Performance Prediction for Exponential Language Models

Stanley F. Chen
IBM T.J. Watson Research Center
P.O. Box 218, Yorktown Heights, NY 10598
`stanchen@watson.ibm.com`

Abstract

We investigate the task of performance prediction for language models belonging to the exponential family. First, we attempt to empirically discover a formula for predicting test set cross-entropy for n-gram language models. We build models over varying domains, data set sizes, and n-gram orders, and perform linear regression to see whether we can model test set performance as a simple function of training set performance and various model statistics. Remarkably, we find a simple relationship that predicts test set performance with a correlation of 0.9997. We analyze why this relationship holds and show that it holds for other exponential language models as well, including class-based models and minimum discrimination information models. Finally, we discuss how this relationship can be applied to improve language model performance.

1 Introduction

In this paper, we investigate the following question for language models belonging to the exponential family: given some training data and test data drawn from the same distribution, can we accurately predict the test set performance of a model estimated from the training data? This problem is known as *performance prediction* and is relevant for *model selection*, the task of selecting the best model from a set of candidate models given data.[1]

Let us first define some notation. Events have the form (x, y), where we attempt to predict the current word y given previous words x. We denote the training data as $\mathcal{D} = (x_1, y_1), \ldots, (x_D, y_D)$ and define $\tilde{p}(x, y) = \mathit{count}_{\mathcal{D}}(x, y)/D$ to be the empirical distribution of the training data. Similarly, we have a test set $\mathcal{D}^*$ and an associated empirical distribution $p^*(x, y)$. We take the performance of a conditional language model $p(y|x)$ to be the cross-entropy $\mathcal{H}(p^*, p)$ between the empirical test distribution p^* and the model $p(y|x)$:

$$\mathcal{H}(p^*, p) = -\sum_{x,y} p^*(x, y) \log p(y|x) \tag{1}$$

This is equivalent to the negative mean log-likelihood per event, as well as to log perplexity.

We only consider models in the exponential family. An exponential model $p_\Lambda(y|x)$ is a model with a set of *features* $\{f_1(x, y), \ldots, f_F(x, y)\}$ and equal number of parameters $\Lambda = \{\lambda_1, \ldots, \lambda_F\}$ where

$$p_\Lambda(y|x) = \frac{\exp(\sum_{i=1}^{F} \lambda_i f_i(x, y))}{Z_\Lambda(x)} \tag{2}$$

and where $Z_\Lambda(x)$ is a normalization factor.

One of the seminal methods for performance prediction is the Akaike Information Criterion (AIC) (Akaike, 1973). For a model, let $\hat{\Lambda}$ be the maximum likelihood estimate of Λ on some training data. Akaike derived the following estimate for the expected value of the test set cross-entropy $\mathcal{H}(p^*, p_{\hat{\Lambda}})$:

$$\mathcal{H}(p^*, p_{\hat{\Lambda}}) \approx \mathcal{H}(\tilde{p}, p_{\hat{\Lambda}}) + \frac{F}{D} \tag{3}$$

$\mathcal{H}(\tilde{p}, p_{\hat{\Lambda}})$ is the cross-entropy of the training set, F is the number of parameters in the model, and D is the number of events in the training data. However, maximum likelihood estimates for language models typically yield infinite cross-entropy on test data, and thus AIC behaves poorly for these domains.

In this work, instead of deriving a performance prediction relationship theoretically, we attempt to *empirically* discover a formula for predicting test performance. Initially, we consider only n-gram language models, and build models over varying domains, data set sizes, and n-gram orders. We perform linear regression to discover whether we can

[1] A long version of this paper can be found at (Chen, 2008).

model test set cross-entropy as a simple function of training set cross-entropy and other model statistics. For the 200+ n-gram models we evaluate, we find that the empirical relationship

$$\mathcal{H}(p^*, p_{\tilde{\Lambda}}) \approx \mathcal{H}(\tilde{p}, p_{\tilde{\Lambda}}) + \frac{\gamma}{D} \sum_{i=1}^{F} |\tilde{\lambda}_i| \quad (4)$$

holds with a correlation of 0.9997 where γ is a constant and where $\tilde{\Lambda} = \{\tilde{\lambda}_i\}$ are *regularized* parameter estimates; *i.e.*, rather than estimating performance for maximum likelihood models as in AIC, we do this for regularized models. In other words, test set cross-entropy can be approximated by the sum of the training set cross-entropy and the scaled sum of the magnitudes of the model parameters.

To maximize the correlation achieved by eq. (4), we find that it is necessary to use the same regularization method and regularization hyperparameters across models and that the optimal value of γ depends on the values of the hyperparameters. Consequently, we first evaluate several types of regularization and find which of these (and which hyperparameter values) work best across all domains, and use these values in all subsequent experiments. While ℓ_2^2 regularization gives the best performance reported in the literature for n-gram models, we find here that $\ell_1 + \ell_2^2$ regularization works even better.

The organization of this paper is as follows: In Section 2, we evaluate various regularization techniques for n-gram models and select the method and hyperparameter values that give the best overall performance. In Section 3, we discuss experiments to find a formula for predicting n-gram model performance, and provide an explanation for why eq. (4) works so well. In Section 4, we evaluate how well eq. (4) holds for several class-based language models and minimum discrimination information models. Finally, in Sections 5 and 6 we discuss related work and conclusions.

