Painless Unsupervised Learning with Features

Taylor Berg-Kirkpatrick **Alexandre Bouchard-Côté** **John DeNero** **Dan Klein**
Computer Science Division
University of California at Berkeley
{tberg, bouchard, denero, klein}@cs.berkeley.edu

Abstract

We show how features can easily be added to standard generative models for unsupervised learning, without requiring complex new training methods. In particular, each component multinomial of a generative model can be turned into a miniature logistic regression model if feature locality permits. The intuitive EM algorithm still applies, but with a gradient-based M-step familiar from discriminative training of logistic regression models. We apply this technique to part-of-speech induction, grammar induction, word alignment, and word segmentation, incorporating a few linguistically-motivated features into the standard generative model for each task. These feature-enhanced models each outperform their basic counterparts by a substantial margin, and even compete with and surpass more complex state-of-the-art models.

1 Introduction

Unsupervised learning methods have been increasingly successful in recent NLP research. The reasons are varied: increased supplies of unlabeled data, improved understanding of modeling methods, additional choices of optimization algorithms, and, perhaps most importantly for the present work, incorporation of richer domain knowledge into structured models. Unfortunately, that knowledge has generally been encoded in the form of conditional independence structure, which means that injecting it is both tricky (because the connection between independence and knowledge is subtle) and time-consuming (because new structure often necessitates new inference algorithms).

In this paper, we present a range of experiments wherein we improve existing unsupervised models by declaratively adding richer features. In particular, we parameterize the local multinomials of existing generative models using features, in a way which does not require complex new machinery but which still provides substantial flexibility. In the feature-engineering paradigm, one can worry less about the backbone structure and instead use hand-designed features to declaratively inject domain knowledge into a model. While feature engineering has historically been associated with discriminative, supervised learning settings, we argue that it can and should be applied more broadly to the unsupervised setting.

The idea of using features in unsupervised learning is neither new nor even controversial. Many top unsupervised results use feature-based models (Smith and Eisner, 2005; Haghighi and Klein, 2006). However, such approaches have presented their own barriers, from challenging normalization problems, to neighborhood design, to the need for complex optimization procedures. As a result, most work still focuses on the stable and intuitive approach of using the EM algorithm to optimize data likelihood in locally normalized, generative models.

The primary contribution of this paper is to demonstrate the clear empirical success of a simple and accessible approach to unsupervised learning with features, which can be optimized by using standard NLP building blocks. We consider the same generative, locally-normalized models that dominate past work on a range of tasks. However, we follow Chen (2003), Bisani and Ney (2008), and Bouchard-Côté et al. (2008), and allow each component multinomial of the model to be a miniature multi-class logistic regression model. In this case, the EM algorithm still applies with the E-step unchanged. The M-step involves gradient-based training familiar from standard supervised logistic regression (i.e., maximum entropy models). By integrating these two familiar learning techniques, we add features to unsupervised models without any

specialized learning or inference.

A second contribution of this work is to show that further gains can be achieved by directly optimizing data likelihood with LBFGS (Liu et al., 1989). This alternative optimization procedure requires no additional machinery beyond what EM uses. This approach is still very simple to implement, and we found that it empirically outperforms EM.

This paper is largely empirical; the underlying optimization techniques are known, even if the overall approach will be novel to many readers. As an empirical demonstration, our results span an array of unsupervised learning tasks: part-of-speech induction, grammar induction, word alignment, and word segmentation. In each task, we show that declaring a few linguistically motivated feature templates yields state-of-the-art results.

2 Models

We start by explaining our feature-enhanced model for part-of-speech (POS) induction. This particular example illustrates our approach to adding features to unsupervised models in a well-known NLP task. We then explain how the technique applies more generally.

2.1 Example: Part-of-Speech Induction

POS induction consists of labeling words in text with POS tags. A hidden Markov model (HMM) is a standard model for this task, used in both a frequentist setting (Merialdo, 1994; Elworthy, 1994) and in a Bayesian setting (Goldwater and Griffiths, 2007; Johnson, 2007).

A POS HMM generates a sequence of words in order. In each generation step, an observed word emission y_i and a hidden successor POS tag z_{i+1} are generated independently, conditioned on the current POS tag z_i . This process continues until an absorbing stop state is generated by the transition model.

