In: *Proceedings of CoNLL-2000 and LLL-2000*, pages 1–6, Lisbon, Portugal, 2000.

Learning in Natural Language: Theory and Algorithmic Approaches*

Dan Roth
Department of Computer Science
University of Illinois at Urbana-Champaign
1304 W Springfield Ave., Urbana, IL 61801
danr@cs.uiuc.edu

Abstract

This article summarizes work on developing a learning theory account for the major learning and statistics based approaches used in natural language processing. It shows that these approaches can all be explained using a single distribution free inductive principle related to the pac model of learning. Furthermore, they all make predictions using the same simple knowledge representation – a linear representation over a common feature space. This is significant both to explaining the generalization and robustness properties of these methods and to understanding how these methods might be extended to learn from more structured, knowledge intensive examples, as part of a learning centered approach to higher level natural language inferences.

1 Introduction

Many important natural language inferences can be viewed as problems of resolving phonetic, syntactic, semantics or pragmatics ambiguities, based on properties of the surrounding context. It is generally accepted that a learning component must have a central role in resolving these context sensitive ambiguities, and a significant amount of work has been devoted in the last few years to developing learning methods for these tasks, with considerable success. Yet, our understanding of when and why learning works in this domain and how it can be used to support increasingly higher level tasks is still lacking. This article summarizes work on developing a learning theory account for the major learning approaches used in NL.

While the major statistics based methods used in NLP are typically developed with a Bayesian view in mind, the Bayesian principle cannot directly explain the success and robustness of these methods, since their probabilistic assumptions typically do not hold in the data. Instead, we provide this explanation using a single, distribution free inductive principle related to the pac model of learning. We describe the unified learning framework and show that, in addition to explaining the success and robustness of the statistics based methods, it also applies to other machine learning methods, such as rule based and memory based methods.

An important component of the view developed is the observation that most methods use the same simple knowledge representation. This is a linear representation over a new feature space – a transformation of the original instance space to a higher dimensional and more expressive space. Methods vary mostly algorithmicly, in ways they derive weights for features in this space. This is significant both to explaining the generalization properties of these methods and to developing an understanding for how and when can these methods be extended to learn from more structured, knowledge intensive examples, perhaps hierarchically. These issues are briefly discussed and we emphasize the importance of studying knowledge representation and inference in developing a learning centered approach to NL inferences.

2 Learning Frameworks

Generative probability models provide a principled way to the study of statistical classification in complex domains such as NL. It is common to assume a generative model for such data, estimate its parameters from training data and then use Bayes rule to obtain a classifier for this model. In the context of NL most classifiers are derived from probabilistic language models which estimate the probability of a sentence s using Bayes rule, and then decompose this probability into a product of conditional

* This research is supported by NSF grants IIS-9801638, SBR-9873450 and IIS-9984168.

probabilities according to the generative model.

$$\begin{aligned} Pr(s) &= Pr(w_1, w_2, \ldots w_n) = \\ &= \Pi_{i=1}^{n} Pr(w_i | w_1, \ldots w_{i-1}) = \Pi_{i=1}^{n} Pr(w_i | h_i) \end{aligned}$$

where h_i is the relevant *history* when predicting w_i, and s is any sequence of tokens, words, part-of-speech (pos) tags or other terms.

This general scheme has been used to derive classifiers for a variety of natural language applications including speech applications (Rab89), pos tagging (Kup92; Sch95), word-sense ambiguation (GCY93) and context-sensitive spelling correction (Gol95). While the use of Bayes rule is harmless, most of the work in statistical language modeling and ambiguity resolution is devoted to estimating terms of the form $Pr(w|h)$. The generative models used to estimate these terms typically make Markov or other independence assumptions. It is evident from studying language data that these assumptions are often patently false and that there are significant global dependencies both within and across sentences. For example, when using (Hidden) Markov Model (HMM) as a generative model for pos tagging, estimating the probability of a sequence of tags involves assuming that the pos tag t_i of the word w_i is independent of other words in the sentence, given the preceding tag t_{i-1}. It is not surprising therefore that this results in a poor estimate of the probability density function. However, classifiers built based on these false assumptions nevertheless seem to behave quite robustly in many cases.

