COMPACT KERNEL MODELS FOR ACOUSTIC MODELING VIA RANDOM FEATURE SELECTION

Avner May[⋆] *Michael Collins*[⋆1] *Daniel Hsu*[⋆] *Brian Kingsbury*[†]

[⋆] Department of Computer Science, Columbia University, New York, NY 10025, USA
[†]IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA

ABSTRACT

A simple but effective method is proposed for learning compact random feature models that approximate non-linear kernel methods, in the context of acoustic modeling. The method is able to explore a large number of non-linear features while maintaining a compact model via feature selection more efficiently than existing approaches. For certain kernels, this random feature selection may be regarded as a means of non-linear feature selection at the level of the raw input features, which motivates additional methods for computational improvements. An empirical evaluation demonstrates the effectiveness of the proposed method relative to the natural baseline method for kernel approximation.[2]

Index Terms— Kernel Methods, Acoustic Modeling, Feature Selection, Logistic Regression, Neural Networks.

1. INTRODUCTION

Kernel methods are statistically effective methods for non-linear learning [1], but they are computationally difficult to use with large data sets. Indeed, the $\Theta(n^2)$ size of the kernel matrix is a bottleneck in training with data sets of size n, and the typical $\Theta(n)$ size of the resulting models [2] makes them prohibitive for deployment.

Much recent effort has been devoted to the development of approximations to kernel methods, primarily via the Nyström approximation [3] or via random feature expansion [e.g., 4, 5]. These methods yield explicit feature representations on which linear learning methods can provide good approximations to the original non-linear kernel method. However, these approximations may be rather coarse, and theoretical analysis suggest that the size of the feature representations may also need to grow linearly with n for these methods to be effective. These limitations of kernel methods and their approximations have hindered their application in many large-scale settings.

In this work, we propose a simple but effective method for learning *compact* random feature models that approximate non-linear kernel methods. We iteratively select features from large pools of random features, using learned weights in the selection criterion. This has two clear benefits: (i) the subsequent training on the selected features is considerably faster than training on the entire pool of random features, and (ii) the resulting model is also much smaller. For certain kernels, this feature selection approach—which is applied at the level of the random features—can be regarded as a non-linear method for feature selection at the level of the raw input features, and we use this observation to motivate additional speed-ups.

We conduct an empirical evaluation of this proposed method on two large-scale acoustic modeling datasets, and demonstrate large improvements over the natural baseline of using random features *without* feature selection. Given that deep neural networks (DNNs) are the state-of-the-art method on these acoustic modeling problems, we also compare our performance to that of DNNs. We show that our method is generally able to match or beat the DNN in terms of test cross-entropy, but lags behind on test token error rate (TER)[3]. This leaves open an interesting question: how is it that these kernel methods match the DNNs on cross-entropy, but not on TER?

2. RELATED WORK

Many recent works have been developed to speed-up the random features approach to kernel method approximation. One line of work attempts to reduce the time (and memory) needed to compute the random feature expansions by imposing structure on the random projection matrix [6, 7]. It is also possible to use doubly-stochastic methods to speed-up stochastic gradient training of models based on the random features [8]. Neither of these speed-ups, however, provide a reduction in the number of random features required for good performance, which is the primary aim in our present work.

A recent method that does share our primary aim is the Sparse Random Features algorithm of Yen et. al. [9]. This algorithm is a coordinate descent method for smooth convex optimization problems in the (infinite) space of non-linear features: each step involves solving a batch ℓ_1-regularized convex optimization problem over randomly chosen non-linear features (note that a natural extension of this method to multi-class problems is to use mixed norms such as ℓ_1/ℓ_2). Here, the ℓ_1-regularization may cause the learned solution to only depend on a smaller number of such features. A drawback of this approach is the computational burden of fully solving many batch optimization problems, which is prohibitive for large data sets. In our attempts to implement an online variant of this method with SGD, using FOBOS [10] and ℓ_1/ℓ_2-regularization for the multi-class setting, we observed that very strong regularization was required to obtain any intermediate sparsity, which in turn severely hurt prediction performance. Effectively, the regularization was so strong that it made the learning meaningless, and the selected fea-

[1]Currently on leave at Google Inc. New York.

