Collocation Lattices and Maximum Entropy Models

Andrei Mikheev
HCRC, Language Technology Group, University of Edinburgh,
2 Buccleuch Place, Edinburgh EH8 9LW, Scotland, UK.
e-mail: Andrei.Mikheev@ed.ac.uk
Steven Finch
Thomson Technical Labs,
1375 Piccard Drive, Suite 250, Rockville Maryland, 20850
e-mail: sfinch@thomtech.com

July 9, 1997

Abstract

Maximum entropy framework proved to be expressive and powerful for the statistical language modelling, but it suffers from the computational expensiveness of the model building. The iterative scaling algorithm that is used for the parameter estimation is computationally expensive while the feature selection process requires to estimate parameters of the model for many candidate features many times. In this paper we present a novel approach for building maximum entropy models. Our approach uses a features collocation lattice and selects the atomic features without resorting to iterative scaling. After the atomic features have been selected we, using the iterative scaling, compile a fully saturated model for the maximal constraint space and then start to eliminate the most specific constraints. Since during constraint deselection at every point we have a fully fit maximum entropy model, we rank the constraints on the basis of their weights in the model. Therefore we don't have to use the iterative scaling during constraint ranking and apply it only for linear model regression. Another important improvement is that since the simplified model deviates from the previous larger model only in a small number of constraints, we use the parameters of the old model as the initial values of the parameters for the iterative scaling of the new one. This proved to decrease the number of required iterations by about tenfold. As practical results we discuss how our method has been applied to several tasks of language modelling such as sentence boundary disambiguation, part-of-speech tagging and automatic document abstracting.

1 Introduction

Maximum entropy modelling has been recently introduced to the NLP community and proved to be an expressive and powerful framework. The maximum entropy model is a model which fits to a set of pre-defined constraints and assumes maximum ignorance about everything which is not subject to its constraints thus assigning such cases with the most uniform distribution. The most uniform distribution will have the entropy on its maximum and the model is chosen according to:

$$model = argmax_{p \in constraints}\, H(p) \quad where \quad H(p) = -\sum_i p_i * log_2\, p_i \qquad (1)$$

For instance, if we want to disambiguate part-of-speech of a word and we observed that in 50% of the times a noun is preceeded by a determiner and in 30% of the times it is preceeded by an adjective, we can state these observations as constraints to the model. Thus our model will have to choose a probability distribution for parts-of-speech which will agree with our observations and which will assign all the cases when a word which can be a noun is preceeded with neither a determiner nor an adjective with equal probabilities.

One of the most popular maximum entropy distributions is known as Gibbs distribution. It defines a model as a set of Lagrange multipliers $\langle Z, \lambda_0..\lambda_n \rangle$ and has an exponential form:

$$p(w_i) = \frac{1}{Z} e^{\sum_{\chi_j \in \chi} \lambda_{\chi_j} * f_{\chi_j}(w_i)} \quad where \tag{2}$$

- Υ is a set of instantiated by their values atomic features of the model;
- w_i is a described by the model entity which can be represented as a configuration of the instantiated atomic features from Υ. We will call w_i a configuration from configuration space W. The configuration space W includes not only observed configurations (w) but rather all possible in the domain configurations many of which might have not ever been observed;
- χ_j is a constraint from the constraint space χ imposed to the model. It is essentially a combination of instantiated atomic features too. We can look at the constraints as at employed by the model features[1]. In the rest of the paper we will use the terms constraint feature, feature and constraint interchangeably;
- $f_{\chi_j}(w_i)$ is the indicator function which indicates whether or not the j-th constraint (χ_j) is active for the configuration w_i. This function takes two values: 1 - if the constraint is active and 0 otherwise;
- λ_{χ_j} (Lagrange multiplier) is the weight of the j-th constraint (χ_j);
- Z is the normalization constant which ensures that the probabilities for all configurations sum up to 1:
$$Z = \sum_{w_i \in W} e^{\sum_{\chi_j \in \chi} \lambda_{\chi_j} * f_{\chi_j}(w_i)} \tag{3}$$

Apart from being a distribution of maximum entropy this distribution also possesses a very important property of model decomposition. For instance, if our atomic feature set Υ includes word length $(1, 2, 3, 4, 5..)$, its capitalization (Cap) and whether it ends with the full-stop ($fstop$), and we want to obtain the probability of spelling the word "Mr." , we will have the equation as follows:

$$p(Mr.) = \frac{1}{Z} e^{\lambda_1 * 0 + \lambda_2 * 0 + \lambda_3 * 1 + \lambda_4 * 0 + \lambda_5 * 0 + \lambda_{Cap} * 1 + \lambda_{fstop} * 1} = \frac{1}{Z} e^{\lambda_3 + \lambda_{Cap} + \lambda_{fstop}}$$

There is nothing interesting about the model above since it constrains only atomic *non-overlapping* i.e. independent features. The main strength of the Gibbs distribution is that it can handle complex overlapping features and therefore account for feature interaction. For instance, we might notice that the capitalization, when it is seen particularly with words of length 3, has a different distribution from its general one. To model this observation we can introduce a *complex feature* (i.e. we set a complex constraint) which is a logical conjunction or collocation of the two atomic features 3 and Cap. Now our model will predict the probability for the word "Mr." as:

[1]Note here the difference between atomic features and constraint features: constraint features consist from atomic features but we can have a set of constraints which does not include some or even all atomic features per se but only their combinations.

$$p(Mr.) = \frac{1}{Z} e^{\lambda_3 + \lambda_{Cap} + \lambda_{fstop} + \lambda_{(3,Cap)}}$$

Important thing to note here is that we still constrain the atomic features λ_3 and λ_{Cap} together with their collocation feature $\lambda_{(3,Cap)}$. So $\lambda_{(3,Cap)}$ has only the excess weight which differentiates $p(3, Cap)$ from the product of $p(3)$ and $p(Cap)$ – the case if there were no feature interaction. Such decomposition of complex features into simpler ones provides an elegant way of representing cases with interactions of many overlapping features of high complexity.

Because of its ability to handle overlapping features the maximum entropy framework provides a better way to incorporate multiple knowledge sources than traditionally used for this purpose linear interpolation and Katz back-off method (Katz 1987). Rosenfeld 1996 evaluates in detail a maximum entropy model which combines unigrams, bigrams, trigrams and long-distance trigger words for the prediction of the next word. The linear interpolation combines such knowledge sources simply by weighting them as $p_{combined} = \sum_{i=1}^{k} \lambda_i p_i$ for k knowledge sources. It does not, however, model the interaction between different knowledge sources[2] and only provides the best weights for the them under the assumption of independence. The back-off method, in fact, does not combine different knowledge sources but rather *rank* them. This allows for using the most informative method first and back-off to a less informative method if there is not enough information for a more informative one. For instance, we can try to use the trigram model first, and only when there is no suitable trigram known to the model, we back-off to the bigram model. As the linear interpolation the back-off method does not account for possible interactions between different knowledge sources which can lead to overestimation of some events.