A different, distribution free inductive principle that is related to the pac model of learning is the basis for the account developed here.

In an instance of the agnostic variant of pac learning (Val84; Hau92; KSS94), a learner is given data elements (x, l) that are sampled according to some fixed but arbitrary distribution D on $X \times \{0, 1\}$. X is the instance space and $l \in \{0, 1\}$ is the label[1]. D may simply reflect the distribution of the data as it occurs "in nature" (including contradictions) without assuming that the labels are generated according to some "rule". Given a sample, the goal of the learning algorithm is to eventually output a hypothesis h from some hypothesis class $\mathcal{H}$ that closely approximates the data. The true error of the hypothesis h is defined to be $error_D(h) = Pr_{(x,l)\in D}[h(x) \neq l]$, and the goal of the (agnostic) pac learner is to compute, for any distribution D, with high probability ($> 1 - \delta$), a hypothesis $h \in \mathcal{H}$ with true error no larger than $\varepsilon + inf_{h\in\mathcal{H}} error_D(h)$. In practice, one cannot compute the true error $error_D(h)$. Instead, the input to the learning algorithm is a sample $S = \{(x^i, l^i)\}_{i=1}^{m}$ of m labeled examples and the learner tries to find a hypothesis h with a small *empirical error* $error_S(h) = |\{x \in S | h(x) \neq l\}|/|S|$, and hopes that it behaves well on future examples. The hope that a classifier learned from a training set will perform well on previously unseen examples is based on the basic inductive principle underlying learning theory (Val84; Vap95) which, stated informally, guarantees that if the training and the test data are sampled from the same distribution, good performance on large enough training sample guarantees good performance on the test data (i.e., good "true" error). Moreover, the quality of the generalization is inversely proportional to the expressivity of the class $\mathcal{H}$. Equivalently, for a fixed sample size $|S|$, the quantified version of this principle (e.g. (Hau92)) indicates how much can one count on a hypothesis selected according to its performance on S. Finally, notice the underlying assumption that the training and test data are sampled from the same distribution; this framework addresses this issue. (See (GR99).)

In our discussion functions learned over the instance space X are not defined directly over the raw instances but rather over a transformation of it to a feature space. A *feature* is an indicator function $\chi : X \rightarrow \{0, 1\}$ which defines a subset of the instance space – all those elements in X which are mapped to 1 by χ. $\mathcal{X}$ denotes a class of such functions and can be viewed as a transformation of the instance space; each example $(x_1, \ldots x_n) \in X$ is mapped to an example $(\chi_1, \ldots \chi_{|\mathcal{X}|})$ in the new space. We sometimes view a feature as an indicator function over the *labeled* instance space $X \times \{0, 1\}$ and say that $\chi(x, l) = 1$ for examples $x \in \chi(X)$ with label l.

3 Explaining Probabilistic Methods

Using the abovementioned inductive principle we describe a learning theory account that explains the success and robustness of statistics based classifiers (Rot99a). A variety of meth-

[1] The model can be extended to deal with any discrete or continuous range of the labels.

ods used for learning in NL are shown to make their prediction using *Linear Statistical Queries (LSQ)* hypotheses. This is a family of linear predictors over a set of features which are directly related to the independence assumptions of the probabilistic model assumed. The success of these classification methods is then shown to be due to the combination of two factors:

- Low expressive power of the derived classifier.
- Robustness properties shared by all linear statistical queries hypotheses.

Since the hypotheses are computed over a feature space chosen so that they perform well on training data, learning theory implies that they perform well on previously unseen data, irrespective of whether the underlying probabilistic assumptions hold.

3.1 Robust Learning

This section defines a learning algorithm and a class of hypotheses with some generalization properties, that capture many probabilistic learning methods used in NLP. The learning algorithm is a *Statistical Queries(SQ)* algorithm (Kea93). An SQ algorithm can be viewed as a learning algorithm that interacts with its environment in a restricted way. Rather than viewing *examples*, the algorithm only requests the values of various *statistics* on the distribution of the examples to construct its hypothesis. (E.g. "What is the probability that a randomly chosen example (x,l) has $x_i = 0$ and $l = 1$"?)