[2]This research is supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense U.S. Army Research Laboratory (DoD / ARL) contract number W911NF-12-C-0012. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government.

[3]For our Cantonese dataset, 'token error rate' corresponds to 'character error rate'. For our Bengali dataset, 'token error rate' corresponds to 'word error rate'

tures were basically random. Our approach for selecting random features is less computationally intensive, and is more direct at ensuring sparsity than ℓ_1-regularization.

Feature selection in general is well-studied in machine learning and statistics [e.g., 11, 12]. A related problem in neural network literature is network pruning [13, 14]. Our aim here is to identify a method of feature selection that is effective and efficient with random features for producing compact models.

One application-specific aim of this work is to evaluate kernel methods in acoustic modeling for speech recognition, where the scale of modern speech data sets has precluded the use of standard kernel methods. Some recent evaluations of random features have been conducted on the TIMIT data set [15], which is rather small compared to more recent speech data sets. Our evaluations tackle much larger-scale and more challenging acoustic modeling problems, where the state-of-the-art methods are deep neural networks [16, 17].

3. BACKGROUND

Our goal is to learn a multi-class prediction function $f\colon \mathcal{X} \to \mathcal{Y}$ from a collection of n labeled examples $S \in (\mathcal{X} \times \mathcal{Y})^n$, where $\mathcal{Y} = \{1, 2, \ldots, C\}$, and $x \in \mathcal{X} \subseteq \mathbb{R}^d$.

3.1. Linear and non-linear models

In linear models, the prediction function f is represented using a weight matrix $W = [W_1|W_2|\cdots|W_C] \in \mathbb{R}^{d\times C}$, where the prediction on input x is $f(x) = \arg\max_{c\in[C]}\langle W_c, x\rangle$. In this work, we consider simple non-linear models based on non-linear feature expansions $z\colon \mathbb{R}^d \to \mathbb{R}^D$, and learn a linear model on top of these features. We learn the weight matrix W by minimizing the empirical risk $R_S(W) = \frac{1}{|S|}\sum_{(x,y)\in S}\ell(W;(x,y))$. In our work, we use the logistic loss function, $\ell(W;(x,y)) = -\langle W_y, z(x)\rangle + \ln(\sum_{c=1}^{C}\exp(\langle W_c, z(x)\rangle)) = -\log(p(y|x;W))$, and use stochastic gradient descent (SGD) [e.g., 18, 19] to approximately minimize the empirical R_S.

3.2. Kernel methods and random features

Kernel methods are based on symmetric positive-definite kernel functions $K\colon \mathcal{X}\times\mathcal{X} \to \mathbb{R}$. The kernel value $K(x, x')$ can be regarded as an inner product $K(x, x') = \langle\phi(x), \phi(x')\rangle_{\mathcal{H}}$ between $\phi(x)$ and $\phi(x')$ for some feature expansion $\phi\colon \mathbb{R}^d \to \mathcal{H}$ to a feature space $\mathcal{H}$, where $\mathcal{H}$ can be infinite-dimensional. Standard kernel methods employ a trick to avoid ever computing $\phi(x)$ explicitly, but this generally comes at the cost of computing all kernel values $K(x, x')$ for every pair $\{x, x'\}$ in the training data set S, which can be prohibitive for large $n = |S|$.

The random features approximation seeks an explicit feature map $z\colon \mathbb{R}^d \to \mathbb{R}^D$ such that $\langle z(x), z(x')\rangle \approx K(x, x')$. In this work, we focus on *shift-invariant kernels*—i.e., kernels K that satisfy $K(x, x') = K(x - x', 0)$ for all $x \in \mathcal{X}$. As demonstrated in [4], Bochner's theorem implies that for shift-invariant kernels, the following explicit feature map satisfies the above kernel approximation property:

$$z(x)_j = \sqrt{\frac{2}{D}}\cos\left(\langle\theta^{(j)}, x\rangle + b^{(j)}\right)$$

for $\theta^{(j)}$ drawn from the Fourier transform of the function $\delta \mapsto K(\delta, 0)$, and for $b^{(j)}$ drawn uniformly at random from $[0, 2\pi]$.