2 Selecting Regularization Settings

In this section, we address the issue of how to perform regularization in our later experiments. Following the terminology of Dudík and Schapire (2006), the most widely-used and effective methods for regularizing exponential models are ℓ_1 regularization (Tibshirani, 1994; Kazama and Tsujii, 2003; Goodman, 2004) and ℓ_2^2 regularization (Lau, 1994; Chen and Rosenfeld, 2000; Lebanon and Lafferty, 2001). While not as popular, another regularization scheme that has been shown to be effective is *2-norm inequality* regularization (Kazama and Tsujii, 2003) which is an instance of $\ell_1+\ell_2^2$ regularization as noted by Dudík and Schapire (2006). Under $\ell_1 + \ell_2^2$ regularization, the regularized parameter estimates $\tilde{\Lambda}$ are chosen to optimize the objective function

$$\mathcal{O}_{\ell_1+\ell_2^2}(\Lambda) = \mathcal{H}(\tilde{p}, p_\Lambda) + \frac{\alpha}{D} \sum_{i=1}^{F} |\lambda_i| + \frac{1}{2\sigma^2 D} \sum_{i=1}^{F} \lambda_i^2 \quad (5)$$

Note that ℓ_1 regularization can be considered a special case of this (by taking $\sigma = \infty$) as can ℓ_2^2 regularization (by taking $\alpha = 0$).

Here, we evaluate ℓ_1, ℓ_2^2, and $\ell_1 + \ell_2^2$ regularization for exponential n-gram models. An exponential n-gram model contains a binary feature f_ω for each n'-gram ω occurring in the training data for $n' \leq n$, where $f_\omega(x, y) = 1$ iff xy ends in ω. We would like to find the regularization method and associated hyperparameters that work best across different domains, training set sizes, and n-gram orders. As it is computationally expensive to evaluate a large number of hyperparameter settings over a large collection of models, we divide this search into two phases. First, we evaluate a large set of hyperparameters on a limited set of models to come up with a short list of candidate hyperparameters. We then evaluate these candidates on our full model set to find the best one.

We build n-gram models over data from five different sources and consider three different vocabulary sizes for one source, giving us seven "domains"

	data source	token type	range of n	training sents.	voc. size
A	RH	letter	2–7	100–75k	27
B	WSJ	POS	2–7	100–30k	45
C	WSJ	word	2–5	100–100k	300
D	WSJ	word	2–5	100–100k	3k
E	WSJ	word	2–5	100–100k	21k
F	BN	word	2–5	100–100k	84k
G	SWB	word	2–5	100–100k	19k

Table 1: Statistics of data sets. RH = Random House dictionary; WSJ = Wall Street Journal; BN = Broadcast News; SWB = Switchboard.

in total. We refer to these domains by the letters *A*–*G*; summary statistics for each domain are given in Table 1. The domains *C*–*G* consist of regular word data, while domains *A* and *B* consist of letter and part-of-speech (POS) sequences, respectively. Domains *C*–*E* differ only in vocabulary.

For each domain, we first randomize the order of sentences in that data. We partition off two development sets and an evaluation set (5000 "sentences" each in domain *A* and 2500 sentences elsewhere) and use the remaining data as training data. In this way, we assure that our training and test data are drawn from the same distribution as is assumed in our later experiments. Training set sizes in sentences are 100, 300, 1000, 3000, etc., up to the maximums given in Table 1. Building models for each training set size and n-gram order in Table 1 gives us a total of 218 models over the seven domains.

In the first phase of hyperparameter search, we choose a subset of these models (57 total) and evaluate many different values for (α, σ^2) with $\ell_1 + \ell_2^2$ regularization on each. We perform a grid search, trying each value $\alpha \in \{0.0, 0.1, 0.2, \ldots, 1.2\}$ with each value $\sigma^2 \in \{1, 1.2, 1.5, 2, 2.5, 3, 4, 5, 6, 7, 8, 10, \infty\}$ where $\sigma = \infty$ corresponds to ℓ_1 regularization and $\alpha = 0$ corresponds to ℓ_2^2 regularization. We use a variant of iterative scaling for parameter estimation. For each model and each (α, σ^2), we denote the cross-entropy of the development data as $H^m_{\alpha,\sigma}$ for the mth model, $m \in \{1, \ldots, 57\}$. Then, for each m and (α, σ^2), we can compute how much worse the settings (α, σ^2) perform with model m as compared to the best hyperparameter settings for that model:

$$\hat{H}^m_{\alpha,\sigma} = H^m_{\alpha,\sigma} - \min_{\alpha',\sigma'} H^m_{\alpha',\sigma'} \quad (6)$$

We would like to select (α, σ^2) for which $\hat{H}^m_{\alpha,\sigma}$ tends to be small; in particular, we choose (α, σ^2) that minimizes the root mean squared (RMS) error

$$\hat{H}^{\text{RMS}}_{\alpha,\sigma} = \sqrt{\frac{1}{57} \sum_{m=1}^{57} (\hat{H}^m_{\alpha,\sigma})^2} \quad (7)$$

