RAPID ADAPTATION FOR MOBILE SPEECH APPLICATIONS

Michiel Bacchiani

Google Inc.,
New York, NY 10011
michiel@google.com

ABSTRACT

We investigate the use of iVector-based rapid adaptation for recognition in mobile speech applications. We show that on this task, the proposed approach has two merits over a linear-transform based approach. First it provides larger error reductions (11% vs. 6%) as it is better suited for the short utterances and varied recording conditions. Second it omits the need for speaker data pooling and/or clustering and the very large infrastructure complexity that accompanies that. Empirical results show that although the proposed utterance-based training algorithm leads to large data fragmentation, the resulting model re-estimation performs well. Our implementation within the MapReduce framework allows processing of the large statistics that this approach gives rise to when applied on a database of thousands of hours.

Index Terms— iVectors, rapid adaptation, Eigenvoices, GMM

1. INTRODUCTION

Acoustic model adaptation has been shown to give significant error rate reductions of automatic speech recognition for numerous large vocabulary transcription applications. A large number of the algorithms still commonly used today were developed in light of the DARPA Wall Street Journal read speech task [1, 2, 3, 4] but also showed effective when focus shifted toward spontaneous speech like the Broadcast News [5] and Switchboard [6] tasks.

The most widely used algorithms for adapting Hidden Markov Models (HMMs) that use Gaussian Mixture Models (GMMs) are Maximum A Posteriori (MAP) adaptation [7], Maximum Likelihood Linear Regression (MLLR) adaptation [8, 9, 10] and Constrained Maximum Likelihood Linear Regression (CMLLR) adaptation [11, 12]. For all these techniques, a Speaker Independent (SI) system is first used to get an initial transcript of the test data and an association of the acoustic observations with the model parameters. Using that association, statistics can be gathered to compute a Speaker Adapted (SA) setup. The various adaptation techniques need to balance the amount of data available for adaptation parameter estimation with the adaptation model parameter set size. The number of parameters in MAP adaptation can be as large as the model itself whereas in MLLR/CMLLR the adaptation parameter space is limited to one or more (possibly structured) linear transforms [8, 13, 14]. This allows tailoring of the adaptation model to the size of the adaptation data. However, when the adaptation sample becomes very small, the transform needs to be very constrained and as a result the effectiveness of adaptation deteriorates.

In some applications, data pooling can be employed to gather adaptation data for a speaker from multiple utterances. This is for example common in Broadcast News transcription where multiple speakers take turns within one recording, each turn frequently only a very short utterance. Supervised speaker turns or unsupervised clustering of utterances to find pooled speaker data for linear transform-based adaptation was shown to be an effective adaptation approach in that case [5].

Adaptation on very small adaptation samples directly is referred to as rapid adaptation which received a large amount of attention under the names Eigenvoices [15, 16, 17] and Cluster Adaptive Training (CAT) [18]. Common among these approaches is to form a subspace basis for the acoustic model at training time, requiring only the computation of a position in that subspace at test time. Since the subspace dimension is small, the amount of data required to estimate the subspace position is small. Given that the different subspace bases can represent varied subsets of the training data, they can differ significantly. Hence, even though the adaptation parameters are parsimonious, it can have a large impact on the characteristics of the model and hence provide large adaptation effectiveness.

More recently, the eigenvoice model has become popular in speaker identification as well. There, the subspace coordinates themselves are the focus as they have shown to be effective in characterizing speaker identity [19]. In this field, the the framework is known as iVector modeling.

In Eigenvoice modeling, training data pools for various training speakers (and/or recording conditions) are formed to estimate the model subspace bases. Here the model bases are represented by recognition GMMs. In contrast, in the iVector training, the model bases are for a text-independent GMM which is generally significantly smaller than a recognition GMM system. However, the iVector model interpolation parameters (the iVectors) are computed for each utterance rather than per speaker (allowing within speaker normalization using the resulting iVectors). Since the text independent GMM is smaller, the iVector approach generally uses a larger subspace than Eigenvoice modeling.