Algorithm 1 Random feature selection

input Target number of random features D, data subset size R, selection schedule $0 = s_0 < s_1 < \cdots < s_T = D$.

1: **initialize** feature pool $P := \emptyset$.
2: **for** $t = 1, 2, \ldots, T$ **do**
3: Generate $D - s_{t-1}$ new random features, and add them to P.
4: Learn weights $W \in \mathbb{R}^{P\times C}$ over the D features in P using a single pass of SGD over R randomly selected training examples.
5: Select s_t features $j \in P$ for which $\sum_{c=1}^{C}(W_{j,c})^2$ are largest; discard the remaining $D - s_t$.
6: **end for**
7: **return** Final collection of D random features P.

3.3. Acoustic modeling

A basic acoustic model provides a conditional probability distribution $p(y|x)$ over C possible acoustic states, conditioned on a short acoustic frame x encoded in some raw feature representation. The acoustic states correspond to context-dependent phoneme states, and in modern speech recognition systems, the number of such states is of the order 10^3 to 10^4. The acoustic model is used within probabilistic systems for decoding speech signals into word sequences.

4. RANDOM FEATURE SELECTION

4.1. Feature selection method

Our proposed random feature selection method, shown in Algorithm 1, is based on a general iterative framework. In each iteration, random features are added to the pool of features, and a subset of them are selected, while the rest are discarded. The selection criterion is based on a feature's SGD-learned weights.

While this feature selection method is rather simplistic, it has the following advantages. The overall computational cost is rather mild, as it requires just T passes through subsets of the data of size R (equivalent to $\approx TR/n$ full SGD epochs). In fact, in our experiments, we find it sufficient to use $R = O(D)$. Note that this is less computationally demanding than fully solving an ℓ_1-regularized optimization problem, as in the Sparse Random Features method of [9]. Moreover, the method is able to explore a large number of non-linear features, while maintaining a compact model. If $s_t = Dt/T$, then the learning algorithm is exposed to roughly $DT/2$ random features throughout the feature selection process. We show in Section 5 that this empirically increases the predictive quality of selected non-linear features.

4.2. The Laplacian kernel and sparse non-linear combinations

Recall that for the Laplacian kernel, the sampling distribution used in random Fourier features is the multivariate Cauchy density $p(\theta) \propto \prod_{i=1}^{d}(1 + \theta_i^2)^{-1}$ (we set $\sigma = 1$ for simplicity). If $\theta = (\theta_1, \theta_2, \ldots, \theta_d) \sim p$, then each θ_i has a two-sided fat tail distribution, and hence a draw θ will typically contain some entries much larger than the rest.

This property of the sampling distribution implies that many of the random features generated in this way will each effectively concentrate on a few of the raw input features. We can thus regard each such random feature as being a non-linear combination of a small number of the original input features. Thus, the feature selection

method for selecting random features is effectively picking out useful non-linear interactions between small sets of raw input features.

We can also directly construct sparse non-linear combinations of the input features. Instead of relying on the properties of the Cauchy distribution, we can actually choose a small number k of coordinates in $F \subseteq \{1, 2, \ldots, d\}$, say, uniformly at random, and then choose the random vector θ so that it is always zero in positions outside of F; the same non-linearity (e.g., $x \mapsto \cos(\langle \theta, x \rangle + b)$) can be applied once the sparse random vector is chosen. Compared to the random Fourier feature approximation to the Laplacian kernel, the vectors θ chosen this way are truly sparse, which can make the random feature expansion more computationally efficient to apply.

