ADAPTATION OF CONTEXT-DEPENDENT DEEP NEURAL NETWORKS FOR AUTOMATIC SPEECH RECOGNITION

Kaisheng Yao[1], Dong Yu[2], Frank Seide[3], Hang Su[3,4], Li Deng[2], and Yifan Gong[1]

[1]Online Service Division, Microsoft Corporation, Redmond, 98052, WA, USA
[2]Microsoft Research, Redmond, 98052, WA, USA
[3]Microsoft Research Asia, Beijing, China
[4]Tsinghua University, Beijing, China

ABSTRACT

In this paper, we evaluate the effectiveness of adaptation methods for context-dependent deep-neural-network hidden Markov models (CD-DNN-HMMs) for automatic speech recognition. We investigate the affine transformation and several of its variants for adapting the top hidden layer. We compare the affine transformations against direct adaptation of the softmax layer weights. The feature-space discriminative linear regression (fDLR) method with the affine transformations on the input layer is also evaluated. On a large vocabulary speech recognition task, a stochastic gradient ascent implementation of the fDLR and the top hidden layer adaptation is shown to reduce word error rates (WERs) by 17% and 14%, respectively, compared to the baseline DNN performances. With a batch update implementation, the softmax layer adaptation technique reduces WERs by 10%. We observe that using bias shift performs as well as doing scaling plus bias shift.

Index Terms— Context-Dependent Deep-Neural-Networks, HMM, speech recognition, speaker adaptation

1. INTRODUCTION

The context-dependent deep-neural-network hidden Markov model (CD-DNN-HMM) is a recent acoustic modeling technique that significantly outperforms the earlier state-of-the-art Gaussian-mixture-model based HMMs (GMM-HMMs) on several automatic speech recognition (ASR) tasks [1, 2]. CD-DNN-HMMs differ from the earlier artificial neural network (ANN)/HMM hybrids in that they have typically three or more hidden layers (hence deep) instead of one and that the output layer is much larger. Specifically, CD-DNN-HMMs model the tied *context-dependent* states (a.k.a. senones) directly.

Even though ASR systems are trained on large amounts of data, mismatches between the training and testing conditions are still unavoidable due to the speaker and environment differences. Much like other systems built upon statistical techniques, CD-DNN-HMM based ASR systems may fail to produce the same level of performance when tested under mismatched conditions.

A substantial amount of work has been conducted to improve the performance of the GMM-HMM based ASR systems under the mismatched conditions. A comprehensive review of the existing adaptation techniques designed for the GMM-HMMs can be found in [3]. The key idea of these techniques is to adapt either the model or the feature so that the mismatch between the training and testing conditions can be reduced. In the literature, several methods have also been proposed to compensate for the mismatches in the ANN/HMM hybrid systems [4, 5, 6]. Here we briefly review these techniques that motivated this work.

The first technique applies affine transformations to the inputs and outputs of a neural network [4]. The linear input network (LIN) [4, 7] augments a speaker-independent neural network (SINN) with a linear input layer. The augmented layer has the same dimension as the original neural network input layer. Similarly, the linear output network (LON) augments the SINN with a linear layer at the output, right before the softmax functions are employed. It was shown, however, in [4] that LON performs worse than the baseline SINN. Transformations on the weights of both input and hidden layers of an ANN are investigated in [5]. As the second technique developed for ANN, the linear hidden network (LHN) in [5] applies a linear transformation to the activations of the internal hidden layers. Similar to the LIN and LON, the weights of the LHN are estimated with the standard back-propagation algorithm [8] keeping frozen the weights of the original network. Better results are reported in [5] with LHN than with LIN. The third technique [6] changes the shape of the activation function to better fit the speaker-specific features.

Effort has also been made to reduce the mismatch in the context of DNN. For example, [9] investigates several widely used feature-space transformations, such as the vocal tract length normalization (VTLN) [10] and feature space maximum likelihood linear regression (fMLLR) [11], that are originally developed for GMM-HMMs. The proposed feature-space discriminative linear regression or fDLR is similar to LIN [4, 7, 12] with two differences. First, it is applied to DNN instead of shallow networks. Second, the weights of the augmented linear layer in fDLR is shared by all the frames in the input window. The fDLR [9] was observed to perform similarly for DNNs as fMLLR for GMM-HMMs.

This paper further investigates adaptation techniques on the top hidden layer of CD-DNN-HMMs. The motivation is based on a view of the DNN as a two-step process: nonlinear feature extraction at lower layers followed by a log-linear classification layer at the top. We report in this paper two variants of affine transformation in the top hidden layer adaptation and report their results. In addition, we investigate an adaptation method that directly changes bias parameters of the top-level log-linear model. We also report results that combine the fDLR and the top layer adaptation.