The maximum entropy framework naturally combines the good sides of the two methods and at the same time it accounts for the interactions between features. Every knowledge source produces a set of constraints which are used together with constraints from other knowledge sources – so no interpolation needed. Simple and complex features together with their overlaps are naturally incorporated into the model and all the interactions are naturally accounted for. Because of the decompositional nature of the maximum entropy model, it can act as a back-off model too – overlapping simpler features naturally coexist with more complex ones and the weights of the complex features are just the excess on which they different from their constituent simpler features.

Applying the exponential distribution discussed above the maximum entropy approach developed in Della Pietra et al. 1995 defines a framework for the selection of the best performing constraint set and for the estimation of the weights (λs) for these constraints. The model induction procedure has two parts: *feature selection* and *parameter estimation*, both of which agree with the principle of maximum entropy. In this paper we present a novel approach to feature selection for the maximum entropy models. This approach requires less computational load than the one developed in Della Pietra et al. 1995 at a price of being not yet suitable for building models with a very large (hundreds of thousands) set of parameters. We also propose a slight modification to the process of parameter estimation for the conditional maximum entropy models. Our method uses assumptions similar to Berger et al. 1996 but is naturally suitable for distributed parallel computations.

2 Parameter Estimation

The parameter estimation for the exponential maximum entropy distribution is based on the Improved Iterative Scaling algorithm presented in Della Pietra et al. 1995. Its task is to determine weights (λ) for all the constraint features of a given constraint space so that the resulting dis-

[2]For instance, the probabilities for bigrams and trigrams are clearly not independent from each other.

tribution will closely model some reference distribution which is usually taken as the empirical distribution of the configuration space. The essence of the parameter estimation is as follows:

- there is a model with a set of atomic features Υ and a constraint feature space χ. Every constraint feature consists of one or more atomic feature from Υ. For instance, we can define a model with atomic features $\Upsilon = \langle 2,3,4,5,Cap,fstop\rangle$ and a possible feature space $\chi = \{2,3,Cap,(2,Cap),(5,Cap,fstop)\}$.
- there is a sample of entities $w = \{w_0 ... w_m\}$ which are representable as configurations of atomic features[3] from Υ. We define a function $\Phi : w \rightarrow \Upsilon$ which maps entities w into Υ, for instance, $\Phi(Mr.) \rightarrow \langle 3, Cap, fstop\rangle$. Here we will refer to w-s as configurations meaning that they are mapped into atomic features.
- all the features from the constraint feature space χ have corresponding indicator functions $f_{\chi_0}(w) ... f_{\chi_n}(w)$ to flag whether a certain constraint feature is active for a particular configuration from the configuration space:

$$f_{\chi_k}(w_i) = \begin{cases} 1 & if \;\; \chi_k \subseteq \Phi(w_i) \\ 0 & otherwise \end{cases} \tag{4}$$

  For instance: $f_{(3,Cap)}(Mr.) = 1$ since $\Phi(Mr.) \rightarrow \langle 3, Cap, fstop\rangle$ and $(3, Cap) \subseteq (3, Cap, fstop)$.
- we choose a reference distribution to fit our model to, so we can associate every feature from the model's constraint space with a corresponding reference probability: $\tilde{p}(\chi_0) ... \tilde{p}(\chi_n)$. Usually the reference distribution is simply the empirical distribution of our features in the configuration space. In this case the reference probability for a feature is computed as:

$$\tilde{p}(\chi_k) = \sum_{w_i \in w} f_{\chi_k}(w_i) * \tilde{p}(w_i) \tag{5}$$

  where $\tilde{p}(w_i)$ is the frequency count of the i-th configuration over the total number of observed configurations. Note here that the total sum of feature probabilities can be greater than 1: $\sum_{\chi_k \in \chi} \tilde{p}(\chi_k) \geq 1$ since features can overlap with each other.
- we constrain our features to their reference probabilities: $\tilde{p}(\chi_k) \approx p(\chi_k)$ and using equation 5 obtain:

$$\tilde{p}(\chi_k) = \sum_{w_i \in w} f_{\chi_k}(w_i) * \tilde{p}(w_i) \approx \sum_{w_i \in W} f_{\chi_k}(w_i) * p(w_i) = p(\chi_k) \tag{6}$$

  where $\tilde{p}(w_i)$ is estimated from the sample of configurations and $p(w_i)$ is computed using equation 2. Note here that in the second part of the equation we sum over all possible configurations (W) in the domain.
- To fit the model to its reference distribution we use the Improved Iterative Scaling algorithm:
  1. we initialize all weights (lambdas) of the features from our constraint space with some initial values. Usually if we don't have other evidence we start with the uniform distribution so all lambdas are set to the same value e.g. 0: $\lambda_{\chi_0} = 0 ..... \lambda_{\chi_n} = 0$.

[3]Indeed, the feature space and the mapped configuration space can be identical if we want to memorize in our model all and only seen cases. This can lead, however, to the overfitting of the model and such model also will not possess any generalization power, so its performance on unseen cases might be rather poor.

2. we calculate the normalization constant Z using equation 3 and the maximum entropy probabilities $(p(w_0)....p(w_m)$ for each configuration from the total configuration space W using equation 2.
3. for each feature from the model's constraint space we apply the constraint as in equation 6 and compute how its weight should be adjusted $(\Delta\lambda)$ to satisfy the constraint:
$$\tilde{p}(\chi_k) \approx \sum_{w_i \in W} f_{\chi_k}(w_i) * p(w_i) * e^{\Delta\lambda_{\chi_k} * \sum_{\chi_j \in \chi} f_{\chi_j}(w_i)} \quad (7)$$
where $f_{\chi_k}(w_i)$ ensures that only those configurations (w_i) which include the constrained feature χ_k contribute to the mass probability, and $\sum_{\chi_j \in \chi} f_{\chi_j}(w_i)$ is simply the number of all the features from the model's feature space[4] which are active for the i-th contributing configuration (w_i). This ensures that we account only for that proportion which belongs to χ_k in the contributing configurations. In general there is no analytical solution to such equations and the most popular numerical method is Newton's method where we fit $\Delta\lambda$ iteratively.
4. if the greatest $\Delta\lambda$ computed at the previous step is smaller than a certain threshold – the algorithm has converged and we exit. Otherwise we update the model's weights as: $\lambda_{\chi_k} = \lambda_{\chi_k} + \Delta\lambda_{\chi_k}$ and go to step 2 to obtain a better fit of the model.