There are two types of conditional distributions in the model—emission and transition probabilities—that are both multinomial probability distributions. The joint likelihood factors into these distributions:

$$P_{\boldsymbol{\theta}}(\mathbf{Y} = \mathbf{y}, \mathbf{Z} = \mathbf{z}) = P_{\boldsymbol{\theta}}(Z_1 = z_1) \cdot \prod_{i=1}^{|\mathbf{z}|} P_{\boldsymbol{\theta}}(Y_i = y_i | Z_i = z_i) \cdot P_{\boldsymbol{\theta}}(Z_{i+1} = z_{i+1} | Z_i = z_i)$$

The emission distribution $P_{\boldsymbol{\theta}}(Y_i = y_i | Z_i = z_i)$ is parameterized by conditional probabilities $\theta_{y,z,\text{EMIT}}$ for each word y given tag z. Alternatively, we can express this emission distribution as the output of a logistic regression model, replacing the explicit conditional probability table by a logistic function parameterized by weights and features:

$$\theta_{y,z,\text{EMIT}}(\mathbf{w}) = \frac{\exp \langle \mathbf{w}, \mathbf{f}(y, z, \text{EMIT}) \rangle}{\sum_{y'} \exp \langle \mathbf{w}, \mathbf{f}(y', z, \text{EMIT}) \rangle}$$

This feature-based logistic expression is equivalent to the flat multinomial in the case that the feature function $\mathbf{f}(y, z, \text{EMIT})$ consists of all indicator features on tuples (y, z, EMIT), which we call BASIC features. The equivalence follows by setting weight $w_{y,z,\text{EMIT}} = \log(\theta_{y,z,\text{EMIT}})$.[1] This formulation is known as the natural parameterization of the multinomial distribution.

In order to enhance this emission distribution, we include coarse features in $\mathbf{f}(y, z, \text{EMIT})$, in addition to the BASIC features. Crucially, these features can be active across multiple (y, z) values. In this way, the model can abstract general patterns, such as a POS tag co-occurring with an inflectional morpheme. We discuss specific POS features in Section 4.

2.2 General Directed Models

Like the HMM, all of the models we propose are based on locally normalized generative decisions that condition on some context. In general, let $\mathbf{X} = (\mathbf{Z}, \mathbf{Y})$ denote the sequence of generation steps (random variables) where $\mathbf{Z}$ contains all hidden random variables and $\mathbf{Y}$ contains all observed random variables. The joint probability of this directed model factors as:

$$P_{\mathbf{w}}(\mathbf{X} = \mathbf{x}) = \prod_{i \in I} P_{\mathbf{w}} \left(X_i = x_i | X_{\pi(i)} = x_{\pi(i)} \right),$$

where $X_{\pi(i)}$ denotes the parents of X_i and I is the index set of the variables in $\mathbf{X}$.

In the models that we use, each factor in the above expression is the output of a local logistic regression

[1] As long as no transition or emission probabilities are equal to zero. When zeros are present, for instance to model that an absorbing stop state can only transition to itself, it is often possible to absorb these zeros into a *base measure*. All the arguments in this paper carry with a structured base measure; we drop it for simplicity.

model parameterized by $\mathbf{w}$:

$$P_{\mathbf{w}}\big(X_i = d \big| X_{\pi(i)} = c\big) = \frac{\exp\langle \mathbf{w}, \mathbf{f}(d, c, t)\rangle}{\sum_{d'} \exp\langle \mathbf{w}, \mathbf{f}(d', c, t)\rangle}$$

Above, d is the generative decision value for X_i picked by the model, c is the conditioning context tuple of values for the parents of X_i, and t is the type of decision being made. For instance, the POS HMM has two types of decisions: transitions and emissions. In the emission model, the type t is EMIT, the decision d is a word and the context c is a tag. The denominator normalizes the factor to be a probability distribution over decisions.

The objective function we derive from this model is the marginal likelihood of the observations $\mathbf{y}$, along with a regularization term:

$$L(\mathbf{w}) = \log P_{\mathbf{w}}(\mathbf{Y} = \mathbf{y}) - \kappa ||\mathbf{w}||_2^2 \qquad (1)$$

This model has two advantages over the more prevalent form of a feature-rich unsupervised model, the globally normalized Markov random field.[2] First, as we explain in Section 3, optimizing our objective does not require computing expectations over the joint distribution. In the case of the POS HMM, for example, we do not need to enumerate an infinite sum of products of potentials when optimizing, in contrast to Haghighi and Klein (2006). Second, we found that locally normalized models empirically outperform their globally normalized counterparts, despite their efficiency and simplicity.

3 Optimization

3.1 Optimizing with Expectation Maximization

In this section, we describe the EM algorithm applied to our feature-rich, locally normalized models. For models parameterized by standard multinomials, EM optimizes $L(\boldsymbol{\theta}) = \log P_{\boldsymbol{\theta}}(\mathbf{Y} = \mathbf{y})$ (Dempster et al., 1977). The E-step computes expected counts for each tuple of decision d, context c, and multinomial type t:

$$e_{d,c,t} \leftarrow \mathbb{E}_{\boldsymbol{\theta}}\left[\sum_{i \in I} \mathbb{1}(X_i = d, X_{\pi(i)} = c, t) \middle| \mathbf{Y} = \mathbf{y}\right] \qquad (2)$$

[2]The locally normalized model class is actually equivalent to its globally normalized counterpart when the former meets the following three conditions: (1) The graphical model is a directed tree. (2) The BASIC features are included in $\mathbf{f}$. (3) We do not include regularization in the model ($\kappa = 0$). This follows from Smith and Johnson (2007).