2. A BRIEF REVIEW OF CD-DNN

A DNN [1, 2, 13] consists of an input layer, an output layer, and many hidden layers. Denote the input vector at time t at layer ℓ as $\mathbf{v}^\ell(t)$, the weight matrix as $\mathbf{W}^\ell$ and bias vector as $\mathbf{a}^\ell$. The DNN computes the posterior probabilities of conditionally independent

hidden binary units $\mathbf{h}^\ell(t)$ given the input vector. In practice, the i-th element, $\mathbf{h}_i^\ell(t)$, is computed as follows

$$\mathbf{v}_i^{\ell+1}(t) = E^\ell_{\mathbf{h}|\mathbf{v}}\{\mathbf{h}_i^\ell(t)|\mathbf{v}^\ell(t)\} = \sigma(\mathbf{a}_i^\ell + \mathbf{W}_i^\ell \mathbf{v}^\ell(t)), \quad (1)$$

where $\sigma(x) = (1+\exp(-x))^{-1}$ is the sigmoid function. The above elements are the inputs to the next layer $\ell+1$. The input to the lowest layer $\ell = 0$ is the observation vector $\mathbf{o}(t)$.

For CD-DNN-HMM [1], the top layer L is a softmax function for each context-dependent phone state (a.k.a senone) id s; i.e.,

$$p_s^L(s|\mathbf{v}^L(t)) = \frac{\exp(\mathbf{a}_s^L + \mathbf{W}_s^L \mathbf{v}^L(t))}{\sum_j \exp(\mathbf{a}_j^L + \mathbf{W}_j^L \mathbf{v}^L(t))}. \quad (2)$$

The DNN may be initialized using "pre-training", which uses a contrast divergence algorithm on restricted Boltzmann machine RBM [14] for layers up to $L-1$. Alternatively, it may use random initialization. The top layer weights are usually randomly initialized. Once the initialization is done, a fine-tuning process using back-propagation algorithm updates all of the DNN parameters.

The DNN may be viewed from a discriminative feature learning perspective [9], in which the estimation of the posterior probability $p_s^L(s|\mathbf{o}(t))$ can be considered as a two step process. The first step transforms the observation vector $\mathbf{o}(t)$ into a feature vector $\mathbf{v}^L(t)$ through L layers of non-linear transformations, which may use the sigmoid function. In the second step, the posterior probability $p_s^L(s|\mathbf{o}(t))$ is estimated using the log-linear model with feature $\mathbf{v}^L(t)$ as in (2).

The conventional log-linear model for ASR [15] uses observation vector $\mathbf{o}(t)$ and its derivatives such as second-order differentials as the features. DNN, on the contrary, automatically learns the feature through many layers of non-linear transformations that maximize a discriminative objective function. In fact, our adaptation algorithms are motivated from this discriminative feature-learning view.

3. THE DNN ADAPTATION ALGORITHMS

We consider the outputs from the layer $L-1$ as the observation vectors to the top level log-linear model. For notational convenience, we denote the observation vector as $\mathbf{x}(t)$ as defined by

$$\mathbf{x}(t) = E^{L-1}_{\mathbf{h}|\mathbf{v}}\{\mathbf{h}^{L-1}(t)|\mathbf{v}^{L-1}(t)\} \quad (3)$$

with each element of $\mathbf{x}_i(t) = E^{L-1}_{\mathbf{h}|\mathbf{v}}\{\mathbf{h}_i^{L-1}(t)|\mathbf{v}^{L-1}(t)\}$.

We use hard alignment of senones and observations; i.e., at each time t, there is only one senone associated with the observation $\mathbf{x}(t)$. The objective function is frame-wise maximum mutual information (MMI) over the T samples $\mathbf{X} = \{\mathbf{x}(t)\}$ with ground-truth senone labels $s(t)$; i.e.,

$$D(\mathbf{X}) = \sum_t \log p(s(t)|\mathbf{x}(t)). \quad (4)$$

We introduce an affine transformation of $\mathbf{x}(t)$ as follows:

$$\hat{\mathbf{x}}(t) \quad = \quad \mathbf{M}\mathbf{x}(t) + \mathbf{c}, \quad (5)$$

where $\mathbf{M}$ denotes a rotation and scaling matrix. $\mathbf{c}$ is a bias shift vector. Notice that this affine transformation formula is commonly used as feature space maximum likelihood linear regression (fMLLR) [11] in GMM-HMM adaptation. There are two important differences to the normal fMLLR; First, $\mathbf{x}(t)$ has much larger dimension than the normal MFCCs. In our case, the dimension of $\mathbf{x}(t)$ is 2048, versus 36 of MFCCs features. Second, the transformation $\mathbf{M}$ and $\mathbf{c}$ are estimated discriminatively using the objective function in (6), whereas fMLLR uses maximum likelihood estimates.