- As the result we have a fully specified maximum entropy model of the form: $\langle Z, \lambda_{\chi_0} ......\lambda_{\chi_n}\rangle$

2.1 Computing Conditional Models

Generalized Iterative Scaling algorithm presented above defines a way for the computing of Maximum Entropy models for joint probability distributions. With such models we can answer questions such as what is the probability of *generating* an entity described as a configuration of atomic features. For instance, a joint model can predict how likely it is to generate a capitalised word with suffix "ing" : $p(capitalized = YES, suf = ing)$. In the statistical language modelling we, however, are often concerned with the *conditional probabilities*: what will be the probability of Y to take a certain value y if we see a feature-configuration x. For instance, a conditional model can predict how likely that a word will be a verb if the previous word was a noun and the previous but one word was an adjective: $p(word_i = VB \mid word_{i-1} = NN, word_{i-2} = ADJ)$. For supervised training of conditional models the sample space consists of configurations which include features from two non-overlapping sets: factor features (X) and behavior features (Y). Using such joint configurations $w = (x, y)$ we have to estimate a conditional model which would predict a behaviour variable (Y) given a configuration of factor variables (X): $p(y \mid x)$.

There is an obvious simple way to compute a conditional model by computing a joint model X, Y for every value of the behavior variable separately and then the conditional model is computed as:
$$p(y \mid x) = \frac{p(y,x)}{p(x)} \quad \textit{where} \quad p(x) = \sum_{i \in Y} p(i,x) \quad (8)$$

[4]Note that this is not just the number of atomic features which compose the i-th configuration but rather the number of all registered in the model features (atomic and complex) which are active for it. For instance, if our our feature space is $[3, Cap, fstop, (3, fstop), (Cap, fstop), (3, Cap, fstop)]$ the constraint for the feature (3) will look like:
$\tilde{p}(3) \approx p(w \rightarrow \langle 3 \rangle) * e^{\Delta\lambda_3 * 1} + p(w \rightarrow \langle 3, fstop \rangle) * e^{\Delta\lambda_3 * 3} + p(w \rightarrow \langle 3, Cap \rangle) * e^{\Delta\lambda_3 * 2} + p(w \rightarrow \langle 3, Cap, fstop \rangle) * e^{\Delta\lambda_3 * 6}$

with a suitable choice of Z for each joint model. This approach is naturally suitable for distributed computation but as it was pointed out in Rosenfeld 1996 it is not a good way to proceed because every behavior is activated only by a fraction of possible factors but we will estimate their joint model on the whole space of possible configurations (W). Apart from the computational overload this will require training data well beyond usually available in the training samples.

Berger et al. 1996 presented a way of computing conditional maximum entropy models directly by modifying equation 6 as follows (now instead of w we will explicitly use (x, y)):

$$\tilde{p}(\chi_k) = \sum_{x \in X \; y \in Y} f_{\chi_k}(x,y) * \tilde{p}(x,y) \approx \sum_{x \in X \; y \in Y} f_{\chi_k}(x,y) * \tilde{p}(x) * p(y \mid x) = p(\chi_k) \tag{9}$$

where $\tilde{p}(x, y)$ - is an empirical probability of a joint configuration (w) of certain instantiated factor variables with certain instantiated behavior variables. $\tilde{p}(x)$ is the marginal empirical probability of the factor variables. The constraint solving equation 7 is then correspondingly adjusted as:

$$\tilde{p}(\chi_k) \approx \sum_{x \in X \; y \in Y} f_{\chi_k}(x,y) * \tilde{p}(x) * p(y \mid x) * e^{\Delta\lambda_{\chi_k} * \sum_{\chi_j \in \chi} f_{\chi_j}(x,y)} \tag{10}$$

and the normalization constant $Z(x)$ ensures that $\sum_{y \in Y} p(y \mid x) = 1$. This approach differs from the standard approach for joint distribution (equation 7) in plugging in the empirical marginal estimate $\tilde{p}(x)$. This restricts the constraint set only to those cases which were actually observed in the training samples for a particular value of the behavior variable (y) and in solving a constraint we only sum over seen configurations rather than all possible ones.

In this paper we propose to adjust the first method with restrictions similar to that of the second method. We will compute joint distributions separately thus benefitting from the possibility of distributed computations – each joint model can be independently computed on a separate processor or machine using multi-threading together with remote process calls (RPC). At the same time we will restrict the configuration space from all possible configurations in the domain (W) only to the observed and *logically implied* configurations (w^+) as it is described in section 6. We will require that the normalization constants Z for each joint model $\langle X, y \rangle$ ensure that all probabilities in a joint model sum up to the empirical marginal probability of the behavior variable $\tilde{p}(y)$, thus accounting only for the true proportions of the joint models. We also adopt a further yet simplification suggested in Ristard 1996 to restrict the constraints only to the cases when the overall joint frequency of a feature $\chi_k = (x, y)$ is greater than a certain threshold, for instance 5.

3 Feature Selection

The iterative scaling algorithm applied for the parameter estimation provides us with a set of λs which ensure that the model fits to the reference distribution and does not make spurious assumptions (as required by the maximum entropy principle) about events beyond the reference events. It, however, does not guarantee that the features employed by the model are good features and the model is useful. Thus the most important part of the model building is the feature selection procedure. The key idea of the feature selection is that if we notice an interaction between certain features we should build a more complex feature which will account for this interaction. The newly added feature should improve the model: its Kullback-Leibler divergence from the reference distribution should decrease:

$$model = argmin\ D(\tilde{p} \,||\, p) \quad where \quad D(\tilde{p} \,||\, p) = \sum_{w_i \in w} \tilde{p}(w_i) * log \frac{\tilde{p}(w_i)}{p(w_i)} \tag{11}$$

For a conditional model Kullback-Leibler divergence is computed as:

$$D(\tilde{p} \,\|\, p) = \sum_{x \in X y \in Y} \tilde{p}(x,y) * log \frac{\tilde{p}(y \mid x)}{p(y \mid x)} \tag{12}$$

and the conditional maximum entropy model will also have the greatest log-likelihood (L) value:

$$model = argmax\ L_{\tilde{p}}(p) \quad where \quad L_{\tilde{p}}(p) = \sum_{x \in X y \in Y} \tilde{p}(x,y) * log\, p(y \mid x) \tag{13}$$