In this work, the focus is applying adaptation in mobile speech applications like our VoiceSearch application [20]. Applying adaptation in this domain is challenging not only because utterances are very short but they are from a very large speaker population. Even among the utterances from a given device, it is common to see a wide variety of recording conditions as speech is input while on the go. As such, it is hard to define data pools that are consistent in terms of a speaker (and recording condition).

Experiments with linear transform-based adaptation on data pooled per device (for a population of users that opted into allowing us to use their data for adaptation experiments) showed a 6% relative error rate reduction, much smaller then commonly observed in transcription [10]. Conjecture is that this poor performance is due in part to inhomogeneity in the adaptation data and in part due to a large variance in speaker data pool size (many speakers have only very little data, some have a lot). Note that besides poor performance, the approach of retaining aggregate transform statistics per

device only for those users that opted into using adaptation requires a large complex infrastructure of storing/updating/retrieving those statistics.

In this work, we focus on using rapid adaptation. Such an approach addresses the inhomogeneity issues we observed in the device specific data and omits the infrastructure complexity related to that approach. Given that even in training we cannot form consistent speaker data pools like in the Eigenvoice approach, we will treat each utterance as a different speaker/condition in both training and test, ie. we use the iVector paradigm in training and test. Note that this leads to a very large fragmentation (since we will have a very large "number of speakers") in the training phase potentially making the estimation unstable and having practical ramifications in the sense that the statistics required become large. The fact that the recognition GMM used in this model is much larger than the text independent GMM used in speaker identification further exacerbates this.

The rest of this paper is organized as follows. In section 2 we briefly review the iVector-based model. In section 3 the implementation of the algorithm using the MapReduce framework [21] is described. Section 4 described the experimental results obtained with the proposed adaptation model. Finally section 5 summarizes the results.

2. IVECTOR MODEL

Here we briefly summarize the Eigenvoice algorithm used in this work. We closely follow the very concise write up in [17], and we only summarize the results. Let $M(i)$ denote the Nd dimensional supervector for the i-th utterance obtained by stacking the d dimensional mean vectors for all N components of the GMM. The adapted mean relates to the SI GMM mean supervector M_0 as

$$M(i) = M_0 + Vy(i), \tag{1}$$

where V is a matrix encoding the bases of an R dimensional subspace as the columns of this $Nd \times R$ matrix. The R dimensional $y(i)$ vector is referred to as the iVector for the i-th utterance, signifying a location in the subspace for the i-th utterance.

Let the acoustic observations for the i-th utterance be denoted as $\mathcal{X}^i = x_1, x_2, \ldots, x_{L^i}$ with L^i the number of observations in the i-th utterance. Furthermore, lets define counts for mixture component c

$$N_c^i = \sum_{x \in \mathcal{X}^i} \mathrm{P}(c \mid \mu_c, \Sigma_c, x) \tag{2}$$

and

$$S_c^i = \sum_{x \in \mathcal{X}^i} \mathrm{P}(c \mid \mu_c, \Sigma_c, x)(x - \mu_c) \tag{3}$$

with μ_c and Σ_c the mean and covariance of the c-th GMM component. Let N^i denote the $Nd \times Nd$ block diagonal matrix with the c-th block as $N_c^i I$ where I denotes a $d \times d$ identity matrix. Let Σ denote the model supercovariance matrix structured likewise from the component covariances. Let S^i denotes the supervector obtained by stacking the S_c^i counts for all GMM components c.

The model assumes that the iVectors themselves are random variables with a prior distribution assumed to be a zero-mean, unit variance normal. This gives rise to a Bayesian model where the posterior distribution of the iVectors given the SI GMM and acoustic observations is itself distributed Gaussian with precision