These expected counts are then normalized in the M-step to re-estimate $\boldsymbol{\theta}$:

$$\theta_{d,c,t} \leftarrow \frac{e_{d,c,t}}{\sum_{d'} e_{d',c,t}}$$

Normalizing expected counts in this way maximizes the expected complete log likelihood with respect to the current model parameters.

EM can likewise optimize $L(\mathbf{w})$ for our locally normalized models with logistic parameterizations. The E-step first precomputes multinomial parameters from $\mathbf{w}$ for each decision, context, and type:

$$\theta_{d,c,t}(\mathbf{w}) \leftarrow \frac{\exp\langle \mathbf{w}, \mathbf{f}(d, c, t)\rangle}{\sum_{d'} \exp\langle \mathbf{w}, \mathbf{f}(d', c, t)\rangle}$$

Then, expected counts $\mathbf{e}$ are computed according to Equation 2. In the case of POS induction, expected counts are computed with the forward-backward algorithm in both the standard and logistic parameterizations. The only change is that the conditional probabilities $\boldsymbol{\theta}$ are now functions of $\mathbf{w}$.

The M-step changes more substantially, but still relies on canonical NLP learning methods. We wish to choose $\mathbf{w}$ to optimize the regularized expected complete log likelihood:

$$\ell(\mathbf{w}, \mathbf{e}) = \sum_{d,c,t} e_{d,c,t} \log \theta_{d,c,t}(\mathbf{w}) - \kappa ||\mathbf{w}||_2^2 \qquad (3)$$

We optimize this objective via a gradient-based search algorithm like LBFGS. The gradient with respect to $\mathbf{w}$ takes the form

$$\nabla \ell(\mathbf{w}, \mathbf{e}) = \sum_{d,c,t} e_{d,c,t} \cdot \Delta_{d,c,t}(\mathbf{w}) - 2\kappa \cdot \mathbf{w} \qquad (4)$$

$$\Delta_{d,c,t}(\mathbf{w}) = \mathbf{f}(d, c, t) - \sum_{d'} \theta_{d',c,t}(\mathbf{w}) \mathbf{f}(d', c, t)$$

This gradient matches that of regularized logistic regression in a supervised model: the difference Δ between the observed and expected features, summed over every decision and context. In the supervised case, we would observe the count of occurrences of (d, c, t), but in the unsupervised M-step, we instead substitute expected counts $e_{d,c,t}$.

This gradient-based M-step is an iterative procedure. For each different value of $\mathbf{w}$ considered during the search, we must recompute $\boldsymbol{\theta}(\mathbf{w})$, which requires computation in proportion to the size of the

parameter space. However, $\mathbf{e}$ stays fixed throughout the M-step. Algorithm 1 outlines EM in its entirety. The subroutine climb$(\cdot, \cdot, \cdot)$ represents a generic optimization step such as an LBFGS iteration.

Algorithm 1 Feature-enhanced EM

repeat
  Compute expected counts $\mathbf{e}$ ▷ Eq. 2
  repeat
    Compute $\ell(\mathbf{w}, \mathbf{e})$ ▷ Eq. 3
    Compute $\nabla\ell(\mathbf{w}, \mathbf{e})$ ▷ Eq. 4
    $\mathbf{w} \leftarrow \text{climb}(\mathbf{w}, \ell(\mathbf{w}, \mathbf{e}), \nabla\ell(\mathbf{w}, \mathbf{e}))$
  until convergence
until convergence

3.2 Direct Marginal Likelihood Optimization

Another approach to optimizing Equation 1 is to compute the gradient of the log marginal likelihood directly (Salakhutdinov et al., 2003). The gradient turns out to have the same form as Equation 4, with the key difference that $e_{d,c,t}$ is recomputed for every different value of $\mathbf{w}$. Algorithm 2 outlines the procedure. Justification for this algorithm appears in the Appendix.

Algorithm 2 Feature-enhanced direct gradient

repeat
  Compute expected counts $\mathbf{e}$ ▷ Eq. 2
  Compute $L(\mathbf{w})$ ▷ Eq. 1
  Compute $\nabla\ell(\mathbf{w}, \mathbf{e})$ ▷ Eq. 4
  $\mathbf{w} \leftarrow \text{climb}(\mathbf{w}, L(\mathbf{w}), \nabla\ell(\mathbf{w}, \mathbf{e}))$
until convergence

In practice, we find that this optimization approach leads to higher task accuracy for several models. However, in cases where computing $e_{d,c,t}$ is expensive, EM can be a more efficient alternative.