The basic constraint feature induction algorithm presented in Della Pietra et al. 1995 starts with an empty feature space and iteratively tries all possible feature candidates which are either atomic features or complex features produced as a combination of an atomic feature with the features already selected to the model's feature space. For every feature from the candidate feature set the algorithm prescribes to compute the maximum entropy model using the iterative scaling algorithm described above, and select the feature which minimized the Kullback-Leibler divergence or maximized the log-likelihood of the model in the largest way. This approach, however, is not computationally feasible since the iterative scaling is computationally expensive and to compute models for many candidate features many times is unreal. To make feature ranking computationally tractable in Della Pietra et al. 1995 and Berger et al. 1996 a simplified process proposed: at the feature ranking stage when adding a new feature to the model, all previously computed parameters are kept fixed and, thus, we have to fit only one new constraint imposed by a candidate feature. Then after the best ranked feature has been established it is added to the feature space and the weights for all the features are recomputed. This approach allows for estimating good features relatively fast but it does not guarantee that at every single point we add the best feature because when we add a new feature to the model *all* its parameters can change.

In this paper we present a novel approach to feature selection for the maximum entropy models. Our approach uses a feature collocation lattice and selects the atomic features without resorting to the iterative scaling. After the atomic features have been selected we, using the iterative scaling, compute a fully saturated model for the maximal constraint space and then the algorithm starts to eliminate the most specific constraints. Since during constraint deselection at every point we have a fully fit maximum entropy model, we rank the features on the basis of their weights in the model. Therefore we don't have to use the iterative scaling for constraint ranking and apply it only for linear model regression. Another important improvement is that since the simplified model deviates from the previous larger model only in a small number of constraints, we use the parameters of the old model as the initial values of the parameters for the iterative scaling of the new one. This proved to decrease the number of required iterations by about tenfold. In the rest of the paper we first introduce the feature collocation lattice as a graphical way to represent complex models, then we introduce a feature selection process using the collocation lattice and finally, we present some details of application of our method to the tasks of sentence boundary disambiguation, part-of-speech tagging and automatic document abstracting via sentence extraction.

4 Feature Collocation Lattice

When we have a set of atomic features Υ and a training sample of configurations w, we can build a feature collocation lattice. Such collocation lattice will represent, in fact, the factorial constraint space (χ) for the maximum entropy model and at the same time will contain all seen and logically implied configurations (w^+). Formally, the feature collocation lattice is a 3-ple: $\langle \theta, \subseteq, \xi^w \rangle$ where

- θ is a set of nodes of the lattice which corresponds to the union of the feature space of the maximum entropy model and the configuration space: $\theta = \chi \cup \Phi(w)$. In fact, the nodes in the lattice (θ) can have dual interpretation – on one hand they can act as mapped configurations from the extended configuration space (w^+) and on the other hand they can act as features from the constraint space (χ);
- $\subseteq$ is a transitive, antisymmetric relation over $\theta \times \theta$ – a partial ordering. We also will need the indicator function similar to one in equation 4 to indicate whether the relation $\subseteq$ holds from node i to node k:

$$f_{\theta_i}(\theta_k) = \begin{cases} 1 & \mathit{if}\ \ \theta_i \subseteq \theta_k \\ 0 & \mathit{otherwise} \end{cases}$$

- ξ^w is a set of *configuration frequency counts* of the nodes (θ) of the lattice. This represents how many times we saw this particular configuration in our training samples. Because of the dual interpretation of the nodes a node can also be associated with its *feature frequency count* i.e. the number of times we see this feature combination anywhere in the lattice. The *feature frequency* of a node (similar to equation 5) will then be $\xi^\chi(\theta_k) = \sum_{\theta_i \in \theta} f_{\theta_k}(\theta_i) * \xi^w_{\theta_i}$ which is the sum of all the configuration frequency counts (ξ^w) of the descendant nodes.

  Suppose we have a lattice of nodes $A, B, (AB)$ with obvious relations: $A \subseteq (AB); B \subseteq (AB)$:

$$\begin{tikzcd} & (AB) & \\ A \arrow[ur] & & B \arrow[ul] \end{tikzcd}$$

  The configuration frequency ξ^w_A will be the number of times we saw A but not (AB) and then the feature frequency of A will be: $\xi^\chi_A = \xi^w_A + \xi^w_{AB}$ i.e. the number of times we saw A in all the nodes.

When we construct a feature collocation lattice from a set of samples, each sample represents a feature configuration which we must add to the lattice as a node (θ_k) if is not already there. To support generalizations over domain we also want to add to the lattice those nodes which were not seen on their own but only as common parts of other nodes in the lattice. Thus we add to the lattice all sub-configurations of a newly added configuration which are the intersections with the other nodes. We increment the configuration frequency ($\xi^w_{\theta_k}$) of a node each time we see in the training samples this particular configuration on its own but not as a part of another configuration. For example, if a configuration $(ABCD)$ comes from a training sample and it is still not in the lattice, we create a node $(ABCD)$ and set its configuration frequency $\xi^w_{(ABCD)}$ to 1. If by that time there is a node $(ABDE)$ in the lattice, we then also create the node (ABD), relate it to the nodes $(ABCD)$ and $(ABDE)$ and set its configuration frequency to 0. If $(ABCD)$ had already been in the lattice we would simply incremented its configuration frequency: $\xi^w_{(ABCD)} = \xi^w_{(ABCD)} + 1$.

Thus in the feature lattice we have nodes with non-zero configuration frequencies, which we call *reference nodes* and nodes with zero configuration frequencies which we call latent or *hidden nodes*. Reference nodes actually represent the observed configuration space (w). Hidden nodes are never observed on their own but only as parts of the reference nodes and represent possible generalizations about domain: low-complexity constraints (χ) and logically possible configurations (w^+).