For each of ℓ_1, ℓ_2^2, and $\ell_1 + \ell_2^2$ regularization, we retain the 6–8 best hyperparameter settings. To choose the best single hyperparameter setting from within this candidate set, we repeat the same analysis except over the full set of 218 models.

statistic	RMSE	coeff.
$\frac{1}{D}\sum_{i=1}^{F} \lvert\tilde{\lambda}_i\rvert$	0.043	0.938
$\frac{1}{D}\sum_{i:\tilde{\lambda}_i>0} \tilde{\lambda}_i$	0.044	0.939
$\frac{1}{D}\sum_{i=1}^{F} \tilde{\lambda}_i$	0.047	0.940
$\frac{1}{D}\sum_{i=1}^{F} \lvert\tilde{\lambda}_i\rvert^{\frac{4}{3}}$	0.162	0.755
$\frac{1}{D}\sum_{i=1}^{F} \lvert\tilde{\lambda}_i\rvert^{\frac{3}{2}}$	0.234	0.669
$\frac{1}{D}\sum_{i=1}^{F} \tilde{\lambda}_i^2$	0.429	0.443
$\frac{F_{\neq 0}}{D}$	0.709	1.289
$\frac{F_{\neq 0} \log D}{D}$	0.783	0.129
$\frac{F}{D}$	0.910	1.109
$\frac{F \log D}{D}$	0.952	0.112
1	1.487	1.698
$\frac{F}{D-F-1}$	2.232	-0.028
$\frac{F_{\neq 0}}{D-F_{\neq 0}-1}$	2.236	-0.023

Table 2: Root mean squared error (RMSE) in nats when predicting difference in development set and training set cross-entropy as linear function of a single statistic. The last column is the optimal coefficient for that statistic.

On the development sets, the (α, σ^2) value with the lowest squared error is (0.5, 6), and these are the hyperparameter settings we use in all later experiments unless otherwise noted. The RMS error, mean error, and maximum error for these hyperparameters on the evaluation sets are 0.011, 0.007, and 0.033 nats, respectively.[2] An error of 0.011 nats corresponds to a 1.1% difference in perplexity which is generally considered insignificant. Thus, we can achieve good performance across domains, data set sizes, and n-gram orders using a single set of hyperparameters as compared to optimizing hyperparameters separately for each model.

3 N-Gram Model Performance Prediction

Now that we have established which regularization method and hyperparameters to use, we attempt to empirically discover a simple formula for predicting the test set cross-entropy of regularized n-gram models. The basic strategy is as follows: We first build a large number of n-gram models over different domains, training set sizes, and n-gram orders. Then, we come up with a set of candidate statistics, *e.g.*, training set cross-entropy, number of features, etc., and do linear regression to try to best model test

[2] All cross-entropy values are reported in *nats*, or natural bits, equivalent to $\log_2 e$ regular bits. This will let us directly compare γ values with average discounts in Section 3.1.

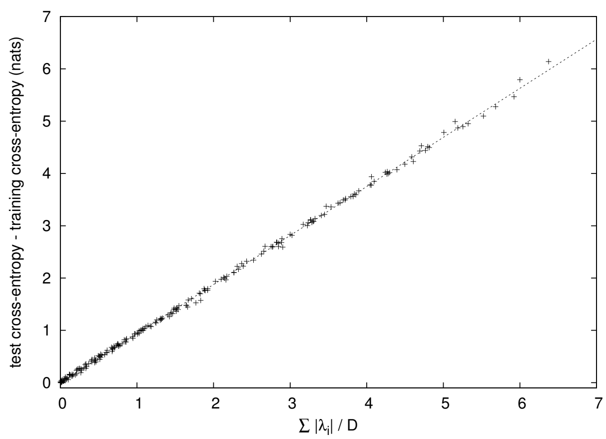

Figure 1: Graph of optimism on evaluation data *vs.* $\frac{1}{D}\sum_{i=1}^{F}|\tilde{\lambda}_i|$ for various n-gram models under $\ell_1 + \ell_2^2$ regularization, $\alpha = 0.5$ and $\sigma^2 = 6$. The line represents the predicted optimism according to eq. (9) with $\gamma = 0.938$.

set cross-entropy as a linear function of these statistics. We assume that training and test data come from the same distribution; otherwise, it would be difficult to predict test performance.

We use the same 218 n-gram models as in Section 2. For each model, we compute training set cross-entropy $\mathcal{H}(\tilde{p}, p_{\tilde{\Lambda}})$ as well as all of the statistics listed on the left in Table 2. The statistics $\frac{F}{D}$, $\frac{F}{D-F-1}$, and $\frac{F \log D}{D}$ are motivated by AIC, AIC$_c$ (Hurvich and Tsai, 1989), and the Bayesian Information Criterion (Schwarz, 1978), respectively. As features f_i with $\tilde{\lambda}_i = 0$ have no effect, instead of F we also consider using $F_{\neq 0}$, the number of features f_i with $\tilde{\lambda}_i \neq 0$. The statistics $\frac{1}{D}\sum_{i=1}^{F}|\tilde{\lambda}_i|$ and $\frac{1}{D}\sum_{i=1}^{F}\tilde{\lambda}_i^2$ are motivated by eq. (5). The statistics with fractional exponents are suggested by Figure 2. The value 1 is present to handle constant offsets.