$$l(i) = I + V^T \Sigma^{-1} N^i V, \tag{4}$$

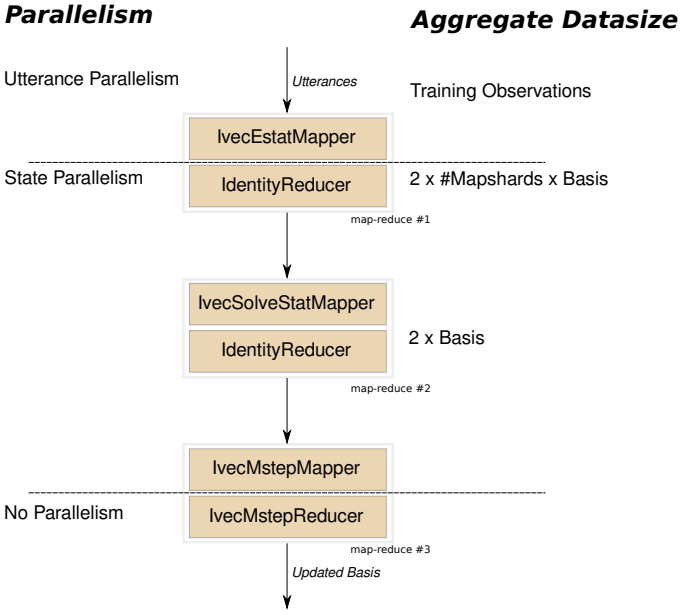

Fig. 1. Outline of the parallelism and data sizes used in the three map-reductions that implement basis training.

(where the T superscript denotes transposition) and mean

$$a(i) = l(i)^{-1} V^T \Sigma^{-1} S^i. \tag{5}$$

This readily provides the MAP iVector estimate for the i-th utterance as $a(i)$ and gives expectations

$$\mathrm{E}\left[y(i)\right] = l(i)^{-1} V^T \Sigma^{-1} S^i \tag{6}$$

$$\mathrm{E}\left[y(i) y(i)^T\right] = \mathrm{E}\left[y(i)\right] \mathrm{E}\left[y(i)^T\right] + l(i)^{-1}. \tag{7}$$

Given the posterior iVector distribution, an EM optimization can be formulated to update the bases V so as to maximize data likelihood by solving in the maximization step the system of linear equations

$$\sum_{i \in \mathcal{O}} N^i V \mathrm{E}\left[y(i) y(i)^T\right] = \sum_{i \in \mathcal{O}} S^i \mathrm{E}\left[y(i)\right], \tag{8}$$

where $\mathcal{O}$ denotes the set of training utterances.

At test time, a SI system is used to get a first transcript for the utterance which allows the computation of the statistics in Equation (2) and (3) for the test data. These statistics allow the computation of the MAP iVector estimate in Equation (5) which provides the SA model.

3. IVECTOR TRAINING IMPLEMENTATION

We implemented the iVector basis training using our MapReduce [21] framework for parallelization. The framework allows an input data set to be partitioned (*sharded*) across a number of mappers. The user implements the map process what will process the input shards in parallel. Each mapper can output records under a key. Once mapping completes, the framework sorts the map outputs (*shuffling*) and provides the outputs with matching keys to a user implementation of a reduce process. If data was shuffeled for multiple keys, the framework allows parallelism in the reduce phase as well.

One iteration of the iVector basis optimization was implemented as a series of three map-reductions as depicted in Figure 1. The picture outlines the parallelism and the total data size that is processed by the series of map-reductions at the various stages. This process might appear overly complex at first glance but it is important to realize that with a large data set (in our experiments thousands of hours of data) fragmented into a large number of utterances (millions in our experiments), the set of utterance statistics is large.

In the first map-reduction, the mapper computes for each input utterance the occupancies of Equations (2) and (3), then computes the iVector expectations of Equations (6) and (7) and finally updates accumulators for the left and righthand sides of Equation (8). This is parallelized across map shards of utterances. Once a mapper finishes a map shard, it outputs the accumulators it collected. As can be seen from Equation (8) the left and righthand sides are V-sized matrices, ie. the GMM means times the subspace dimension. For the model used in the experiments (see section 4) and a subspace dimension of 32, this amounts to about 380MB. In addition, to handle a large training set (in our experiments we used about 2000 hours), one would need utterance parallelism to process the data in a reasonable amount of time. In our experiments, we used about 2000 map shards and used 2000 machines for the first map-reduction leading to an aggregate output size of about 1TB. In order to facilitate counts of this size, we output the accumulators in fragments, outputting the rows of the accumulators for a particular GMM state under the state name key. The reducer of the first map reduction uses an IdentityReducer writing the keyed gathered accumulator counts to disk unaltered.