Sometimes, there develops a class of hidden nodes which does not provide good generalizations – those hidden nodes that directly support less than two higher level nodes. By support here we mean the $\subseteq$ relation and the direct support is that there is no intermediate nodes between two

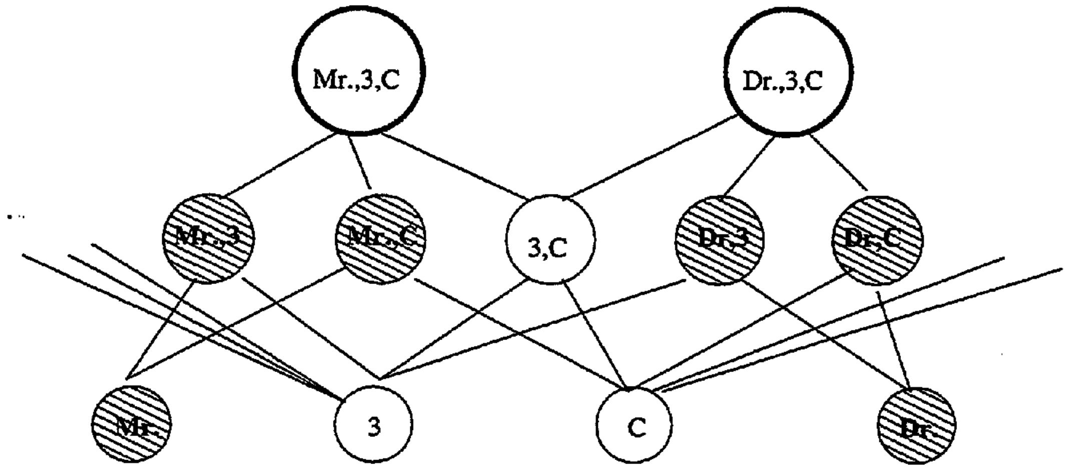


Figure 1: This figure shows a feature lattice where thick circles represent reference nodes and filled circles represent obsolete hidden nodes. 3 indicates that a word has length 3, C indicates that a word is capitalised, $Mr.$ indicates that a word has spelling "Mr." and $Dr.$ indicates that a word has spelling "Dr.".

nodes linked with the $\subseteq$ relation. Such redundant nodes might develop because we explicitly put all the atomic features into the lattice but some of them never act on their own. Figure 1 shows an excerpt from a feature lattice with the atomic features including: word length $(1, 2, 3, 4, 5..)$, capitalization (C), spellings $(Mr., Dr...)$ and others. The nodes $(Dr., 3, C)$, and $(Mr., 3, C)$ are the reference nodes – they were observed in full[5] and have non-zero configuration frequency counts $(\xi^w_{(Dr.,3,C)} > 0 \quad \xi^w_{(Mr.,3,C)} > 0)$. All other nodes on figure 1 are hidden nodes that were observed only as parts of the higher level nodes[6]. Hidden nodes $(Dr., 3)$, $(Mr., 3)$, $(Mr., C)$ and $(Dr., C)$ directly support only one node each and thus do not provide any generalizations. Therefore these nodes are obsolete and can be safely removed from the lattice. The nodes $(Dr.)$ and $(Mr.)$ then become obsolete as well, since they will not support directly any node, so we can safely remove them from the lattice too. Nodes (3) and (C) apart from supporting the node (3,C) support some other nodes not represented on figure 1, and thus should be retained in the lattice.

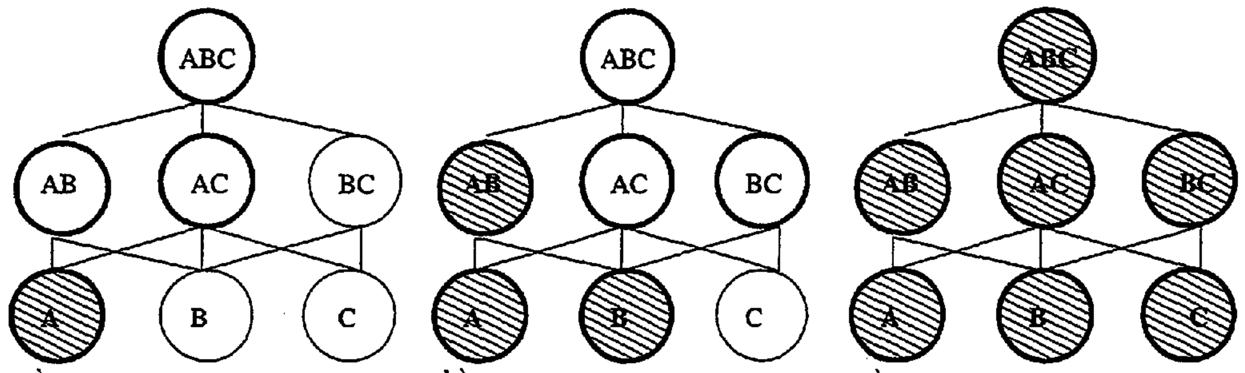
This method of building the feature collocation lattice ensures that along with true observations it contains hidden nodes which can provide generalizations about domain, and at the same time there is no over-generation of the hidden nodes: no logically impossible feature combinations and no hidden nodes without generalization power are included. Such collocation lattice was successfully applied for representing queries for document retrieval (Finch 1997) where it behaved as a combination of the boolean and the vector space models.

5 Atomic Feature Selection

After from a set of samples we have constructed a feature collocation lattice $\langle \theta, \subseteq, \xi^w \rangle$ which we will call the empirical lattice, we try to estimate which atomic features contribute and which do not to the frequency distribution on the reference nodes. Thus we will retain in the lattice only the predictive atomic features. The optimized feature space can be seen as a feature lattice defined

[5]Indeed, any time we see a configuration for the words "Mr." and "Dr." it always includes their length 3 and their capitalization feature C together with their spellings.

[6]For instance, it is impossible to see the configuration $(Dr., C)$ without seeing the configuration $(Dr., 3, C)$.



a)
$\xi'^{w}_{A} = \xi^{w}_{A} + \xi^{w}_{AB} + \xi^{w}_{AC} + \xi^{w}_{ABC}$
$\xi'^{w}_{B} = \xi^{w}_{B} + \xi^{w}_{BC}$
$\xi'^{w}_{AB} = \xi^{w}_{AB} + \xi^{w}_{ABC}$
$\xi'^{w}_{ABC} = \xi^{w}_{ABC}$

b)
$\xi'^{w}_{A} = \xi^{w}_{A} + \xi^{w}_{AC}$
$\xi'^{w}_{B} = \xi^{w}_{B}$
$\xi'^{w}_{AB} = \xi^{w}_{AB}$

c)
$\xi'^{w}_{A} = \xi^{w}_{A}$ $\xi'^{w}_{C} = \xi^{w}_{C}$
$\xi'^{w}_{AC} = \xi^{w}_{AC}$
$\xi'^{w}_{BC} = \xi^{w}_{BC}$

Figure 2: This figure shows the redistribution of configuration frequencies in the optimized feature lattice when adding new atomic nodes. Case a) stands for adding the atomic feature A to the empty lattice, case b) stands for adding the atomic feature B to the lattice with the atomic feature A and case c) stands for adding the atomic feature C to the lattice with the atomic features A and B and their collocations. The unfilled nodes stand for the nodes in the empirical lattice which don't have reference in the optimized lattice. The nodes in bold stand for the nodes decided by the optimized lattice (i.e. they can be assigned with some probabilities).