After some initial investigation, it became clear that training set cross-entropy is a very good (partial) predictor of test set cross-entropy with coefficient 1. As there is ample theoretical support for this, instead of fitting test set performance directly, we chose to model the difference between test and training performance as a function of the remaining statistics. This difference is sometimes referred to as the *optimism* of a model:

$$\textit{optimism}(p_{\tilde{\Lambda}}) \equiv \mathcal{H}(p^*, p_{\tilde{\Lambda}}) - \mathcal{H}(\tilde{p}, p_{\tilde{\Lambda}}) \tag{8}$$

First, we attempt to model optimism as a linear function of a single statistic. For each statistic listed previously, we perform linear regression to minimize root mean squared error when predicting development set optimism. In Table 2, we display the RMSE and best coefficient for each statistic. We see that three statistics have by far the lowest error: $\frac{1}{D}\sum_{i=1}^{F}|\tilde{\lambda}_i|$, $\frac{1}{D}\sum_{i:\tilde{\lambda}_i>0}\tilde{\lambda}_i$, and $\frac{1}{D}\sum_{i=1}^{F}\tilde{\lambda}_i$. In practice, most $\tilde{\lambda}_i$ in n-gram models are positive, so these statistics tend to have similar values. We choose the best ranked of these, $\frac{1}{D}\sum_{i=1}^{F}|\tilde{\lambda}_i|$, and show in Section 3.1 why this statistic is more appealing than the others. Next, we investigate modeling optimism as a linear function of a *pair* of statistics. We find that the best RMSE for two variables (0.042) is only slightly lower than that for one (0.043), so it is doubtful that a second variable helps.

Thus, our analysis suggests that among our candidates, the best predictor of optimism is simply

$$\text{optimism} \approx \frac{\gamma}{D}\sum_{i=1}^{F}|\tilde{\lambda}_i| \tag{9}$$

where $\gamma = 0.938$, with this value being independent of domain, training set size, and n-gram order. In other words, the difference between test and training cross-entropy is a linear function of the sum of parameter magnitudes scaled per event. Substituting into eq. (8) and rearranging, we get eq. (4).

To assess the accuracy of eq. (4), we compute various statistics on our evaluation sets using the best γ from our development data, *i.e.*, $\gamma = 0.938$. In Figure 1, we graph optimism for the evaluation data against $\frac{1}{D}\sum_{i=1}^{F}|\tilde{\lambda}_i|$ for each of our models; we see that the linear correlation is very good. The correlation between the actual and predicted cross-entropy on the evaluation data is 0.9997; the mean absolute prediction error is 0.030 nats; the RMSE is 0.043 nats; and the maximum absolute error is 0.166 nats. Thus, on average we can predict test performance to within 3% in perplexity, though in the worst case we may be off by as much as 18%.[3]

[3]The sampling variation in our test set selection limits the measured accuracy of our performance prediction. To give some idea of the size of this effect, we randomly selected 100 test sets in domain *D* of 2500 sentences each (as in our other experiments). We evaluated their cross-entropies using models trained on 100, 1k, 10k, and 100k sentences. The empiri-

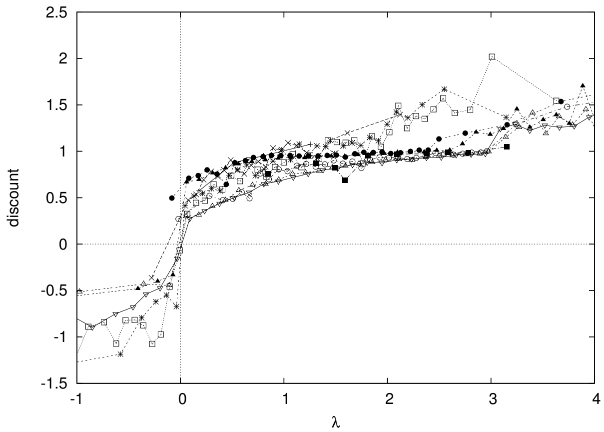

Figure 2: Smoothed graph of discount versus $\tilde{\lambda}_i$ for all features in ten different models built on domains *A* and *E*. Each smoothed point represents the average of at least 512 raw data points.

If we compute the prediction error of eq. (4) over the same models except using ℓ_1 or ℓ_2^2 regularization (with the best corresponding hyperparameter values found in Section 2), the prediction RMSE is 0.054 and 0.139 nats, respectively. Thus, we find that choosing hyperparameters carefully in Section 2 was important in doing well in performance prediction. While hyperparameters were chosen to optimize test performance rather than prediction accuracy, we find that the chosen hyperparameters are favorable for the latter task as well.

3.1 Why Does Prediction Work So Well?

The correlation in Figure 1 is remarkably high, and thus it begs for an explanation. First, let us express the difference in test and training cross-entropy for a model in terms of its parameters Λ. Substituting eq. (2) into eq. (1), we get

$$\mathcal{H}(p^*, p_\Lambda) = -\sum_{i=1}^{F} \lambda_i E_{p^*}[f_i] + \sum_x p^*(x) \log Z_\Lambda(x) \tag{10}$$

where $E_{p^*}[f_i] = \sum_{x,y} p^*(x,y) f_i(x,y)$. Then, we can express the difference in test and training performance as

$$\mathcal{H}(p^*, p_\Lambda) - \mathcal{H}(\tilde{p}, p_\Lambda) = \textstyle\sum_{i=1}^{F} \lambda_i (E_{\tilde{p}}[f_i] - E_{p^*}[f_i]) + \sum_x (p^*(x) - \tilde{p}(x)) \log Z_\Lambda(x) \tag{11}$$

cal standard deviation across test sets was found to be 0.0123, 0.0144, 0.0167, and 0.0174 nats, respectively. This effect can be mitigated by simply using larger test sets.