In the second map reduction the accumulator counts of the first are read in the map processes. These processes see numerous accumulators for a given state name key that were computed in the various mappers of the first map-reduction. The map processes of this second map reduction sum the accumulators of a particular state name key and output the sum again under the state name key. After this summation, the aggregate data size is reduced to the size of the left and righthand sides of Equation (8), ie. two times the size of V or, for the example given, about 380MB. This second map-reduction also uses an IdentityReducer to output the summed accumulators to disk.

Finally, in the third map-reduction, the map processes read in the summed accumulators of the second map-reduction (now one per GMM state key) and output these accumulators under a single key to bring all the summed counts together into a single reduce process. This reduce process takes the summed accumulators from all states, assembles the total accumulator sums from Equation (8) and solves for the updated bases V.

To allow evaluation of various subspace sizes and to aid in initialization of the basis vectors, we implemented a stage-wise subspace dimension increase. Once we obtain a k-dimensional basis from EM training, we obtain added subspace dimensions by taking a random direction and projecting that direction on an orthogonal basis direction by use of QR factorization of the newly formed basis. We repeat this process for the bases of each GMM state in V for bases up to the vector dimensionality. If a larger subspace dimension is desired, initialization is obtained by taking permutations of the state specific basis vectors obtained so far.

4. EXPERIMENTAL RESULTS

Experiments were conducted on a database of mobile speech recordings originating from a number of mobile speech applications: voice search, translation and the voice-based input method used on Android phones. These recordings are anonymized; we only retain the speech recording but remove all infromation indicating which device recorded the data. The training set consists of a sampling of those recordings and consists of 2421558 utterances containing about 1967 hours of speech. A single model is used in evaluations but we use four test sets, one for each application and one (`Unified_1A`) that contains data sampled uniformly from the join of all applications (emphasizing the use frequency of each application). Each test set contains about 20000 utterances or about 20 hours of speech.

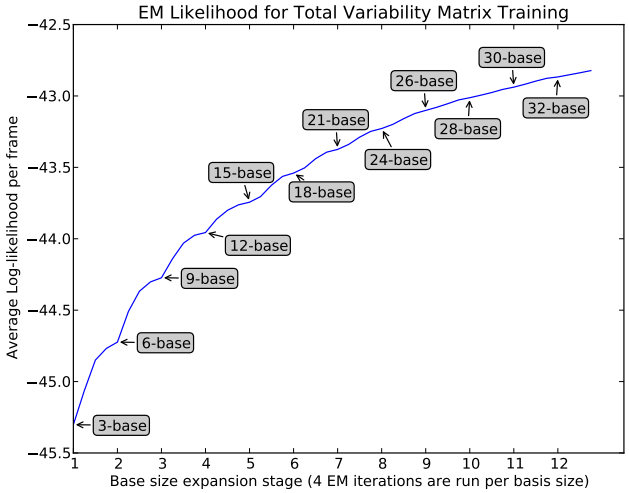

Fig. 2. Average log-likelihood per frame for the iVector basis training.

For the GMM system used in these experiments, we trained a decision tree-based tied-state triphone model with 2284 state distributions. For each state, emissions were modeled by GMM distributions where the number of components were chosen commensurate to the amount of data that was available for the state resulting in a model with 38121 mixture components. This is significantly larger than the 1024 component text independent Gaussian mixture model commonly used in iVector modeling. We used a 39 dimensional LDA+STC feature space computed from 9 consecutive 13 dimensional PLP cepstral vectors. Note that for the subspace models, each added subspace dimension adds a set of 38121 mean vectors. Hence, for the 32-component model used in some of these experiments, the final model size is about 290MB.

Figure 2 shows the average log-likelihood per frame throughout the basis training of the iVector model. The subspace dimension of the model was trained in stages initializing a larger model using the algorithm detailed in section 3. For each size, four EM iterations were run to estimate the bases. The E-step ivector computation and statistics summation were run on 2000 machines for the larger complexity model. For the 32-dimension model, this leads to about 1TB of data resulting from the first E-step map-reduction. For that model complexity, the training iterations took about an hour each.