over the empirical feature lattice: $\theta' \subseteq \theta$ and initially it is empty: $\theta' = \emptyset$. We build the optimized lattice by incrementally adding an atomic feature from the empirical lattice together with the nodes which are the minimal collocations of this atomic feature with the nodes already included into the optimized lattice. So if in the optimized feature lattice there is just one feature A, then when we add the atomic feature B we also have to add the collocation (AB) if it exists in the empirical lattice. The configuration frequency of a node in the optimized lattice (ξ'^{w}) then can be computed as:

$$\xi'^{w}_{\theta_k} = \sum_{[\theta_i \in \theta \ \& \ \theta_i \notin \theta' \ \& \ \nexists \theta_j : \theta_j \in \theta' \ \& \ \theta_k \subseteq \theta_j \ \& \ \theta_j \subseteq \theta_i]} f_{\theta_k}(\theta_i) * \xi^{w}_{\theta_i} \qquad (14)$$

Thus a node in the optimized lattice takes all configuration frequencies (ξ^{w}) of itself and the above related nodes if these nodes do not belong to the optimized lattice themselves and there is no higher node in the optimized lattice related to them.

Figure 2 shows how the configuration frequencies in the optimized lattice are redistributed when adding a new atomic feature. First the lattice is empty. When we add the atomic feature A to the optimized lattice (figure 2.a), because no other features are present in the optimized lattice, it takes all the configuration frequencies of the nodes where we see the feature A: $\xi'^{w}_{A} = \xi^{w}_{A} + \xi^{w}_{AB} + \xi^{w}_{AC} + \xi^{w}_{ABC}$. Case b) of figure 2 represents the situation when we add the atomic feature B to the optimized lattice which already includes the atomic feature A. Apart from the node B we also add to the optimized lattice the collocation of the nodes A and B. Now we have to redistribute the configuration frequencies in the optimized lattice. The configuration frequency of the node A now will become the number of times of seeing the node A but not the node AB: $\xi'^{w}_{A} = \xi^{w}_{A} + \xi^{w}_{AC}$. The configuration frequency of the node B will be the number of times of seeing the node B but not the node AB: $\xi'^{w}_{B} = \xi^{w}_{B} + \xi^{w}_{BC}$. The configuration frequency of the node AB will be: $\xi'^{w}_{AB} = \xi^{w}_{AB} + \xi^{w}_{ABC}$. When we add the atomic feature C to the optimized lattice (figure 2.c) we produce a fully saturated lattice identical to the empirical lattice, since the node C will collocate with the node A producing AC and will collocate with the node B producing BC. These nodes in their turn will collocate with each other and with the node AB producing the node ABC.

During the building of the optimized lattice all the atomic features from the empirical lattice compete, and we include the one which results in a new optimized lattice with the smallest divergence $D(p \parallel p')$ (see equation 11 and equation 12) and therefore with the greatest log-likelihood $L_p(p')$ according to the optimization task stated in equation 13, where:

- in the selection of features for a conditional model the only extra calculation is to convert joint probabilities of nodes $\theta = (x, y)$ to the conditional ones. This is easily done by using the equations as follows:

$$p(y|x) = \frac{p(\theta_i = (x,y))}{\sum_{\theta_k \in \theta} f_{\theta_k}(x) * p(\theta_k)} \quad p'(y|x) = \frac{p'(\theta_i = (x,y))}{\sum_{\theta_k \in \theta} f_{\theta_k}(x) * p'(\theta_k)} \tag{15}$$

  Note here that only reference nodes from the empirical lattice contribute to $D(p \parallel p')$ and $L_p(p')$ estimations since hidden nodes always have their empirical probabilities ($\tilde{p}$) set to 0.

- $p(\theta_i)$ is the probability for the i-th node in the empirical lattice:

$$p(\theta_i) = \frac{\xi^{w}_{\theta_i}}{N} \qquad \textit{where} \quad N = \sum_{\theta_j \in \theta} \xi^{w}_{\theta_j} \tag{16}$$

- $p'(\theta_i)$ is the probability assigned to the i-th node using only the nodes included into the optimized lattice.

$$p'(\theta_i) = \begin{cases} \frac{\xi'^{w}_{\theta_i}}{N} \;\; \textit{if} \;\; \theta_i \in \theta' \\ \frac{\xi'^{w}_{\theta_j}}{N} \;\; \textit{if} \;\; \theta_i \notin \theta' \;\; \& \;\; [\exists \theta_j : \; \theta_j \in \theta' \;\; \& \;\; \theta_j \subseteq \theta_i] \;\; \& \;\; [\nexists \theta_k : \; \theta_k \in \theta' \;\; \& \;\; \theta_k \subseteq \theta_i \;\; \& \;\; \theta_j \subseteq \theta_k] \\ 1/\mid Y \mid \;\; \textit{otherwise} \end{cases} \tag{17}$$

  The optimized lattice assigns the probability to a node in the empirical lattice equal to that of its most specific sub-node[7] from the optimized lattice. For reference nodes which do not have sub-nodes in the optimized lattice at all (undecided nodes) according to the maximum entropy principle we assign the uniform probability of making an arbitrary prediction.

For instance, for the example on figure 2.b the optimized lattice includes only three nodes but there is just one undecided node (C) which is not shown in bold. So the probabilities for the nodes will be:

$$p(A) = \frac{\xi'^{w}_{A}}{N} \;\; p(B) = \frac{\xi'^{w}_{B}}{N} \;\; p(AB) = \frac{\xi'^{w}_{AB}}{N} \;\; p(AC) = \frac{\xi'^{w}_{A}}{N} \;\; p(BC) = \frac{\xi'^{w}_{B}}{N} \;\; p(ABC) = \frac{\xi'^{w}_{AB}}{N} \;\; p(C) = \frac{1}{|Y|}$$

N is the total count on the empirical lattice and is calculated as shown in equation 16:

$$N = \xi^{w}_{A} + \xi^{w}_{B} + \xi^{w}_{C} + \xi^{w}_{AB} + \xi^{w}_{AC} + \xi^{w}_{ABC}.$$

[7]Since when we add an atomic feature to the optimized lattice we also add all its relevant collocations, for every node in the empirical lattice which does not belong to the optimized lattice there is always no more than one most specific sub-node in the optimized lattice. This can be the node itself.