A *statistical query* has the form $[\chi, l, \tau]$, where $\chi \in \mathcal{X}$ is a feature, $l \in \{0,1\}$ is a further (optional) restriction imposed on the query and τ is an error parameter. A call to the SQ oracle returns an estimate $\hat{P}^D_{[\chi,l,\tau]}$ of

$$P^D_{[\chi,l]} = Pr_D\{(x,i) | \chi(x) = 1 \wedge i = l\}$$

which satisfies $|\hat{P}_\chi - P_\chi| < \tau$. (We usually omit τ and/or l from this notation.) A *statistical queries algorithm* is a learning algorithm that constructs its hypothesis only using information received from an SQ oracle. An algorithm is said to use a query space $\mathcal{X}$ if it only makes queries of the form $[\chi, l, \tau]$ where $\chi \in \mathcal{X}$. An *SQ* algorithm is said to be a good learning algorithm if, with high probability, it outputs a hypothesis h with small error, using sample size that is polynomial in the relevant parameters.

Given a query $[\chi, l, \tau]$ the SQ oracle is simulated by drawing a large sample S of labeled examples (x,l) according to D and evaluating

$$Pr_S[\chi(x,l)] = |\{(x,l) : \chi(x,l) = 1|\}/|S|.$$

Chernoff bounds guarantee that the number of examples required to achieve tolerance τ with probability at least $1-\delta$ is polynomial in $1/\tau$ and $\log 1/\delta$. (See (Kea93; Dec93; AD95)).

Let $\mathcal{X}$ be a class of features and $f : \{0,1\} \rightarrow \Re$ a function that depends only on the values $\hat{P}^D_{[\chi,l]}$ for $\chi \in \mathcal{X}$. Given $x \in X$, a *Linear Statistical Queries (LSQ)* hypothesis predicts

$$l = argmax_{l \in \{0,1\}} \sum_{\chi \in \mathcal{X}} f_{[\chi,l]}(\{\hat{P}^D_{[\chi,l]}\}) \cdot \chi(x).$$

Clearly, the LSQ is a linear discriminator over the feature space $\mathcal{X}$, with coefficients f that are computed given (potentially all) the values $\hat{P}^D_{[\chi,l]}$. The definition generalizes naturally to non-binary classifiers; in this case, the discriminator between predicting l and other values is linear. A learning algorithm that outputs an LSQ hypothesis is called an *LSQ algorithm.*

Example 3.1 *The naive Bayes predictor (DH73) is derived using the assumption that given the label $l \in L$ the features' values are statistically independent. Consequently, the Bayes optimal prediction is given by:*

$$h(x) = argmax_{l \in L} \Pi_{i=1}^m Pr(x_i|l) Pr(l),$$

where $Pr(l)$ denotes the prior probability of l (the fraction of examples labeled l) and $Pr(x_i|l)$ are the conditional feature probabilities (the fraction of the examples labeled l in which the ith feature has value x_i). Therefore, we get:

Claim: The naive Bayes algorithm is an *LSQ* algorithm over a set $\mathcal{X}$ which consists of $n+1$ features: $\chi_0 \equiv 1$, $\chi_i = x_i$ for $i = 1, \ldots, n$ and where $f_{[1,l]}() = \log \hat{P}^D_{[1,l]}$ and $f_{[x_i,l]}() = \log \hat{P}^D_{[x_i,l]} / \hat{P}^D_{[1,l]}, \quad i = 1, \ldots, n.$

The observation that the LSQ hypothesis is linear over $\mathcal{X}$ yields the first generalization property of LSQ. VC theory implies that the VC dimension of the class of LSQ hypotheses is bounded above by $|\mathcal{X}|$. Moreover, if the LSQ hypothesis is *sparse* and does not make use of unobserved features in $\mathcal{X}$ (as in Ex. 3.1) it is bounded by the number of features used (Rot00). Together with the basic generalization property described above this implies:

Corollary 3.1 *For LSQ, the number of training examples required in order to maintain a specific generalization performance guarantee scales linearly with the number of features used.*

The robustness property of LSQ can be cast for the case in which the hypothesis is learned using a training set sampled according to a distribution D, but tested over a sample from D'. It still performs well as long as the distributional distance $d(D, D')$ is controlled (Rot99a; Rot00).