Note that random Fourier features with such sparse sampling distributions in fact correspond to shift-invariant kernels that are rather different from the Laplacian kernel. For instance, if the non-zero entries of θ are chosen as i.i.d. $N(0, \sigma^{-2})$, then the corresponding shift-invariant kernel is

$$K(x, x') = \sum_{F \subseteq \{1,2,\ldots,d\} : |F| = k} \prod_{i \in F} \exp\left(-\frac{(x_i - x'_i)^2}{2\sigma^2} \right). \quad (1)$$

The kernel in Eq. (1) puts equal emphasis on all raw feature subsets F of size k. However, the feature selection may effectively bias the distribution of the feature subsets to concentrate on some small family $\mathcal{F}$ of raw feature subsets.

5. EXPERIMENTS

In this section, we describe an empirical study of our proposed random feature selection method in the setting of acoustic modeling. Our aim is to quantify the effect of random feature selection relative to the baseline use of random features without feature selection. We hypothesize that for a fixed number of random features, random feature selection will yield a boost in the predictive performance of the learned model compared to the baseline method.

We instantiate our method with three different shift-invariant kernels: the Gaussian kernel, the Laplacian kernel, and the "k-sparse Gaussian" kernel from Eq. (1) with $k = 5$. In each case, we use the random Fourier features method of [4] to generate the random features, and train a linear model on top of these random features using SGD with logistic loss, as described in Section 3.1. Our implementation of SGD uses a learning rate decay scheme as described in [20–22], and mini-batches of size 250 across all our experiments.

We experiment with two different annotated speech recognition data sets, the IARPA Babel Program Cantonese and Bengali limited language packs[4]. Each data set is comprised of about 20 hours of speech for training, and about another 20 hours for testing. The data sets are preprocessed using a standard pipeline[5]. Ultimately, this yields training and test sets of labeled examples (x, y), where $x \in \mathbb{R}^{360}$ and $y \in \{1, 2, \ldots, 1000\}$ is a label corresponding to a context-dependent phoneme state. We set aside some of the training examples as a hold-out set for parameter tuning. For Cantonese, we have $n_{\text{tr}} = 7{,}675{,}795$ (training), $n_{\text{ho}} = 1{,}047{,}593$ (hold-out), and $n_{\text{te}} = 7{,}052{,}334$ (test); and for Bengali, $n_{\text{tr}} = 7{,}483{,}896$, $n_{\text{ho}} = 882{,}588$, and $n_{\text{te}} = 7{,}197{,}328$.

[4]The specific versions of the Cantonese and Bengali data sets are, respectively, IARPA-babel101-v0.4c and IARPA-babel103b-v0.4b.

[5]Raw audio is aligned with phoneme state labels and 360 acoustic features are extracted for each frame [23], using a fixed frame shift of 10 ms and a frame length of 25 ms (overlapping frames). The features for each frame correspond to the concatenation of nine 40-dimensional feature vectors: one for the frame itself, and one for each of eight surrounding frames.

We vary the number of random features $D \in \{5{\times}10^3, 10^4, 2.5{\times}10^4, 5{\times}10^4, 10^5\}$ that are ultimately used in the final acoustic models. For each iteration of random feature selection, we draw a random subsample of the training data of size $n' = 10^6$ (except when $D = 10^5$, in which case we use $n' = 2{\times}10^6$, to ensure a safe n to D ratio), but ultimately we use all training examples once the random features are selected. Thus, each iteration of feature selection has equivalent computational cost to a n'/n_{tr} fraction of an SGD epoch (roughly $1/7$ or $2/7$ for $D < 10^5$ and $D = 10^5$ respectively, on these speech data sets). We use $T = 50$ iterations of feature selection, and in iteration t, we select $s_t = t \cdot (D/T) = 0.02Dt$ random features. Thus, the total computational cost we incur for feature selection is equivalent to approximately seven (or 14) epochs of training. Kernel bandwidths and the initial SGD learning rate are tuned on the hold-out set.

Due to space considerations, we will only include figures for the Cantonese dataset in this paper; qualitatively, the Cantonese and Bengali results are very similar.

Figure 1 plots the perplexity (i.e., exponential of average logistic loss), as well as the token error rate (TER), of each learned model, computed on the test set, as a function of the number of random features D.