Ignoring the last term on the right, we see that optimism for exponential models is a linear function of the λ_i's with coefficients $E_{\tilde{p}}[f_i] - E_{p^*}[f_i]$.

Then, we can ask what $E_{\tilde{p}}[f_i] - E_{p^*}[f_i]$ values would let us satisfy eq. (4). Consider the relationship

$$(E_{\tilde{p}}[f_i] - E_{p^*}[f_i]) \times D \approx \gamma \,\mathrm{sgn}\, \tilde{\lambda}_i \tag{12}$$

If we substitute this into eq. (11) and ignore the last term on the right again, this gives us exactly eq. (4). We refer to the value $(E_{\tilde{p}}[f_i] - E_{p^*}[f_i]) \times D$ as the *discount* of a feature. It can be thought of as representing how many times less the feature occurs in the test data as opposed to the training data, if the test data were normalized to be the same size as the training data. Discounts for n-grams have been studied extensively, *e.g.*, (Good, 1953; Church and Gale, 1991; Chen and Goodman, 1998), and tend not to vary much across training set sizes.

We can check how well eq. (12) holds for actual regularized n-gram models. We construct a total of ten n-gram models on domains *A* and *E*. We build four letter 5-gram models on domain *A* on training sets ranging in size from 100 words to 30k words, and six models (either trigram or 5-gram) on domain *E* on training sets ranging from 100 sentences to 30k sentences. We create large development test sets (45k words for domain *A* and 70k sentences for domain *E*) to better estimate $E_{p^*}[f_i]$.

Because graphs of discounts as a function of $\tilde{\lambda}_i$ are very noisy, we smooth the data before plotting. We partition data points into buckets containing at least 512 points. We average all of the points in each bucket to get a "smoothed" data point, and plot this single point for each bucket. In Figure 2, we plot smoothed discounts as a function of $\tilde{\lambda}_i$ over the range $\tilde{\lambda}_i \in [-1, 4]$ for all ten models.

We see that eq. (12) holds at a very rough level over the $\tilde{\lambda}_i$ range displayed. If we examine how much different ranges of $\tilde{\lambda}_i$ contribute to the overall value of $\sum_{i=1}^{F} \tilde{\lambda}_i (E_{\tilde{p}}[f_i] - E_{p^*}[f_i])$, we find that the great majority of the mass (90–95%+) is concentrated in the range $\tilde{\lambda}_i \in [0, 4]$ for all ten models under consideration. Thus, to a first approximation, the reason that eq. (4) holds with $\gamma = 0.938$ is because on average, feature expectations have a discount of about this value for $\tilde{\lambda}_i$ in this range.[4]

[4]This analysis provides some insight as to when eq. (4)

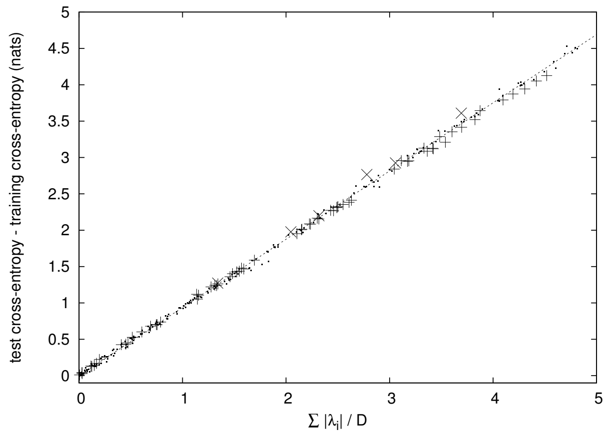

Figure 3: Graph of optimism on evaluation data *vs.* $\frac{1}{D}\sum_{i=1}^{F}|\tilde{\lambda}_i|$ for various models. The '+' marks correspond to models **S**, **M**, and **L** over different training set sizes, n-gram orders, and numbers of classes. The '$\times$' marks correspond to MDI models over different n-gram orders and in-domain training set sizes. The line and small points are taken from Figure 1.

Due to space considerations, we only summarize our other findings; a longer discussion is provided in (Chen, 2008). We find that the absolute error in cross-entropy tends to be quite small across models for several reasons. For non-sparse models, there is significant variation in average discounts, but because $\frac{1}{D}\sum_{i=1}^{F}|\tilde{\lambda}_i|$ is low, the overall error is low. In contrast, sparse models are dominated by single-count n-grams with features whose average discount is quite close to $\gamma = 0.938$. Finally, the last term on the right in eq. (11) also plays a small but significant role in keeping the prediction error low.