Theorem 3.2 *Let $\mathcal{A}$ be an $SQ(\tau, \mathcal{X})$ learning algorithm for a function class $\mathcal{G}$ over the distribution D and assume that $d(D, D') < \gamma$ (for γ inversely polynomial in τ). Then $\mathcal{A}$ is also an $SQ(\tau, \mathcal{X})$ learning algorithm for $\mathcal{G}$ over D'.*

Finally, we mention that the robustness of the algorithm to different distributions depends on the sample size and the richness of the feature class $\mathcal{X}$ plays an important role here. Therefore, for a given size sample, the use of simpler features in the LSQ representation provides better robustness. This, in turn, can be traded off with the ability to express the learned function with an LSQ over a simpler set of features.

3.2 Additional Examples

In addition to the naive Bayes (NB) classifier described above several other widely used probabilistic classifiers can be cast as LSQ hypotheses. This property is maintained even if the NB predictor is generalized in several ways, by allowing hidden variables (GR00) or by assuming a more involved independence structure around the target variable. When the structure is modeled using a general Bayesian network (since we care only about predicting a value for a single variable having observed the others) the Bayes optimal predictor is an LSQ hypothesis over features that are polynomials $\chi = \Pi x_{i_1} x_{i_2} \ldots x_{i_k}$ of degree that depends on the number of neighbors of the target variable. A specific case of great interest to NLP is that of *hidden Markov Models*. In this case there are two types of variables, state variables S and observed ones, O (Rab89). The task of predicting the value of a state variable given values of the others can be cast as an LSQ, where $\mathcal{X} \subseteq \{S, O, 1\} \times \{S, O, 1\}$, a suitably defined set of singletons and pairs of observables and state variables (Rot99a).

Finally, Maximum Entropy (ME) models (Jay82; Rat97) are also LSQ models. In this framework, *constrains* correspond to *features*; the distribution (and the induced classifier) are defined in terms of the *expected value* of the features over the training set. The induced classifier is a linear classifier whose weights are derived from these expectations; the weights are computed iteratively (DR72) since no closed form solution is known for the optimal values.

4 Learning Linear Classifiers

It was shown in (Rot98) that several other learning approaches widely used in NL work also make their predictions by utilizing a linear representation. The SNoW learning architecture (Rot98; CCRR99; MPRZ99) is explicitly presented this way, but this holds also for methods that are presented in different ways, and some effort is required to cast them this way. These include Brill's transformation based method (Bri95)[2], decision lists (Yar94) and back-off estimation (Kat87; CB95).

Moreover, even memory-based methods (ZD97; DBZ99) can be cast in a similar way (Rot99b). They can be reformulated as feature-based algorithms, with features of types that are commonly used by other features-based learning algorithms in NLP; the prediction computed by MBL can be computed by a *linear function* over this set of features.

Some other methods that have been recently used in language related applications, such as Boosting (CS99) and support vector machines are also making use of the same representation.

At a conceptual level all learning methods are therefore quite similar. They transform the original input (e.g., sentence, sentence+pos information) to a new, high dimensional, feature space, whose coordinates are typically small conjunctions (n-grams) over the original input. In this new space they search for a linear function that best separates the training data, and rely on the inductive principle mentioned to yield good behavior on future data. Viewed this way, methods are easy to compare and analyze for their suitability to NL applications and future extensions, as we sketch below.

