initial settings. However, this implies a greater computational burden as IIS will need to be run many times in order to gain a representative sample of models.
- The number of features in the model could be reduced using feature selection methods (for example (Mullen and Osborne, 2000)).

Although IIS is a useful tool for estimating log-linear models, we have since moved-on to estimating models using limited-memory variable-metric methods (Malouf, 2002). Our findings show that convergence, for a range of problems, is faster. An interesting question is seeing the extent to which other numerical methods for estimating log-linear models are sensitive to initial parameter values. Finally, it should be noted that our theoretical results apply to a more general setting than that of log-linear models trained using the IIS algorithm. The problem of overlapping features could in principle occur in any situation in which a model has a linear combination of features, and a 'hill-climbing' algorithm is used to seek a maximum-likelihood solution.

Acknowledgements

We wish to thank Steve Clark for useful discussions about IIS, Rob Malouf for supplying the IIS implementation and the two anonymous reviewers. Iain Bancarz was supported by the EPSRC grant *POEM*.

References

Steven P. Abney. 1997. Stochastic Attribute-Value Grammars. *Computational Linguistics*, 23(4):597–618, December.

Ted Briscoe and John Carroll. 1996. Automatic Extraction of Subcategorization from Corpora. In *Proceedings of the* 5^{th} *Conference on Applied NLP*, pages 356–363, Washington, DC.

Mark Johnson, Stuart Geman, Stephen Cannon, Zhiyi Chi, and Stephan Riezler. 1999. Estimators for Stochastic "Unification-Based" Grammars. In 37^{th} *Annual Meeting of the ACL*.

J. Lafferty, S. Della Pietra, and V. Della Pietra. 1997. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393, April.

Robert Malouf. 2002. A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the joint CoNLL-WVLC Meeting*, Taipei, Taiwan. ACL. To appear.

Tony Mullen and Miles Osborne. 2000. Overfitting Avoidance for Stochastic Modeling of Attribute-Value Grammars. In Claire Cardie, Walter Daelemans, Claire Nedellec, and Erik Tjong Kim Sang, editors, *Proceedings of the Computational Natural Language learning 2000*, pages 49–54. ACL, Lisbon, Portugal.

Kamal Nigam, John Lafferty, , and Andrew McCallum. 1999. Using maximum entropy for text classification. In *IJCAI-99 Workshop on Machine Learning for Information Filtering,*.

Miles Osborne. 2000. Estimation of Stochastic Attribute-Value Grammars using an Informative Sample. In *The* 18^{th} *International Conference on Computational Linguistics*, Saarbrücken, August.