4 Part-of-Speech Induction

We now describe experiments that demonstrate the effectiveness of locally normalized logistic models. We first use the bigram HMM described in Section 2.1 for POS induction, which has two types of multinomials. For type EMIT, the decisions d are words and contexts c are tags. For type TRANS, the decisions and contexts are both tags.

4.1 POS Induction Features

We use the same set of features used by Haghighi and Klein (2006) in their baseline globally normalized Markov random field (MRF) model. These are all coarse features on emission contexts that activate for words with certain orthographic properties. We use only the BASIC features for transitions. For an emission with word y and tag z, we use the following feature templates:

BASIC:	$\mathbb{1}(y = \cdot, z = \cdot)$
CONTAINS-DIGIT:	Check if y contains digit and conjoin with z: $\mathbb{1}(\text{containsDigit}(y) = \cdot, z = \cdot)$
CONTAINS-HYPHEN:	$\mathbb{1}(\text{containsHyphen}(x) = \cdot, z = \cdot)$
INITIAL-CAP:	Check if the first letter of y is capitalized: $\mathbb{1}(\text{isCap}(y) = \cdot, z = \cdot)$
N-GRAM:	Indicator functions for character n-grams of up to length 3 present in y.

4.2 POS Induction Data and Evaluation

We train and test on the entire WSJ tag corpus (Marcus et al., 1993). We attempt the most difficult version of this task where the only information our system can make use of is the unlabeled text itself. In particular, we do not make use of a tagging dictionary. We use 45 tag clusters, the number of POS tags that appear in the WSJ corpus. There is an identifiability issue when evaluating inferred tags. In order to measure accuracy on the hand-labeled corpus, we map each cluster to the tag that gives the highest accuracy, the many-1 evaluation approach (Johnson, 2007). We run all POS induction models for 1000 iterations, with 10 random initializations. The mean and standard deviation of many-1 accuracy appears in Table 1.

4.3 POS Induction Results

We compare our model to the basic HMM and a bigram version of the feature-enhanced MRF model of Haghighi and Klein (2006). Using EM, we achieve a many-1 accuracy of 68.1. This outperforms the basic HMM baseline by a 5.0 margin. The same model, trained using the direct gradient approach, achieves a many-1 accuracy of 75.5, outperforming the basic HMM baseline by a margin of 12.4. These results show that the direct gradient approach can offer additional boosts in performance when used with a feature-enhanced model. We also outperform the

globally normalized MRF, which uses the same set of features and which we train using a direct gradient approach.

To the best of our knowledge, our system achieves the best performance to date on the WSJ corpus for totally unsupervised POS tagging.[3]

5 Grammar Induction

We next apply our technique to a grammar induction task: the unsupervised learning of dependency parse trees via the dependency model with valence (DMV) (Klein and Manning, 2004). A dependency parse is a directed tree over tokens in a sentence. Each edge of the tree specifies a directed dependency from a head token to a dependent, or argument token. Thus, the number of dependencies in a parse is exactly the number of tokens in the sentence, not counting the artificial root token.

5.1 Dependency Model with Valence

The DMV defines a probability distribution over dependency parse trees. In this head-outward attachment model, a parse and the word tokens are derived together through a recursive generative process. For each token generated so far, starting with the root, a set of left dependents is generated, followed by a set of right dependents.

There are two types of multinomial distributions in this model. The Bernoulli STOP probabilities $\theta_{d,c,\text{STOP}}$ capture the valence of a particular head. For this type, the decision d is whether or not to stop generating arguments, and the context c contains the current head h, direction δ and adjacency adj. If a head's stop probability is high, it will be encouraged to accept few arguments. The ATTACH multinomial probability distributions $\theta_{d,c,\text{ATTACH}}$ capture attachment preferences of heads. For this type, a decision d is an argument token a, and the context c consists of a head h and a direction δ.

We take the same approach as previous work (Klein and Manning, 2004; Cohen and Smith, 2009) and use gold POS tags in place of words.

[3]Haghighi and Klein (2006) achieve higher accuracies by making use of labeled prototypes. We do not use any external information.

5.2 Grammar Induction Features

One way to inject knowledge into a dependency model is to encode the similarity between the various morphological variants of nouns and verbs. We encode this similarity by incorporating features into both the STOP and the ATTACH probabilities. The attachment features appear below; the stop feature templates are similar and are therefore omitted.