The goal of blowing up the instance space to a high dimensional space is to increase the expressivity of the classifier so that a linear function could represent the target concepts. Within this space, probabilistic methods are the most limited since they do not actually search in the

[2]This holds only in cases in which the TBL conditions do not depend on the labels, as in Context Sensitive Spelling (MB97) and Prepositional Phrase Attachment (BR94) and not in the general case.

space of linear functions. Given the feature space they directly compute the classifier. In general, even when a simple linear function generates the training data, these methods are not guaranteed to be consistent with it (Rot99a). However, if the feature space is chosen so that they are, the robustness properties shown above become significant. Decision lists and MBL methods have advantages in their ability to represent exceptions and small areas in the feature space. MBL, by using long and very specialized conjunctions (DBZ99) and decision lists, due to their functional form - a linear function with exponentially decreasing weights - at the cost of predicting with a single feature, rather than a combination (Gol95). Learning methods that attempt to find the best linear function (relative to some loss function) are typically more flexible. Of these, we highlight here the SNoW architecture, which has some specific advantages that favor NLP-like domains.

SNoW determines the features' weights using an on-line algorithm that attempts to minimize the number of mistakes on the training data using a multiplicative weight update rule (Lit88). The weight update rule is driven by the maximum entropy principle (KW95). The main implication is that SNoW has significant advantages in sparse spaces, those in which a few of the features are actually relevant to the target concept, as is typical in NLP. In domains with these characteristics, for a given number of training examples, SNoW generalizes better than additive update methods like perceptron and its close relative SVMs (Ros58; FS98) (and in general,it has better learning curves).

Furthermore, although in SNoW the transformation to a large dimensional space needs to be done explicitly (rather than via kernel functions as is possible in perceptron and SVMs) its use of variable size examples nevertheless gives it computational advantages, due to the sparse feature space in NLP applications. It is also significant for extensions to relational domain mentioned later. Finally, SNoW is a multi-class classifier.

5 Future Research Issues

Research on learning in NLP needs to be integrated with work on knowledge representation and inference to enable studying higher level NL tasks. We mention two important directions the implications on the learning issues.

The unified view presented reveals that all methods blow up the dimensionality of the original space in essentially the same way; they generate conjunctive features over the linear structure of the sentence (i.e., n-gram like features in the word and/or pos space).

This does not seem to be expressive enough. Expressing complex concepts and relations necessary for higher level inferences will require more involved intermediate representations ("features") over the input; higher order structural and semantic properties, long term dependencies and relational predicates need to be represented. Learning will stay manageable if done in terms of these intermediate representations as done today, using functionally simple representations (perhaps cascaded).

Inductive logic programming (MDR94; Coh95) is a natural paradigm for this. However, computational limitations that include both learnability and subsumption render this approach inadequate for large scale knowledge intensive problems (KRV99; CR00).

In (CR00) we suggest an approach that addresses the generation of complex and relational intermediate representations and supports efficient learning on top of those. It allows the generation and use of *structured examples* which could encode relational information and long term functional dependencies. This is done using a construct that defines "types" of (potentially, relational) features the learning process might use. These represent infinitely many features, and are not generated explicitly; only those present in the data are generated, on the fly, as part of the learning process. Thus it yields hypotheses that are as expressive as relational learners in a scalable fashion. This approach, however, makes some requirements on the learning process. Most importantly, the learning approach needs to be able to process variable size examples. And, it has to be feature efficient in that its complexity depends mostly on the number of relevant features. This seems to favor the SNoW approach over other algorithms that learn the same representation.

Eventually, we would like to perform inferences that depend on the outcomes of several different classifiers; together these might need to coherently satisfy some constrains arising from the sequential nature of the data or task and domain specific issues. There is a need to study, along with learning and knowledge representation, inference methods that suit this framework (KR97). Work in this direction requires a consistent semantics of the learners (Val99) and will have implications on the knowledge representations and learning methods used. Preliminary work in (PR00) suggests several ways to formalize this problem and is evaluated in the context of identifying phrase structure.

References

J. A. Aslam and S. E. Decatur. Specification and simulation of statistical query algorithms for efficiency and noise tolerance. In *COLT* 1995, pages 437–446.

E. Brill and P. Resnik. A rule-based approach to prepositional phrase attachment disambiguation. In *Proc. of COLING*, 1994.