The presented above method provides us with an efficient way of selecting only important atomic features from the initial set of candidate atomic features without resorting to iterative scaling. When this way we add atomic features to the optimized lattice, some of the features might turn out not to contribute or contribute only on a very small scale to the probability distribution on the lattice. Such redundant atomic features can be classified into three categories: a) features which are simply not informative; b) features which are always seen with some other feature (co-founded features) and c) features which are mutually exclusive with some other feature. In the first case such features are usually present randomly in the reference nodes of the empirical lattice and therefore they do not have any discriminative power. In the second case all the relevant reference nodes will be already correctly decided by the time we will try to add another co-founded feature. Imagine that in the example on figure 2 most of the times when we saw the feature B we saw the feature C as well. So the redistribution of the configuration frequencies as on (figure 2.c) will not bring much advantage and we can safely leave the feature C out of the optimized lattice and therefore use the atomic feature set as on figure 2.b. In the third case the configuration frequency remaining on the parent node will account for the case of having two mutually exclusive higher level nodes. Imagine that B and C in the example above are two mutually exclusive features and we never see them on their own. So our lattice will consist from the nodes A, B, C, AB and AC. Seeing the node AC is equal of not seeing the node AB when seeing A, and this is already presupposed by the configuration frequency $\xi_A^{\prime w}$ when the node AB but not the node AC is in the lattice. The same stands for the features B and C – the frequency of seeing the feature C is the frequency of not seeing the feature B because they are mutually exclusive and thus all the configurations without feature B will account for the presence of the feature C if we don't put it into the lattice.

6 Model Computing and Generalization

After we have chosen a subset of the atomic features for our model, we restrict our feature lattice to the optimized lattice. Now we can compute the maximum entropy model taking the reference probabilities (which are configuration probabilities) as in equation 17. Instead of using total possible configuration space (W) as required for iterative scaling by equation 7 we restrict the configuration space to that actually observed during the lattice building. Here we have two choices. First as the configuration space we can use only the reference nodes (w) from the lattice which makes it similar to the method of Berger et al. 1996 described in section 2.1. We can also use all the nodes from the lattice (reference and hidden) as the extended configuration space (w^+). This can be seen as the union of the observed and logically implied configuration spaces which still usually will be much smaller than the total possible configuration space (W). In this case the computational load will increase proportionally to the number of hidden nodes but the model itself will be fit more accurately. Thus depending on the size of the lattice we can use either the first or the second way.

The nodes from the lattice also serve as potential constraint features to our model. However, even if the lattice is large, only a fraction of the nodes will be relevant as possible constraints for a particular joint distributions $\theta = (X, y)$ since most of the nodes will have zero or very small feature counts $\xi^{\prime X}_{\theta=(x,y)}$[8]. Thus we will consider as possible constraints for the model only those nodes whose feature frequency counts are greater than a certain threshold, e.g.: $\xi^{\prime X}_{\theta=(x,y)} > 5$. This means that we constrain only features with reliable estimates and at the same time we drastically decrease the computational load.

Initially we constrain all the nodes which satisfy the above requirement. Then for each behavior variable we run the Improved Iterative Scaling algorithm as described in section 1 and produce a

[8] $\xi^{\prime X}(\theta_k) = \sum_{\theta_i \in \theta'} f_{\theta_k}(\theta_i) * \xi^{\prime w}_{\theta_i}$

joint model with parameters $(Z, \lambda_0 ... \lambda_n)$. Then we use these joint models for the computing of a conditional model as described in section 2.1. This model will closely fit the reference distribution $\tilde{p}$ of the optimized feature lattice but usually it will be too specific and might poorly predict the unseen cases. In theory we want to constrain only some general hidden nodes so they would accurately predict the reference nodes and we hope that they will be good as well for the unseen configurations.

In order to generalize and simplify our maximum entropy model we unconstrain the most specific features, compute a new simplified maximum entropy model and if it still predicts well, we repeat the process. So our aim is to remove from the constraints as many top level nodes as possible without loosing the model fitness to the reference distribution ($\tilde{p}$) of the optimized feature lattice. The necessary condition for a node to be taking as a candidate for being removed from the constraint set is that this node shouldn't have any constrained nodes above it. There is also a natural ranking for the candidate nodes: the closer to 1 the weight (λ) of a constrained node is, the less it is important for the model. We can set a certain threshold on the weights, so all the candidate nodes whose λs differ from 1 less than this threshold will be unconstrained in one go. Therefore we don't have to use the iterative scaling for feature ranking and apply it only for linear model regression, possibly un-constraining several feature configurations (nodes) at once. This method, in fact, resembles the Backward Sequential Search (BSS) proposed in Pedersen&Bruce 1997 for decomposable models. Unlike there, however, we don't believe that models with complex overlapping feature interactions can be estimated directly from their feature distribution and use the iterative scaling algorithm instead. Another important improvement in our method is that since the generalized smaller model deviates from the previous larger model only in a small number of constraints, we use the parameters of that larger model[9] as the initial values for the iterative scaling algorithm. This proved to decrease the number of required iterations to fit the simplified model by about tenfold, which makes a tremendous saving in time.

There can be many possible criteria when to stop the generalization algorithm. The simplest one is just to set a predefined threshold on the deviation $D(\tilde{p} \,||\, p)$ of the generalized model from the reference distribution. Pedersen&Bruce 1997 suggest to use Akaike's Information Criteria (AIC) to judge the acceptability of a new model. AIC rewards good model fit and penalizes models with high complexity measured in the number of features. We adopted the stop condition suggested in Berger et al. 1996 – the maximization of the likelihood on a cross-validation set of samples which is unseen at the parameter estimation.

7 Applications of the Method

We applied the above described method of building maximum entropy models to several tasks: sentence boundary disambiguation, part-of-speech tagging and document abstracting via sentence extraction. In this section we deliberately will not go into details of training and evaluation – this will be the subject for a separate paper – but rather we will concentrate on the feasibility of the proposed method for real world applications.

Sentence boundary disambiguation has recently gained certain attention of the language engineering community. It is required for most text processing tasks such as, tagging, parsing, parallel corpora alignment etc., and, as it turned out to be, it is a non-trivial task itself. A period can act as the end of sentence or be a part of an abbreviation, but when an abbreviation is the last word in a sentence, the period denotes the end of sentence as well. The simplest "period-space-capital_letter" approach works well for simple texts but is rather unreliable for texts with many proper names

[9]instead of the uniform distribution as prescribed in the step 1 of the Improved Iterative Scaling algorithm.

and abbreviations at the end of sentence as, for instance, the Wall Street Journal (WSJ) corpus (Marcus et al.1993).