BASIC:	$\mathbb{1}(a = \cdot, h = \cdot, \delta = \cdot)$
NOUN:	Generalize the morphological variants of nouns by using isNoun$(\cdot)$: $\mathbb{1}(a = \cdot, \text{isNoun}(h) = \cdot, \delta = \cdot)$ $\mathbb{1}(\text{isNoun}(a) = \cdot, h = \cdot, \delta = \cdot)$ $\mathbb{1}(\text{isNoun}(a) = \cdot, \text{isNoun}(h) = \cdot, \delta = \cdot)$
VERB:	Same as above, generalizing verbs instead of nouns by using isVerb$(\cdot)$
NOUN-VERB:	Same as above, generalizing with isVerbOrNoun$(\cdot) =$ isVerb$(\cdot) \vee$ isNoun$(\cdot)$
BACK-OFF:	We add versions of all other features that ignore direction or adjacency.

While the model has the expressive power to allow specific morphological variants to have their own behaviors, the existence of coarse features encourages uniform analyses, which in turn gives better accuracies.

Cohen and Smith's (2009) method has similar characteristics. They add a shared logistic-normal prior (SLN) to the DMV in order to tie multinomial parameters across related derivation events. They achieve their best results by only tying parameters between different multinomials when the corresponding contexts are headed by nouns and verbs. This observation motivates the features we choose to incorporate into the DMV.

5.3 Grammar Induction Data and Evaluation

For our English experiments we train and report directed attachment accuracy on portions of the WSJ corpus. We work with a standard, reduced version of WSJ, WSJ10, that contains only sentences of length 10 or less after punctuation has been removed. We train on sections 2-21, and use section 22 as a development set. We report accuracy on section 23. These are the same training, development, and test sets used by Cohen and Smith (2009). The regularization parameter (κ) is tuned on the development set to maximize accuracy.

For our Chinese experiments, we use the same corpus and training/test split as Cohen and Smith

(2009). We train on sections 1-270 of the Penn Chinese Treebank (Xue et al., 2002), similarly reduced (CTB10). We test on sections 271-300 of CTB10, and use sections 400-454 as a development set.

The DMV is known to be sensitive to initialization. We use the deterministic harmonic initializer from Klein and Manning (2004). We ran each optimization procedure for 100 iterations. The results are reported in Table 1.

5.4 Grammar Induction Results

We are able to outperform Cohen and Smith's (2009) best system, which requires a more complicated variational inference method, on both English and Chinese data sets. Their system achieves an accuracy of 61.3 for English and an accuracy of 51.9 for Chinese.[4] Our feature-enhanced model, trained using the direct gradient approach, achieves an accuracy of 63.0 for English, and an accuracy of 53.6 for Chinese. To our knowledge, our method for feature-based dependency parse induction outperforms all existing methods that make the same set of conditional independence assumptions as the DMV.

6 Word Alignment

Word alignment is a core machine learning component of statistical machine translation systems, and one of the few NLP tasks that is dominantly solved using unsupervised techniques. The purpose of word alignment models is to induce a correspondence between the words of a sentence and the words of its translation.

6.1 Word Alignment Models

We consider two classic generative alignment models that are both used heavily today, IBM Model 1 (Brown et al., 1994) and the HMM alignment model (Ney and Vogel, 1996). These models generate a hidden alignment vector $\mathbf{z}$ and an observed foreign sentence $\mathbf{y}$, all conditioned on an observed English sentence $\mathbf{e}$. The likelihood of both models takes the form:

$$P(\mathbf{y}, \mathbf{z}|\mathbf{e}) = \prod_j p(z_j = i|z_{j-1}) \cdot \theta_{y_j, e_i, \text{ALIGN}}$$

[4]Using additional bilingual data, Cohen and Smith (2009) achieve an accuracy of 62.0 for English, and an accuracy of 52.0 for Chinese, still below our results.

Model		Inference	Reg	Eval
POS Induction			κ	Many-1
WSJ	Basic-HMM	EM	–	63.1 (1.3)
	Feature-MRF	LBFGS	0.1	59.6 (6.9)
	Feature-HMM	EM	1.0	68.1 (1.7)
		LBFGS	1.0	**75.5** (1.1)
Grammar Induction			κ	Dir
WSJ10	Basic-DMV	EM	–	47.8
	Feature-DMV	EM	0.05	48.3
		LBFGS	10.0	**63.0**
	(Cohen and Smith, 2009)			61.3
CTB10	Basic-DMV	EM	–	42.5
	Feature-DMV	EM	1.0	49.9
		LBFGS	5.0	**53.6**
	(Cohen and Smith, 2009)			51.9
Word Alignment			κ	AER
NIST ChEn	Basic-Model 1	EM	–	38.0
	Feature-Model 1	EM	–	**35.6**
	Basic-HMM	EM	–	33.8
	Feature-HMM	EM	–	**30.0**
Word Segmentation			κ	F1
BR	Basic-Unigram	EM	–	76.9 (0.1)
	Feature-Unigram	EM	0.2	84.5 (0.5)
		LBFGS	0.2	**88.0** (0.1)
	(Johnson and Goldwater, 2009)			87

Table 1: Locally normalized feature-based models outperform all proposed baselines for all four tasks. LBFGS outperformed EM in all cases where the algorithm was sufficiently fast to run. Details of each experiment appear in the main text.