E. Brill. Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. *Computational Linguistics*, 21(4):543–565, 1995.

M. Collins and J Brooks. Prepositional phrase attachment through a backed-off model. In *WVLC* 1995.

A. Carleson, C. Cumby, J. Rosen, and D. Roth. The SNoW learning architecture. Technical Report UIUCDCS-R-99-2101, UIUC CS, May 1999.

W. Cohen. PAC-learning recursive logic programs: Efficient algorithms. *JAIR*, 2:501–539, 1995.

C. Cumby and D. Roth. Relational representations that facilitate learning. In *KR* 2000, pages 425–434.

M. Collins and Y. Singer. Unsupervised models for name entity classification. In *EMNLP-VLC'99*.

W. Daelemans, A. van den Bosch, and J. Zavrel. Forgetting exceptions is harmful in language learning. *Machine Learning*, 34(1-3):11–43, 1999.

S. E. Decatur. Statistical queries and faulty PAC oracles. In *COLT* 1993, pages 262–268.

R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley, 1973.

J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43(5):1470–1480, 1972.

Y. Freund and R. Schapire. Large margin classification using the Perceptron algorithm. In *COLT* 1998.

W. Gale, K. Church, and D. Yarowsky. A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, 26:415–439, 1993.

A. R. Golding. A Bayesian hybrid method for context-sensitive spelling correction. In *Proceedings of the 3rd workshop on very large corpora, ACL-95*, 1995.

A. R. Golding and D. Roth. A Winnow based approach to context-sensitive spelling correction. *Machine Learning*, 34(1-3):107–130, 1999.

A. Grove and D. Roth. Linear concepts and hidden variables. *Machine Learning*, 2000. To Appear.

D. Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Inform. Comput.*, 100(1):78–150, 1992.

E. T. Jaynes. On the rationale of maximum-entropy methods. *Proc. of the IEEE*, 70(9):939–952, 1982.

S. M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, speech, and Signal Processing*, 35(3):400–401, 1987.

M. Kearns. Efficient noise-tolerant learning from statistical queries. In *COLT* 1993, pages 392–401.

R. Khardon and D. Roth. Learning to reason. *Journal of the ACM*, 44(5):697–725, Sept. 1997.

R. Khardon, D. Roth, and L. G. Valiant. Relational learning for NLP using linear threshold elements. In *IJCAI* 1999, pages 911–917.

M. J. Kearns, R. E. Schapire, and L. M. Sellie. Toward efficient agnostic learning. *Machine Learning*, 17(2/3):115–142, 1994.

J. Kupiec. Robust part-of-speech tagging using a hidden Markov model. *Computer Speech and Language*, 6:225–242, 1992.

J. Kivinen and M. K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. In *STOC*, 1995.

N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

L. Mangu and E. Brill. Automatic rule acquisition for spelling correction. In *ICML* 1997, pages 734–741.

S. Muggleton and L. De Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 20:629–679, 1994.

M. Munoz, V. Punyakanok, D. Roth, and D. Zimak. A learning approach to shallow parsing. In *EMNLP-VLC'99*, pages 168–178.

V. Punyakanok and D. Roth. Inference with classifiers. Technical Report UIUCDCS-R-2000-2181, UIUC CS.

L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–285, 1989.

A. Ratnaparkhi. A linear observed time statistical parser based on maximum entropy models. In *EMNLP-97*.

F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–407, 1958.

D. Roth. Learning to resolve natural language ambiguities: A unified approach. In *AAAI'98*, pages 806–813.

D. Roth. Learning in natural language. In *IJCAI* 1999, pages 898–904.

D. Roth. Memory based learning in NLP. Technical Report UIUCDCS-R-99-2125, UIUC CS, March 1999.

D. Roth. Learning in natural language. Technical Report UIUCDCS-R-2000-2180, UIUC CS July 2000.

H. Schütze. Distributional part-of-speech tagging. In *EACL* 1995.

L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, November 1984.

L. G. Valiant. Robust logic. In *STOC* 1999.

V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995.

D. Yarowsky. Decision lists for lexical ambiguity resolution: application to accent restoration in Spanish and French. In *ACL* 1994, pages 88–95.

J. Zavrel and W. Daelemans. Memory-based learning: Using similarity for smoothing. In *ACL*, 1997.