4 Other Exponential Language Models

In (Chen, 2009), we show how eq. (4) can be used to motivate a novel class-based language model and a regularized version of minimum discrimination information (MDI) models (Della Pietra et al., 1992). In this section, we analyze whether in addition to word n-gram models, eq. (4) holds for these other exponential language models as well.

won't hold. For example, if a feature function f_i is doubled, its expectations and discount will also double. Thus, eq. (4) won't hold in general for models with continuous feature values, as average discounts may vary widely.

4.1 Class-Based Language Models

We assume a word w is always mapped to the same class $c(w)$. For a sentence $w_1 \cdots w_l$, we have

$$p(w_1 \cdots w_l) = \prod_{j=1}^{l+1} p(c_j|c_1 \cdots c_{j-1}, w_1 \cdots w_{j-1}) \times \prod_{j=1}^{l} p(w_j|c_1 \cdots c_j, w_1 \cdots w_{j-1}) \quad (13)$$

where $c_j = c(w_j)$ and where c_{l+1} is an end-of-sentence token. We use the notation $p_{\text{ng}}(y|\omega)$ to denote an exponential n-gram model as defined in Section 2, where we have features for each suffix of each ωy occurring in the training set. We use the notation $p_{\text{ng}}(y|\omega_1, \omega_2)$ to denote a model containing all features in the models $p_{\text{ng}}(y|\omega_1)$ and $p_{\text{ng}}(y|\omega_2)$.

We consider three class models, models **S**, **M**, and **L**, defined as

$$p_S(c_j|c_1 \cdots c_{j-1}, w_1 \cdots w_{j-1}) = p_{\text{ng}}(c_j|c_{j-2}c_{j-1})$$
$$p_S(w_j|c_1 \cdots c_j, w_1 \cdots w_{j-1}) = p_{\text{ng}}(w_j|c_j)$$

$$p_M(c_j|c_1 \cdots c_{j-1}, w_1 \cdots w_{j-1}) = p_{\text{ng}}(c_j|c_{j-2}c_{j-1}, w_{j-2}w_{j-1})$$
$$p_M(w_j|c_1 \cdots c_j, w_1 \cdots w_{j-1}) = p_{\text{ng}}(w_j|w_{j-2}w_{j-1}c_j)$$

$$p_L(c_j|c_1 \cdots c_{j-1}, w_1 \cdots w_{j-1}) = p_{\text{ng}}(c_j|w_{j-2}c_{j-2}w_{j-1}c_{j-1})$$
$$p_L(w_j|c_1 \cdots c_j, w_1 \cdots w_{j-1}) = p_{\text{ng}}(w_j|w_{j-2}c_{j-2}w_{j-1}c_{j-1}c_j)$$

Model **S** is an exponential version of the class-based n-gram model from (Brown et al., 1992); model **M** is a novel model introduced in (Chen, 2009); and model **L** is an exponential version of the model *indexpredict* from (Goodman, 2001).

To evaluate whether eq. (4) can accurately predict test performance for these class-based models, we use the WSJ data and vocabulary from domain *E* and consider training set sizes of 1k, 10k, 100k, and 900k sentences. We create three different word classings containing 50, 150, and 500 classes using the algorithm of Brown et al. (1992) on the largest training set. For each training set and number of classes, we build both 3-gram and 4-gram versions of each of our three class models.

In Figure 3, we plot optimism (*i.e.*, test minus training cross-entropy) versus $\frac{1}{D}\sum_{i=1}^{F}|\tilde{\lambda}_i|$ for these models (66 in total) on our WSJ evaluation set. The '+' marks correspond to our class n-gram models, while the small points replicate the points for word n-gram models from Figure 1. Remarkably, eq. (4) appears to accurately predict performance for our class n-gram models using the same $\gamma = 0.938$ value found for word n-gram models. The mean absolute prediction error is 0.029 nats, comparable to that found for word n-gram models.

It is interesting that eq. (4) works for class-based models despite their being composed of two submodels, one for word prediction and one for class prediction. However, taking the log of eq. (13), we note that the cross-entropy of text can be expressed as the sum of the cross-entropy of its word tokens and the cross-entropy of its class tokens. It would not be surprising if eq. (4) holds separately for the class prediction model predicting class data and the word prediction model predicting word data, since all of these component models are basically n-gram models. Summing, this explains why eq. (4) holds for the whole class model.

4.2 Models with Prior Distributions

Minimum discrimination information models (Della Pietra et al., 1992) are exponential models with a *prior* distribution $q(y|x)$:

$$p_\Lambda(y|x) = q(y|x)\frac{\exp(\sum_{i=1}^{F} \lambda_i f_i(x,y))}{Z_\Lambda(x)} \tag{14}$$

The central issue in performance prediction for MDI models is whether $q(y|x)$ needs to be accounted for. That is, if we assume q is an exponential model, should its parameters λ_i^q be included in the sum in eq. (4)? From eq. (11), we note that if $E_{\tilde{p}}[f_i] - E_{p^*}[f_i] = 0$ for a feature f_i, then the feature does not affect the difference between test and training cross-entropy (ignoring its impact on the last term). By assumption, the training and test set for p come from the same distribution while q is derived from an independent data set. It follows that we expect $E_{\tilde{p}}[f_i^q] - E_{p^*}[f_i^q]$ to be zero for features in q, and we should ignore q when applying eq. (4).