Some observations regarding the perplexity results: First, as expected, there is a general trend of increasing performance with more random features. Second, in all cases, random feature selection delivers a substantial improvement in perplexity compared to the baseline. For instance, on the Cantonese data set with the Laplacian kernel approximation, the model with 10^4 random features obtained using random feature selection equals the performance of the baseline model with $5{\times}10^4$ random features. Third, in terms of perplexity, the k-sparse Gaussian (i.e., "Gaussian-k") and Laplacian kernel approximations outperform the Gaussian kernel approximations. The best results are delivered by applying random feature selection to the Laplacian kernel approximation.

Now, some observations regarding the TER results: First, it is important to note that these results are "noisier" than the perplexity results, suggesting there is a non-trivial relationship between perplexity and TER. Second, performing feature selection typically helps, most notably for the Laplacian kernel, giving a full 1% improvement in many cases. However, the gains are much weaker for Gaussian and Gaussian-k kernels, And there is even a case where feature selection hurts TER performance.

To assess these results on an absolute scale, we trained acoustic models using deep neural networks (DNNs) on both the Cantonese and Bengali data sets, following the approach detailed in [22]. For Cantonese, the DNN with best hold-out perplexity has four hidden layers, with 2000 units per hidden layer; for Bengali, the best DNN has four hidden layers with 3000 units per hidden layer. We use the $\tanh$ activation function. We compare these to the best kernel model on the hold-out set, which uses the Laplacian kernel with random feature selection and $D = 100{,}000$. The test results are in table 1:

The kernel model has better perplexity than the best DNN on the Cantonese data set, but it is the reverse on Bengali. It turns out many of the errors of the kernel model are due to confusions between multiple "silence" states. We therefore also compute a "collapsed perplexity" score, where all "silence" states are treated as the same state. We find that on this measure, the kernel model outperforms the best DNN on both data sets. Interestingly, even though the DNN and kernel models have very similar test perplexity, the DNN outperforms the kernel model in Token Error Rate (TER). An important area of future work is better understanding why this occurs. Note that this result once again highlights the complicated relation-

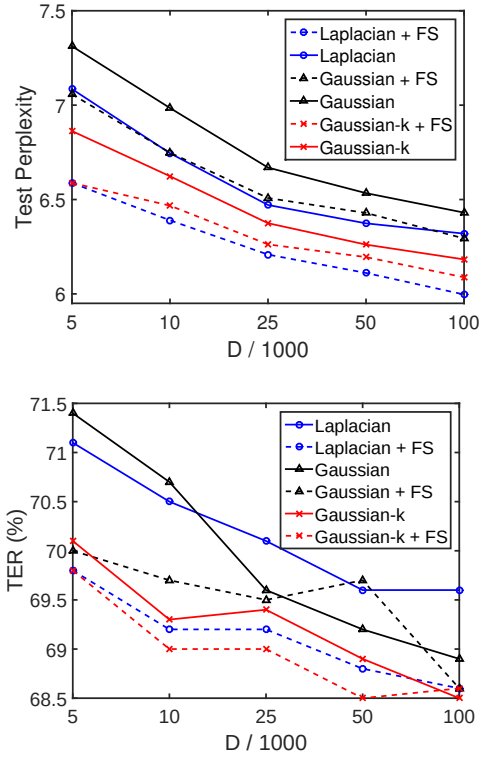

Fig. 1. Test performance for acoustic modeling experiments on Cantonese dataset, in terms of perplexity as well as TER. The appended "+FS" indicates use of random feature selection.

Dataset	Method	Perplexity	Collapsed	TER
Cant.	DNN	6.127	4.316	**67.3%**
	Lap+FS	**5.997**	**4.176**	68.6%
Beng.	DNN	**3.616**	3.256	**71.3%**
	Lap+FS	3.678	**3.233**	72.7%

Table 1. Test results: best kernel model vs. best DNN model.

ship between perplexity and TER, which we observed in the analysis of figure 1: One important caveat here is that our best kernel model ($D = 100{,}000$) has 100 million trainable parameters, which is more than these competing DNNs: ≈ 14 million and ≈ 31 million parameters, for the 2000-wide and 3000-wide networks, respectively.