3.1. Gradients and updating formulae for $\mathbf{M}$ and $\mathbf{c}$

The objective function is

$$D(\mathbf{X}, \mathbf{M}, \mathbf{c}) = \sum_t \log p(s(t)|\mathbf{x}(t), \mathbf{M}, \mathbf{c}) \quad (6)$$

For ease of reading, we derive the estimation of $\mathbf{c}$ in more detail. Estimations of $\mathbf{M}$ can be easily followed.

Expanding (6) with (2), we have the gradient with respect to $\mathbf{c}$ as

$$\begin{aligned}\frac{\partial D}{\partial \mathbf{c}} &= \sum_t \frac{\partial}{\partial \mathbf{c}}(\mathbf{a}^L_{s(t)} + \mathbf{W}^L_{s(t)}(\mathbf{M}\mathbf{x}(t)+\mathbf{c}) \\ &\quad - \log \sum_j \exp(\mathbf{a}_j^L + \mathbf{W}_j^L(\mathbf{M}\mathbf{x}(t)+\mathbf{c}))) \\ &= \sum_t {\mathbf{W}^L_{s(t)}}^T - \sum_j \frac{\exp(\mathbf{a}_j^L + \mathbf{W}_j^L(\mathbf{M}\mathbf{x}(t)+\mathbf{c})}{\sum_j \exp(\mathbf{a}_j^L + \mathbf{W}_j^L(\mathbf{M}\mathbf{x}(t)+\mathbf{c})} {\mathbf{W}_j^L}^T \\ &= \sum_t \left(\mathbf{W}^L_{s(t)} - \sum_j p(j|\mathbf{x}(t), \mathbf{M}, \mathbf{c})\mathbf{W}_j^L\right)^T, \end{aligned} \quad (7)$$

where superscript T denotes transpose.

The gradient with respect to $\mathbf{M}$ is derived similarly to above as

$$\frac{\partial D}{\partial \mathbf{M}} \quad = \quad \sum_t \left(\mathbf{W}^L_{s(t)} - \sum_j p(j|\mathbf{x}(t), \mathbf{M}, \mathbf{c})\mathbf{W}_j^L\right)^T \mathbf{x}(t)^T. \quad (8)$$

Because data for speaker adaptation may be limited, we usually only use the diagonal elements of $\mathbf{M}$. In this case, the formula above is the same except that only the diagonal elements are updated.

The transformation $\mathbf{M}$ and $\mathbf{c}$ are updated iteratively with the above gradients according to

$$\mathbf{M} \quad \leftarrow \quad \mathbf{M} + \epsilon_{\mathbf{M}} \frac{\partial D}{\partial \mathbf{M}} \quad (9)$$

$$\mathbf{c} \quad \leftarrow \quad \mathbf{c} + \epsilon_{\mathbf{c}} \frac{\partial D}{\partial \mathbf{c}}, \quad (10)$$

where $\epsilon_{\mathbf{M}}$ and $\epsilon_{\mathbf{c}}$ are the step size for updating $\mathbf{M}$ and $\mathbf{c}$, respectively. To decide these step sizes, we first compute the l^2 norm of the gradients, and then divide the gradient with the norm and multiply each element with a constant (e.g. 0.1).

For the example of $\mathbf{c}$, the step size for updating is

$$\epsilon_{\mathbf{c}} = \frac{\xi}{|\frac{\partial D}{\partial \mathbf{c}}|}, \quad (11)$$

where ξ is the small constant. In such a way, the change of $\mathbf{c}$ is always a fraction of the norm of $\frac{\partial D}{\partial \mathbf{c}}$.

3.2. Updating the top-level log-linear model parameters

Notice that by applying $\mathbf{M}$ and $\mathbf{c}$ we get

$$\begin{aligned}\mathbf{a}_j^L + \mathbf{W}_j^L(\mathbf{M}\mathbf{x}(t)+\mathbf{c}) &= \mathbf{a}_j^L + \hat{\mathbf{W}}_j^L\mathbf{x}(t) + \mathbf{W}_j^L\mathbf{c} \\ &= \hat{\mathbf{a}}_j^L + \hat{\mathbf{W}}_j^L\mathbf{x}(t), \end{aligned} \quad (12)$$

which corresponds to an indirect bias update $\hat{\mathbf{a}}_j^L = \mathbf{a}_j^L + \mathbf{W}_j^L\mathbf{c}$ and an indirect weight matrix update $\hat{\mathbf{W}}_j^L = \mathbf{W}_j^L\mathbf{M}$ to the top-level log-linear model parameters. This fact suggests that instead of transforming $\mathbf{x}(t)$, alternatively, we can estimate and update the top-level log-linear model parameters $\hat{\mathbf{a}}_j^L$ and $\hat{\mathbf{W}}_j^L$ directly as the mechanism for adaptation.