To evaluate whether eq. (4) holds for MDI models, we use the same WSJ training and evaluation sets from domain *E* as in Section 4.1. We consider three different training set sizes: 1k, 10k, and 100k sentences. To train q, we use the 100k sentence BN training set from domain *F*. We build both trigram and 4-gram versions of each model.

In Figure 3, we plot test minus training cross-entropy versus $\frac{1}{D}\sum_{i=1}^{F}|\tilde{\lambda}_i|$ for these models on our WSJ evaluation set; the '$\times$' marks correspond to the MDI models. As expected, eq. (4) appears to work quite well for MDI models using the same $\gamma = 0.938$ value as before; the mean absolute prediction error is 0.077 nats.

5 Related Work

We group existing performance prediction methods into two categories: *non-data-splitting* methods and *data-splitting* methods. In non-data-splitting methods, test performance is directly estimated from training set performance and/or other statistics of a model. Data-splitting methods involve partitioning training data into a truncated training set and a surrogate test set and using surrogate test set performance to estimate true performance.

The most popular non-data-splitting methods for predicting test set cross-entropy (or likelihood) are AIC and variants such as AIC$_c$, quasi-AIC (QAIC), and QAIC$_c$ (Akaike, 1973; Hurvich and Tsai, 1989; Lebreton et al., 1992). In Section 3, we considered performance prediction formulae with the same form as AIC and AIC$_c$ (except using regularized parameter estimates), and neither performed as well as eq. (4); *e.g.*, see Table 2.

There are many techniques for bounding test set classification error including the Occam's Razor bound (Blumer et al., 1987; McAllester, 1999), PAC-Bayes bound (McAllester, 1999), and the sample compression bound (Littlestone and Warmuth, 1986; Floyd and Warmuth, 1995). These methods derive theoretical guarantees that the true error rate of a classifier will be below (or above) some value with a certain probability. Langford (2005) evaluates these techniques over many data sets; while the bounds can sometimes be fairly tight, in many data sets the bounds are quite loose.

When learning an element from a set of target classifiers, the Vapnik-Chervonenkis (VC) dimension of the set can be used to bound the true error rate relative to the training error rate with some probability (Vapnik, 1998); this technique has been used to compute error bounds for many types of classifiers. For example, Bartlett (1998) shows that for a neural network with small weights and small training set squared error, the true error depends on the size of its weights rather than the number of weights; this finding is similar in spirit to eq. (4).

In practice, the most accurate methods for performance prediction in many contexts are data-splitting methods (Guyon et al., 2006). These techniques include the hold-out method; leave-one-out and k-fold cross-validation; and bootstrapping (Allen, 1974; Stone, 1974; Geisser, 1975; Craven and Wahba, 1979; Efron, 1983). However, unlike non-data-splitting methods, these methods do not lend themselves well to providing insight into model design as discussed in Section 6.

6 Discussion

We show that for several types of exponential language models, it is possible to accurately predict the cross-entropy of test data using the simple relationship given in eq. (4). When using $\ell_1 + \ell_2^2$ regularization with $(\alpha = 0.5, \sigma^2 = 6)$, the value $\gamma = 0.938$ works well across varying model types, domains, vocabulary sizes, training set sizes, and n-gram orders, yielding a mean absolute error of about 0.03 nats (3% in perplexity). We evaluate ~300 language models in total, including word and class n-gram models and n-gram models with prior distributions.

While there has been a great deal of work in performance prediction, the vast majority of work on non-data-splitting methods has focused on finding theoretically-motivated approximations or probabilistic bounds on test performance. In contrast, we developed eq. (4) on a purely empirical basis, and there has been little, if any, existing work that has shown comparable performance prediction accuracy over such a large number of models and data sets. In addition, there has been little, if any, previous work on performance prediction for language modeling.[5]

While eq. (4) performs well as compared to other non-data-splitting methods for performance prediction, the prediction error can be several percent in perplexity, which means we cannot reliably rank models that are close in quality. In addition, in speech recognition and many other applications, an external test set is typically provided, which means we can measure test set performance directly. Thus, in practice, eq. (4) is not terribly useful for the task of model selection; instead, what eq. (4) gives us is insight into *model design*. That is, instead of selecting between candidate models *once they have been built* as in model selection, it is desirable to be able to select between models at the *model design* stage. Being able to intelligently compare models (without implementation) requires that we know which aspects of a model impact test performance, and this is exactly what eq. (4) tells us.

Intuitively, simpler models should perform better on test data given equivalent training performance, and model structure (as opposed to parameter values) is an important aspect of the complexity of a model. Accordingly, there have been many methods for model selection that measure the size of a model in terms of the number of features or parameters in the model, *e.g.*, (Akaike, 1973; Rissanen, 1978; Schwarz, 1978). Surprisingly, for exponential language models, the number of model parameters seems to matter not at all; all that matters are the magnitudes of the parameter values. Consequently, one can improve such models by adding features (or a prior model) that reduce parameter values while maintaining training performance.

In (Chen, 2009), we show how these ideas can be used to motivate heuristics for improving the performance of existing language models, and use these heuristics to develop a novel class-based model and a regularized version of MDI models that outperform comparable methods in both perplexity and speech recognition word-error rate on WSJ data. In addition, we show how the tradeoff between training set performance and model size impacts aspects of language modeling as diverse as backoff n-gram features, class-based models, and domain adaptation. In sum, eq. (4) provides a new and valuable framework for characterizing, analyzing, and designing statistical models.