The distortion term $p(z_j = i|z_{j-1})$ is uniform in Model 1, and Markovian in the HMM. See Liang et al. (2006) for details on the specific variant of the distortion model of the HMM that we used. We use these standard distortion models in both the baseline and feature-enhanced word alignment systems.

The bilexical emission model $\theta_{y,e,\text{ALIGN}}$ differentiates our feature-enhanced system from the baseline system. In the former, the emission model is a standard conditional multinomial that represents the probability that decision word y is generated from context word e, while in our system, the emission model is re-parameterized as a logistic regression model and feature-enhanced.

Many supervised feature-based alignment models have been developed. In fact, this logistic parameterization of the HMM has been proposed before and yielded alignment improvements, but was trained using supervised estimation techniques (Varea et al., 2002).[5] However, most full translation systems to-

[5]Varea et al. (2002) describes unsupervised EM optimization with logistic regression models at a high level—their *dynamic training* approach—but provides no experiments.

day rely on unsupervised learning so that the models may be applied easily to many language pairs. Our approach provides efficient and consistent unsupervised estimation for feature-rich alignment models.

6.2 Word Alignment Features

The BASIC features on pairs of lexical items provide strong baseline performance. We add coarse features to the model in order to inject prior knowledge and tie together lexical items with similar characteristics.

BASIC:	$\mathbb{1}(e = \cdot, y = \cdot)$
EDIT-DISTANCE:	$\mathbb{1}(\text{dist}(y, e) = \cdot)$
DICTIONARY:	$\mathbb{1}((y, e) \in D)$ for dictionary D.
STEM:	$\mathbb{1}(\text{stem}(e) = \cdot, y = \cdot)$ for Porter stemmer.
PREFIX:	$\mathbb{1}(\text{prefix}(e) = \cdot, y = \cdot)$ for prefixes of length 4.
CHARACTER:	$\mathbb{1}(e = \cdot, \text{charAt}(y, i) = \cdot)$ for index i in the Chinese word.

These features correspond to several common augmentations of word alignment models, such as adding dictionary priors and truncating long words, but here we integrate them all coherently into a single model.

6.3 Word Alignment Data and Evaluation

We evaluate on the standard hand-aligned portion of the NIST 2002 Chinese-English development set (Ayan et al., 2005). The set is annotated with sure S and possible P alignments. We measure alignment quality using alignment error rate (AER) (Och and Ney, 2000).

We train the models on 10,000 sentences of FBIS Chinese-English newswire. This is not a large-scale experiment, but large enough to be relevant for low-resource languages. LBFGS experiments are not provided because computing expectations in these models is too computationally intensive to run for many iterations. Hence, EM training is a more appropriate optimization approach: computing the M-step gradient requires only summing over word type pairs, while the marginal likelihood gradient needed for LBFGS requires summing over training sentence alignments. The final alignments, in both the baseline and the feature-enhanced models, are computed by training the generative models in both directions, combining the result with hard union competitive thresholding (DeNero and Klein, 2007), and using agreement training for the HMM (Liang et al., 2006). The combination of these techniques yields a state-of-the-art unsupervised baseline for Chinese-English.

6.4 Word Alignment Results

For both IBM Model 1 and the HMM alignment model, EM training with feature-enhanced models outperforms the standard multinomial models, by 2.4 and 3.8 AER respectively.[6] As expected, large positive weights are assigned to both the dictionary and edit distance features. Stem and character features also contribute to the performance gain.

7 Word Segmentation

Finally, we show that it is possible to improve upon the simple and effective word segmentation model presented in Liang and Klein (2009) by adding phonological features. Unsupervised word segmentation is the task of identifying word boundaries in sentences where spaces have been removed. For a sequence of characters $\mathbf{y} = (y_1, ..., y_n)$, a segmentation is a sequence of segments $\mathbf{z} = (z_1, ..., z_{|\mathbf{z}|})$ such that $\mathbf{z}$ is a partition of $\mathbf{y}$ and each z_i is a contiguous subsequence of $\mathbf{y}$. Unsupervised models for this task infer word boundaries from corpora of sentences of characters without ever seeing examples of well-formed words.