To tackle this problem we built two maximum entropy models. The first model used a lexicon of words associated with one or more categories from the set: abbreviation, proper noun, content word, closed-class word. This model employed atomic features such as the lexicon information for the words before and after the period, their capitalization and spellings. For training we collected from the WSJ corpus 51,000 samples of the form $\langle Y, F..F \rangle$ and $\langle N, F..F \rangle$, where Y stands for the end of sentence, N stands for otherwise and Fs stand for the features of the model. We built a model out of 238 most frequent atomic features which gave us the collocation lattice of 8,245 nodes in 43 minutes of processor time on SUN Ultra-1 workstation. When we applied the atomic feature selection algorithm (section 5), we in 53 minutes boiled the lattice down to 3,769 nodes. Then constraining all the nodes we compiled a maximum entropy model in about three hours and then using the constraint removal process in two hours boiled the constraint space down to 619. For the evaluation we used the same 27,294 sentences as in Palmer&Hearst 1994 and Palmer&Hearst 1997[10] which were also used by Reynar&Ratnaparkhi 1997 in the evaluation of their system. These sentences, of course, were not seen at the training phase of our model. Our model achieved 99,2% accuracy which is the highest quoted score on this test set known to the authors.

We also built a maximum entropy model to deal with unknown abbreviations, i.e. the model classifies whether or not an unknown to the lexicon word is an abbreviation. This model relies on the surface lexical features of words such as, word capitalization (C), class of the character (consonant (c), vowel (v), punctuation (p) or digit (d)) in the four last positions of the word and the length $(1, 2, 3, 4, 5, 6+)$ of the word. From the test samples we collected the frequencies of all our features and included into the initial atomic feature set only those features which appeared more than 1000 times in the positive training samples. There were 49 of such features but they had a very high level of co-occurrence and produced the empirical feature collocation lattice of 7,626 nodes. This took about 67 minutes of the processor time. Then we run the atomic feature selection algorithm and our atomic feature set in 46 minutes was boiled down to 42 atomic features and the feature collocation lattice to 3,028 nodes. Then the constraint removing algorithm boiled the constraint space down to 1,031 in two hours. The accuracy of the classification of the produced model reached 96,4% on unseen words.

Using our method we built a maximum entropy model for part-of-speech tagging. We considered as features bigram and trigram combinations together with unigrams of possible parts-of-speech for words in question. We also included into the constraint set the actual spellings of the most frequent words. We collected training samples from the Brown Corpus distributed with the Penn Treebank (Marcus et al.1993). This gave us 1,763 atomic features, but the lattice itself was not-surprisingly flat and had 38,564 nodes. It was boiled down to 16,078 in 40 hours of the processor time. We didn't specifically evaluate the model but it is about 1.2% more accurate than a bigram Hidden Markov Model which we used before.

We also built a maximum entropy model for the task of extraction of the most informative sentences for automatic document abstracting. Here we used about 120 atomic features such as sentence position, sentence length, q-phrases, etc. The initial lattice was of 28,114 nodes. After the atomic feature selection it was reduced down to 3,792 nodes. This was boiled down to 311 nodes by the constraint removal algorithm. It actually shows that there was only a very small interdependency among the features and not-surprisingly our model improved only about 1.1% over a simple Bayesian classifier achieving just under 70% precision.

[10]We want to thank David Palmer for making his test data available to us.

8 Conclusion

In this paper we presented a novel approach for building maximum entropy models. Our approach uses a feature collocation lattice and selects the atomic features without resorting to iterative scaling. After the atomic features have been selected we, using the iterative scaling, compute a fully saturated model for the maximal constraint space and then start to eliminate the most specific constraints. Since during constraint deselection at every point we have a fully fit maximum entropy model, we rank the constraints on the basis of their weights in the model. Therefore we don't have to use the iterative scaling for constraint ranking and apply it only for linear model regression. Another important improvement is that since the smaller model deviates from the previous larger model only in a small number of constraints, we use the parameters of the old model as the initial values of the parameters for the iterative scaling of the new one. This proved to decrease the number of required iterations by about tenfold. We applied the described method to several language modelling tasks and proved its feasibility for selecting and building the models with the complexity of tens of thousands constraints. A potential drawback of our approach is that we require to build a maximum entropy model for the whole observed feature-space which might not be feasible for applications with hundreds of thousands of features. So one of the directions in our future work is to find efficient ways for a decomposition of the feature lattice into non-overlapping sub-lattices which then can be handled by our method. Another avenue for further improvement is to introduce the "or" operation on the nodes of the lattice. This can provide a further generalization over the employed by the model features.

References

Berger et al. 1996 A. Berger, S. Della Pietra, V. Della Pietra, 1996. A Maximum Entropy Approach to Natural Language Processing In *Computational Linguistics* vol.22(1)

Della Pietra et al. 1995 S. Della Pietra, V.. Della Pietra, and J. Lafferty 1995. Inducing Features of Random Fields Technical report CMU-CS-95-144

Finch 1997 S. Finch 1997. Query Domains: Towards an Algebra for Collocation. TTSG Technical Report TTLNLP/97/003, Rockville, MD.

Katz 1987 S. Katz 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. In *IEEE Transactions on Acoustics, Speech and Signal Processing*, ASSP-35

Marcus et al.1993 M. Marcus, M.A. Marcinkiewicz, and B. Santorini 1993. Building a Large Annotated Corpus of English: The Penn Treebank. In *Computational Linguistics*, vol 19/2 pp.313-329

Palmer&Hearst 1994 D. D. Palmer and M. A. Hearst 1994. Adaptive Sentence Boundary Disambiguation. In *Proceedings of the Fourth ACL Conference on Applied Natural Language Processing (ANLP'94)*, ACL.

Palmer&Hearst 1997 D. D. Palmer and M. A. Hearst 1997. Adaptive Multilingual Sentence Boundary Disambiguation. In *Computational Linguistics*, ACL.

Pedersen&Bruce 1997 T. Pedersen and R. Bruce 1997. A New Supervised Learning Algorithm for Word Sense Disambiguation. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, Providence, RI.

Reynar&Ratnaparkhi 1997 J. C. Reynar and A. Ratnaparkhi 1997. A Maximum Entropy Approach to Identifying Sentence Boundaries. In *Proceedings of the Fifth ACL Conference on Applied Natural Language Processing (ANLP'97)*, Washington D.C., ACL.

Ristard 1996 E. S. Ristard 1996. Maximum Entropy Modelling Toolkit. *Documentation for Version 1.3 Beta, Draft*, available at cmp-lg/96120005

Rosenfeld 1996 R. Rosenfeld 1996. A Maximum Entropy Approach to Adaptive Statistical Language Learning. In *Computer Speech and Language*, vol.10(3) Academic Press Limited