Acknowledgements

We thank Bhuvana Ramabhadran and the anonymous reviewers for their comments on this and earlier versions of the paper.

[5]Here, we refer to predicting test set performance from *training set* performance and other model statistics. However, there has been a good deal of work on predicting speech recognition word-error rate from *test set* perplexity and other statistics, *e.g.*, (Klakow and Peters, 2002).

References

Hirotugu Akaike. 1973. Information theory and an extension of the maximum likelihood principle. In *Sec-*

ond Intl. Symp. on Information Theory, pp. 267–281.

David M. Allen. 1974. The relationship between variable selection and data augmentation and a method for prediction. *Technometrics*, 16(1):125–127.

Peter L. Bartlett. 1998. The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. *IEEE Trans. on Information Theory*, 44(2):525–536.

Alselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. 1987. Occam's razor. *Information Processing Letters*, 24(6):377–380.

Peter F. Brown, Vincent J. Della Pietra, Peter V. deSouza, Jennifer C. Lai, and Robert L. Mercer. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479, December.

Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. Tech. Report TR-10-98, Harvard Univ.

Stanley F. Chen and Ronald Rosenfeld. 2000. A survey of smoothing techniques for maximum entropy models. *IEEE Trans. Speech and Aud. Proc.*, 8(1):37–50.

Stanley F. Chen. 2008. Performance prediction for exponential language models. Tech. Report RC 24671, IBM Research Division, October.

Stanley F. Chen. 2009. Shrinking exponential language models. In *Proc. of HLT-NAACL*.

Kenneth W. Church and William A. Gale. 1991. A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams. *Computer Speech and Language*, 5:19–54.

Peter Craven and Grace Wahba. 1979. Smoothing noisy data with spline functions: estimating the correct degree of smoothing by the method of generalized cross-validation. *Numerische Mathematik*, 31:377–403.

Stephen Della Pietra, Vincent Della Pietra, Robert L. Mercer, and Salim Roukos. 1992. Adaptive language modeling using minimum discriminant estimation. In *Proc. Speech and Natural Lang. DARPA Workshop*.

Miroslav Dudík and Robert E. Schapire. 2006. Maximum entropy distribution estimation with generalized regularization. In *Proc. of COLT*.

Bradley Efron. 1983. Estimating the error rate of a prediction rule: Improvement on cross-validation. *J. of the American Statistical Assoc.*, 78(382):316–331.

Sally Floyd and Manfred Warmuth. 1995. Sample compression, learnability, and the Vapnik-Chervonenkis dimension. *Machine Learning*, 21(3):269–304.

Seymour Geisser. 1975. The predictive sample reuse method with applications. *J. of the American Statistical Assoc.*, 70:320–328.

I.J. Good. 1953. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3 and 4):237–264.

Joshua T. Goodman. 2001. A bit of progress in language modeling. MSR-TR-2001-72, Microsoft Research.

Joshua Goodman. 2004. Exponential priors for maximum entropy models. In *Proc. of NAACL*.

Isabelle Guyon, Amir Saffari, Gideon Dror, and Joachim Buhmann. 2006. Performance prediction challenge. In *Proc. of Intl. Conference on Neural Networks (IJCNN06), IEEE World Congress on Computational Intelligence (WCCI06)*, pp. 2958–2965, July.

Clifford M. Hurvich and Chih-Ling Tsai. 1989. Regression and time series model selection in small samples. *Biometrika*, 76(2):297–307, June.

Jun'ichi Kazama and Jun'ichi Tsujii. 2003. Evaluation and extension of maximum entropy models with inequality constraints. In *Proc. of EMNLP*, pp. 137–144.

Dietrich Klakow and Jochen Peters. 2002. Testing the correlation of word error rate and perplexity. *Speech Communications*, 38(1):19–28.

John Langford. 2005. Tutorial on practical prediction theory for classification. *J. of Machine Learning Research*, 6:273–306.

Raymond Lau. 1994. Adaptive statistical language modelling. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA.

Guy Lebanon and John Lafferty. 2001. Boosting and maximum likelihood for exponential models. Tech. Report CMU-CS-01-144, Carnegie Mellon Univ.

Jean-Dominique Lebreton, Kenneth P. Burnham, Jean Clobert, and David R. Anderson. 1992. Modeling survival and testing biological hypotheses using marked animals: a unified approach with case studies. *Ecological Monographs*, 62:67–118.

Nick Littlestone and Manfred K. Warmuth. 1986. Relating data compression and learnability. Tech. report, Univ. of California, Santa Cruz.

David A. McAllester. 1999. PAC-Bayesian model averaging. In *Proc. of COLT*, pp. 164–170.

Jorma Rissanen. 1978. Modeling by the shortest data description. *Automatica*, 14:465–471.

Gideon Schwarz. 1978. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464.

M. Stone. 1974. Cross-validatory choice and assessment of statistical predictions. *J. of the Royal Statistical Society B*, 36:111–147.

Robert Tibshirani. 1994. Regression shrinkage and selection via the lasso. Tech. report, Univ. of Toronto.

Vladimir N. Vapnik. 1998. *Statistical Learning Theory*. John Wiley, New York.