7.1 Unigram Double-Exponential Model

Liang and Klein's (2009) unigram double-exponential model corresponds to a simple derivational process where sentences of characters $\mathbf{x}$ are generated a word at a time, drawn from a multinomial over all possible strings $\theta_{z,\text{SEGMENT}}$. For this type, there is no context and the decision is the particular string generated. In order to avoid the degenerate MLE that assigns mass only to single segment sentences it is helpful to independently generate a length for each segment from a fixed distribution. Liang and Klein (2009) constrain individual segments to have maximum length 10 and generate lengths from the following distribution: $\theta_{l,\text{LENGTH}} = \exp(-l^{1.6})$ when $1 \leq l \leq 10$. Their model is deficient since it is possible to generate

[6] The best published results for this dataset are supervised, and trained on 17 times more data (Haghighi et al., 2009).

lengths that are inconsistent with the actual lengths of the generated segments. The likelihood equation is given by:

$$P(\mathbf{Y} = \mathbf{y}, \mathbf{Z} = \mathbf{z}) = \theta_{\text{STOP}} \prod_{i=1}^{|\mathbf{z}|} \left[(1 - \theta_{\text{STOP}}) \, \theta_{z_i, \text{SEGMENT}} \exp(-|z_i|^{1.6}) \right]$$

7.2 Segmentation Data and Evaluation

We train and test on the phonetic version of the Bernstein-Ratner corpus (1987). This is the same set-up used by Liang and Klein (2009), Goldwater et al. (2006), and Johnson and Goldwater (2009). This corpus consists of 9790 child-directed utterances transcribed using a phonetic representation. We measure segment F1 score on the entire corpus.

We run all word segmentation models for 300 iterations with 10 random initializations and report the mean and standard deviation of F1 in Table 1.

7.3 Segmentation Features

The SEGMENT multinomial is the important distribution in this model. We use the following features:

BASIC:	$\mathbb{1}(z = \cdot)$
LENGTH:	$\mathbb{1}(\text{length}(z) = \cdot)$
NUMBER-VOWELS:	$\mathbb{1}(\text{numVowels}(z) = \cdot)$
PHONO-CLASS-PREF:	$\mathbb{1}(\text{prefix}(\text{coarsePhonemes}(z)) = \cdot)$
PHONO-CLASS-PREF:	$\mathbb{1}(\text{suffix}(\text{coarsePhonemes}(z)) = \cdot)$

The phonological class prefix and suffix features project each phoneme of a string to a coarser class and then take prefix and suffix indicators on the string of projected characters. We include two versions of these features that use projections with different levels of coarseness. The goal of these features is to help the model learn general phonetic shapes that correspond to well-formed word boundaries.

As is the case in general for our method, the feature-enhanced unigram model still respects the conditional independence assumptions that the standard unigram model makes, and inference is still performed using a simple dynamic program to compute expected sufficient statistics, which are just segment counts.

7.4 Segmentation Results

To our knowledge our system achieves the best performance to date on the Bernstein-Ratner corpus, with an F1 of 88.0. It is substantially simpler than the non-parametric Bayesian models proposed by Johnson et al. (2007), which require sampling procedures to perform inference and achieve an F1 of 87 (Johnson and Goldwater, 2009). Similar to our other results, the direct gradient approach outperforms EM for feature-enhanced models, and both approaches outperform the baseline, which achieves an F1 of 76.9.

8 Conclusion

We have shown that simple, locally normalized models can effectively incorporate features into unsupervised models. These enriched models can be easily optimized using standard NLP building blocks. Beyond the four tasks explored in this paper—POS tagging, DMV grammar induction, word alignment, and word segmentation—the method can be applied to many other tasks, for example grounded semantics, unsupervised PCFG induction, document clustering, and anaphora resolution.

Acknowledgements

We thank Percy Liang for making his word segmentation code available to us, and the anonymous reviewers for their comments.

Appendix: Optimization

In this section, we derive the gradient of the log marginal likelihood needed for the direct gradient approach. Let $\mathbf{w}_0$ be the current weights in Algorithm 2 and $\mathbf{e} = \mathbf{e}(\mathbf{w}_0)$ be the expectations under these weights as computed in Equation 2. In order to justify Algorithm 2, we need to prove that $\nabla L(\mathbf{w}_0) = \nabla \ell(\mathbf{w}_0, \mathbf{e})$.

We use the following simple lemma: if ϕ, ψ are real-valued functions such that: (1) $\phi(\mathbf{w}_0) = \psi(\mathbf{w}_0)$ for some $\mathbf{w}_0$; (2) $\phi(\mathbf{w}) \leq \psi(\mathbf{w})$ on an open set containing $\mathbf{w}_0$; and (3), ϕ and ψ are differentiable at $\mathbf{w}_0$; then $\nabla\psi(\mathbf{w}_0) = \nabla\phi(\mathbf{w}_0)$.