5.1. Effects of random feature selection

We now explore whether the proposed feature selection criteria is robust; in our method, there is no guarantee that a feature selected in one iteration will be selected in the next. In Figure 2, we plot the fraction of the s_t features selected in iteration t that actually remain in the model after all T iterations. We only show the results for Cantonese for $D = 50{,}000$, as the plots of other values of D are qualitatively similar. In nearly all iterations and for all kernels, over half of the selected features survive to the final model. For instance, over 90% of the Laplacian kernel features selected at iteration 10 survive the remaining 40 rounds of selection. For comparison, we also plot the expected fraction of the s_t features selected in iteration

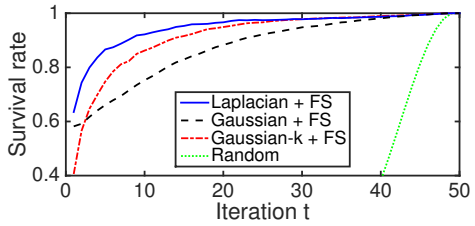

Fig. 2. Fraction of the s_t features selected in iteration t that are in the final model (survival rate) for Cantonese dataset.

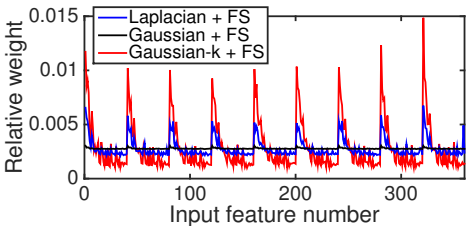

Fig. 3. The relative weight of each input feature in the random matrix Θ, for Cantonese dataset, $D = 50{,}000$.

t that would survive until the end if the selected features in each iteration were chosen uniformly at random from the pool. Since we use $s_t = Dt/T$, the expected fraction in iteration t is $T!/(t! \cdot T^{T-t})$, which is exponentially small in T until $t = \Omega(T/\log(T))$.

Finally, we consider how the random feature selection can be regarded as selecting non-linear combinations of input features. Consider the final matrix of random vectors $\Theta := [\theta^{(1)}|\theta^{(1)}|\cdots|\theta^{(D)}] \in \mathbb{R}^{d\times D}$ after random feature selection. A coarse measure of how much influence an input feature $i \in \{1, 2, \ldots, d\}$ has in the final feature map is the relative "weight" of the i-th row of Θ.

In Figure 3, we plot $\sum_{j=1}^{D} |\Theta_{i,j}|/Z$ for each input feature $i \in \{1, 2, \ldots, d\}$. Here, $Z = \sum_{i,j} |\Theta_{i,j}|$ is a normalization term. There is a strong periodic effect as a function of the input feature number. The reason for this stems from the way the acoustic features are generated. Recall that the features are the concatenation of nine 40-dimensional acoustic feature vectors for nine audio frames. An examination of the feature pipeline from [23] reveals that these 40 features are ordered by a measure of discriminative quality (via linear discriminant analysis). Thus, it is expected that the features with low $(i-1) \bmod 40$ value may be more useful than the others; indeed, this is evident in the plot. Note that this effect exists, but is extremely weak, with the Gaussian kernel. We believe this is because Gaussian random vectors in $\mathbb{R}^d$ are likely to have *all* their entries be bounded in magnitude by $O(\sqrt{\log(d)})$.

6. CONCLUSION

The random feature selection method proposed in this work delivers consistent improvements in perplexity over the baseline use of random features for kernel approximation, matching or beating DNN performance on this metric. Nonetheless, DNNs outperform the kernel models we have trained in terms of TER, suggesting that the strong performance of DNNs cannot be entirely explained by their ability to minimize test perplexity effectively. Further analysis of this phenomenon will be the topic of future work.

References

[1] B. Schölkopf and A.J. Smola, *Learning with Kernels*, MIT Press, Cambridge, MA, 2002.