Figure 3 shows the error rate of the iVector adapted model as function of the subspace dimension on the four test sets. Adaptation shows relative error rate reductions ranging from about 8% on the search task to about 11% on the sampled `Unified_1A` test set. The results show that performance gains peak for all tests when using a subspace dimension of about 28. Note that in contrast to linear transform adaptation, the proposed iVector adaptation uses no data beside the short utterances (about 4 seconds on average) and requires

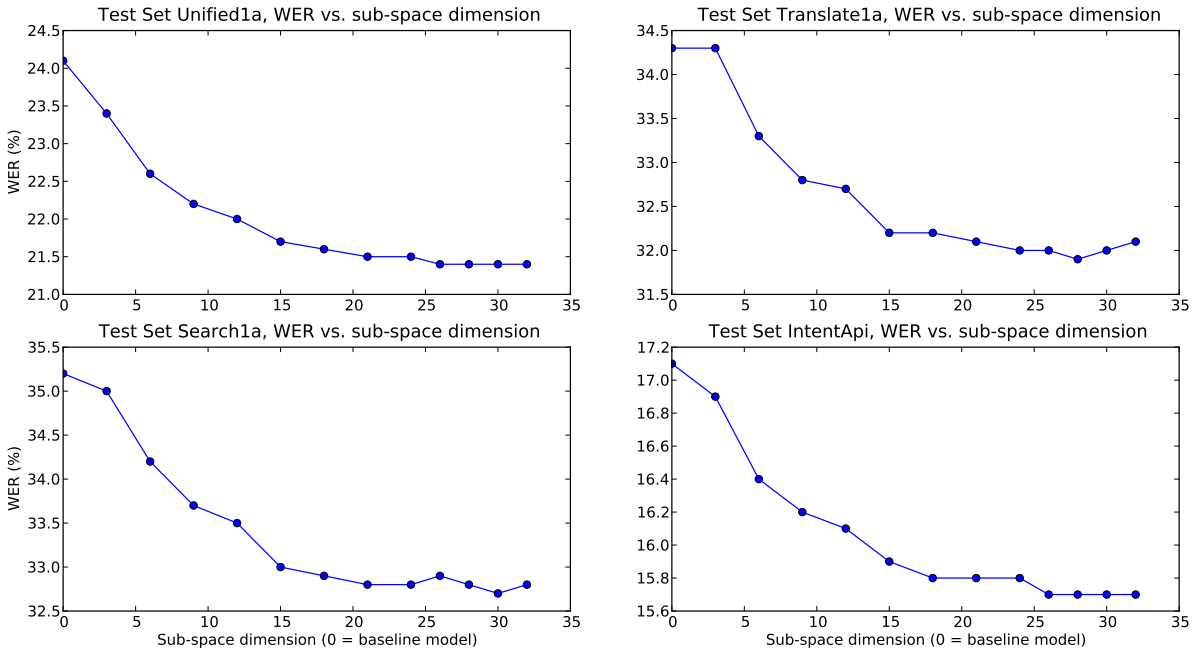

Fig. 3. Word error rates on various test sets as a function of the subspace dimension of the iVector model.

no data pooling or other metadata.

5. CONCLUSIONS

The proposed iVector-based rapid adaptation algorithm shows merit for use in mobile speech applications. Not only does it show better adaptation performance than the linear transform-based approach, it also alleviates the need for device-based statistics aggregation and/or clustering.

The empirical evidence shows that the algorithm appears to scale to use of a large speech recognition GMM (relative to the text independent GMM generally used iVector modeling). Although some attention needs to be paid to making the sizable statistic summation practical, our use of the MapReduce framework seems to successfully address this as it allowed us to train a 32-dimensional subspace model on the millions of utterances in the 2000 hour training set.

Ackowledgements

The author would like to thank Olivier Siohan for his care in reviewing and discussing the software and manuscript for this paper and Hank Liao for sharing the linear transform-based results.

6. REFERENCES

[1] P. Woodland and C. Leggetter and J. Odell and V. Valtchev and S. Young, "The 1994 HTK Large Vocabulary Speech Recognition System," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, 1995, vol. 1, pp. 73–76.

[2] G. Zavaliagkos and R. Schwartz and J. Makhoul, "Batch, Incremental and Instantaneous Adaptation Techniques for Speech Recognition," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, 1995, vol. 1, pp. 676–679.