We set $\psi(\mathbf{w}) = L(\mathbf{w})$ and $\phi(\mathbf{w}) = \ell(\mathbf{w}, \mathbf{e}) - \sum_{\mathbf{z}} P_{\mathbf{w}_0}(\mathbf{Z} = \mathbf{z}|\mathbf{Y} = \mathbf{y}) \log P_{\mathbf{w}_0}(\mathbf{Z} = \mathbf{z}|\mathbf{Y} = \mathbf{y})$. If we can show that ψ, ϕ satisfy the conditions of the lemma we are done since the second term of ϕ depends on $\mathbf{w}_0$, but not on $\mathbf{w}$.

Property (3) can be easily checked, and property (2) follows from Jensen's inequality. Finally, property (1) follows from Lemma 2 of Neal and Hinton (1998).

References

N. F. Ayan, B. Dorr, and C. Monz. 2005. Combining word alignments using neural networks. In *Empirical Methods in Natural Language Processing*.

N. Bernstein-Ratner. 1987. *The phonology of parent-child speech*. K. Nelson and A. van Kleeck.

M. Bisani and H. Ney. 2008. Joint-sequence models for grapheme-to-phoneme conversion.

A. Bouchard-Côté, P. Liang, D. Klein, and T. L. Griffiths. 2008. A probabilistic approach to language change. In *Neural Information Processing Systems*.

P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. 1994. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*.

S. F. Chen. 2003. Conditional and joint models for grapheme-to-phoneme conversion. In *Eurospeech*.

S. B. Cohen and N. A. Smith. 2009. Shared logistic normal distributions for soft parameter tying in unsupervised grammar induction. In *North American Chapter of the Association for Computational Linguistics*.

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 (Methodological)*.

J. DeNero and D. Klein. 2007. Tailoring word alignments to syntactic machine translation. In *Association for Computational Linguistics*.

D. Elworthy. 1994. Does Baum-Welch re-estimation help taggers? In *Association for Computational Linguistics*.

S. Goldwater and T. L. Griffiths. 2007. A fully Bayesian approach to unsupervised part-of-speech tagging. In *Association for Computational Linguistics*.

S. Goldwater, T. L. Griffiths, and M. Johnson. 2006. Contextual dependencies in unsupervised word segmentation. In *International Conference on Computational Linguistics/Association for Computational Linguistics*.

A. Haghighi and D. Klein. 2006. Prototype-driven learning for sequence models. In *Association for Computational Linguistics*.

A. Haghighi, J. Blitzer, J. DeNero, and D. Klein. 2009. Better word alignments with supervised ITG models. In *Association for Computational Linguistics*.

M. Johnson and S. Goldwater. 2009. Improving nonparametric Bayesian inference: Experiments on unsupervised word segmentation with adaptor grammars. In *North American Chapter of the Association for Computational Linguistics*.

M. Johnson, T. L. Griffiths, and S. Goldwater. 2007. Adaptor grammars: a framework for specifying compositional nonparametric Bayesian models. In *Neural Information Processing Systems*.

M. Johnson. 2007. Why doesnt EM find good HMM POS-taggers? In *Empirical Methods in Natural Language Processing/Computational Natural Language Learning*.

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

P. Liang and D. Klein. 2009. Online EM for unsupervised models. In *North American Chapter of the Association for Computational Linguistics*.

P. Liang, B. Taskar, and D. Klein. 2006. Alignment by agreement. In *North American Chapter of the Association for Computational Linguistics*.

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

M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini. 1993. Building a large annotated corpus of English: the penn treebank. *Computational Linguistics*.

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

R. Neal and G. E. Hinton. 1998. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*. Kluwer Academic Publishers.

H. Ney and S. Vogel. 1996. HMM-based word alignment in statistical translation. In *International Conference on Computational Linguistics*.

F. J. Och and H. Ney. 2000. Improved statistical alignment models. In *Association for Computational Linguistics*.

R. Salakhutdinov, S. Roweis, and Z. Ghahramani. 2003. Optimization with EM and expectation-conjugate-gradient. In *International Conference on Machine Learning*.

N. A. Smith and J. Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Association for Computational Linguistics*.

N. A. Smith and M. Johnson. 2007. Weighted and probabilistic context-free grammars are equally expressive. *Computational Linguistics*.

I. G. Varea, F. J. Och, H. Ney, and F. Casacuberta. 2002. Refined lexicon models for statistical machine translation using a maximum entropy approach. In *Association for Computational Linguistics*.

N. Xue, F-D Chiou, and M. Palmer. 2002. Building a large-scale annotated Chinese corpus. In *International Conference on Computational Linguistics*.