[2] I. Steinwart, "Sparseness of support vector machines—some asymptotically sharp bounds," in *Advances in Neural Information Processing Systems 16*, 2004.

[3] C.K.I. Williams and M. Seeger, "Using the Nyström method to speed up kernel machines," in *Advances in Neural Information Processing Systems 13*, T.K. Leen, T.G. Dietterich, and V. Tresp, Eds., pp. 682–688. MIT Press, 2001.

[4] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Proceedings of NIPS*, 2007.

[5] R. Hamid, A. Gittens, Y. Xiao, and D. Decoste, "Compact random feature maps," in *ICML*, 2014.

[6] Q. V. Le, T. Sarlós, and A. J. Smola, "Fastfood: Approximate kernel expansions in loglinear time," *CoRR*, vol. abs/1408.3060, 2014.

[7] F. X. Yu, S. Kumar, H. Rowley, and S.-F. Chang, "Compact Nonlinear Maps and Circulant Extensions," *ArXiv e-prints*, Mar. 2015.

[8] B. Dai, B. Xie, N. He, Y. Liang, A. Raj, M.-F. Balcan, and L. Song, "Scalable kernel methods via doubly stochastic gradients," in *Advances in Neural Information Processing Systems 27*. 2014.

[9] E.-H. Yen, T.-W. Lin, S.-D. Lin, P.K. Ravikumar, and I.S. Dhillon, "Sparse random feature algorithm as coordinate descent in hilbert space," in *Advances in Neural Information Processing Systems 27*, 2014.

[10] John C. Duchi and Yoram Singer, "Efficient online and batch learning using forward backward splitting," *Journal of Machine Learning Research*, vol. 10, pp. 2899–2934, 2009.

[11] R. Tibshirani, "Regression shrinkage and selection via the Lasso," *Journal of the Royal Statistical Society, Series B*, vol. 58, pp. 267–288, 1994.

[12] A. Y. Ng, "Feature selection, l1 vs. l2 regularization, and rotational invariance," in *Proceedings of the Twenty-first International Conference on Machine Learning*, 2004.

[13] Yann Le Cun, John S. Denker, and Sara A. Solla, "Optimal brain damage," in *Advances in Neural Information Processing Systems*. 1990, pp. 598–605, Morgan Kaufmann.

[14] Song Han, Jeff Pool, John Tran, and William Dally, "Learning both weights and connections for efficient neural network," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N.D. Lawrence, D.D. Lee, M. Sugiyama, and R. Garnett, Eds., pp. 1135–1143. Curran Associates, Inc., 2015.

[15] P. Huang, H. Avron, T.N. Sainath, V. Sindhwani, and B. Ramabhadran, "Kernel methods match deep neural networks on timit," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2014, vol. 1, p. 6.

[16] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[17] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 1, pp. 30–42, 2012.

[18] T. Zhang, "Solving large scale linear prediction problems using stochastic gradient descent algorithms," in *ICML*, 2004.

[19] L. Bottou and O. Bousquet, "The tradeoffs of large scale learning," in *Advances in Neural Information Processing Systems 20*. 2008.

[20] N. Morgan and H. Bourlard, "Generalization and parameter estimation in feedforward nets: Some experiments," in *Advances in Neural Information Processing Systems 2*, 1990.

[21] T.N. Sainath, B. Kingsbury, V. Sindhwani, E. Arisoy, and B. Ramabhadran, "Low-rank matrix factorization for deep neural network training with high-dimensional output targets," in *ICASSP*, 2013.

[22] T.N. Sainath, B. Kingsbury, H. Soltau, and B. Ramabhadran, "Optimization techniques to improve training speed of deep neural networks for large speech tasks," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 21, no. 11, pp. 2267–2276, Nov 2013.

[23] B. Kingsbury, J. Cui, X. Cui, M. JF Gales, K. Knill, J. Mamou, L. Mangu, D. Nolden, M. Picheny, B. Ramabhadran, R. Schlüter, A. Sethy, and P. C. Woodland, "A high-performance cantonese keyword search system," in *ICASSP*, 2013.