[3] M. Riley and A. Ljolje and D. Hindle and F. Pereira, "The AT&T 60,000 Word Speech-to-Text System," in *Proc. European Conf. on Speech Communication and Technology*, 1995, pp. 207–210.

[4] L. Bahl, S. Balakrishnan-Aiyer and J. Bellgarda and M. Franz and P. Gopalakrishnan and D. Nahamoo and M. Novak and M. Padmanabhan and M. Picheny and S. Roukos, "Performance of the IBM Large Vocabulary Continuous Speech Recognition System on the ARPA Wall Street Journal task," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, 1995, vol. 1, pp. 41–44.

[5] D. Pallet, "Overview of the 1997 DARPA Speech Recognition Workshop," in *Proceedings of the DARPA Speech Recognition Workshop*, 1997.

[6] J. Godfrey and E. Holliman and J. McDaniel, "SWITCHBOARD: Telephone Speech Corpus for Research and Development ," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, 1992, vol. 1, pp. 517–520.

[7] J. Gauvain and C. Lee, "Maximum a-posteriori estimation for multivariate Gaussian mixture observations of Markov chains," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 2, pp. 291–298, 1994.

[8] C. Leggetter and P. Woodland, "Maximum Likelihood Linear Regression for Speaker Adaptation of Continuous Density Hidden Markov Models," *Computer Speech and Language*, vol. 9, pp. 171–185, 1995.

[9] M. Padmanabhan and L. Bahl and D. Nahamoo and M. Picheny, "Speaker clustering and transformation for speaker adaptation in large-vocabulary speech recognition systems," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, 1996, vol. 2, pp. 701–704.

[10] T. Anastasakos and J. McDonough and R. Schwartz and J. Makhoul, "A Compact Model for Speaker-Adaptive Training," in *Proc. Int. Conf. on Spoken Language Processing*, 1996, vol. 2, pp. 1137–1140.

[11] V. Digalakis, D. Rtischev, L. Neumeyer, "Speaker Adaptation Using Constrained Estimation of Gaussian Mixtures," *IEEE Transactions on Speech and Audio Processing*, vol. 3, no. 5, pp. 357–366, 1995.

[12] M. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer Speech and Language*, vol. 12, no. 2, 1998.

[13] E. Bocchieri,V. Digalakis, A. Corduneanu, C. Boulis, "Correlation modeling of MLLR transform biases for rapid HMM adaptation to new speakers," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, 1999, vol. 2, pp. 773–776.

[14] R. Haeb-Umbach, "Automatic Generation of Phonetic Regression Class Trees for MLLR Adaptation," *IEEE Transactions on Speech and Audio Processing*, vol. 9, no. 3, pp. 299–302, 2001.

[15] P. Nguyen and C. Wellekens and J.-C. Junqua, "Maximum Likelihood Eigenspace and MLLR for Speech Recognition in Noisy Environments," in *Proc. European Conf. on Speech Communication and Technology*, 1999.

[16] R. Kuhn and J.-C. Junqua and P. Nguyen and N. Niedzielski, "Rapid Speaker Adaptation in Eigenvoice Space," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 6, pp. 695–707, 2000.

[17] P. Kenny and G. Boulianne and P. Dumouchel, "Eigenvoice Modeling With Sparse Training Data," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 3, pp. 345–354, 2005.

[18] M. Gales, "Cluster Adaptive Training of Hidden Markov Models," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 4, pp. 417–428, 2000.

[19] N. Dehak and P. Kenny and R. Dehak and P. Dumouchel and P. Ouellet, "Front-End Factor Analysis for Speaker Verification," *IEEE Transactions on Audio Speech and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[20] J. Schalkwyk and D. Beeferman and F. Beaufays and B. Byrne and C. Chelba and M. Cohen and M. Kamvar and B. Strope, "Google Search by Voice: A Case Study," in *Visions of Speech: Exploring New Voice Apps in Mobile Environments, Call Centers and Clinics*, A. Neustein, Ed. Springer, 2010.

[21] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," in *OSDI'04: Sixth Symposium on Operating System Design and Implementation*, San Francisco, CA, 2004.