For the sake of speaker adaptation, the amount of adaptation data is often not sufficient to update the weight matrix $\mathbf{W}^L$. Hence, we only derive an update of $\mathbf{a}^L$, the bias vector in the log-linear model [1]. Without considering $\mathbf{M}$ and $\mathbf{c}$ and referring to (4) and (2), the gradient with respect to $\mathbf{a}_s^L$ is

$$\frac{\partial D}{\partial \mathbf{a}_s^L} = \sum_t \left(\delta(s(t)=s) - p(s|\mathbf{x}(t)\right). \quad (13)$$

Similar to (9) and (10), the bias $\mathbf{a}^L$ is updated as

$$\hat{\mathbf{a}}^L = \mathbf{a}^L + \epsilon_{\mathbf{a}^L}\frac{\partial D}{\partial \mathbf{a}^L}, \quad (14)$$

where the step size $\epsilon_{\mathbf{a}^L}$ is derived similarly to (11) as $\frac{\xi}{\|\frac{\partial D}{\partial \mathbf{a}^L}\|}$.

3.3. The feature-space discriminative linear regression

A transformation may be applied to the DNN layers other than the top one. In particular, [9] proposes the fDLR method that applies affine transformation on the input vectors. The affine transformation is estimated using back propagations (BP) [8]. This requires doing back-propagation from the top layer to the bottom layer. In terms of computational cost, it requires L-times of computations of adapting on the top layer.

In case of using a context window of several frames of observations, the fDLR usually estimates a block-diagonal transformation matrix, with each block corresponding to a frame of the observations. The fDLR averages the BP estimated transformations of these blocks to have a smoothed transformation for the context window of observations. We will report the results of the fDLR, together with adaption of the top layer in Sec.4.2.

3.4. Discussions

Notice that for GMM-HMM adaptation of large vocabulary speech recognition (LVSR), the feature dimension is usually much smaller than the number of GMM parameters. Therefore it is reasonable to do feature-space transformation in GMM-HMM, instead of changing model parameters. However, in adapting the top-level log-linear model of CD-DNN, the dimension of the last hidden layer outputs is either larger or comparable to the number of senones. We thus find it is practical to change DNN parameters, especially our change is only on the top layer.

We can either estimate the transformation $\mathbf{M}$ and $\mathbf{c}$ using a batch update or using a stochastic gradient update. In the batch update, the gradients are estimated and accumulated for each adaptation utterance. On the other hand, the stochastic gradient method randomly collects observation samples and forms mini-batches. The batch mode update usually requires loading the whole adaptation data into memory which may not be affordable under some conditions. In contrast, the stochastic gradient update approach demands smaller memory size. However, it is hard to do parallel updating on the stochastic gradient method.

[1] Alternatively, we may introduce regularization similarly as in [16].

Table 1. Word error rates by adaptation on the DNN using the stochastic gradient implementations

Methods	WER	WERR (in %)
GMM-HMM	43.6	
DNN	34.1	-
DNN + $\mathbf{c}$	29.4	13.9
DNN + fDLR	28.5	16.8
DNN + fDLR + $\mathbf{c}$	28.3	17.0

4. EXPERIMENTAL EVALUATION

4.1. Setups

We conduct automatic speech recognition (ASR) experiments on an internal Xbox voice search data set. The scenario supports distant talking voice search (of music catalog, games, movies, etc.) using a microphone array. The training set consists of 40 hours of the voice search data. The evaluation was conducted on data from 6 speakers. For each speaker approximately 200 utterances are used for supervised adaptation and approximately 200 utterances are used for testing. The total number of tested words is 5185. There is no overlap among training, adaptation and test sets. The features are 13-dimension Mel filter-bank cepstral coefficients (MFCC) with delta and delta-delta coefficients, further transformed from 52-dimension to 36-dimension by heterogeneous linear discriminate analysis (HLDA) [17]. Per-device cepstral mean subtraction is applied. The speaker independent (SI) acoustic model has 7k Gaussian components and 1509 senones trained with the standard maximum likelihood estimation (MLE) procedure. The trigram language model used in the system was trained on the transcriptions of all the data to ensure no out-of-vocabulary word exists in the language model. The baseline CD-DNN-HMMs system was trained on the same training data, excluding HLDA, as the GMM-HMMs system. The input has a context window size of 9, forming a vector of 468-dimension (52×9). On top of the input layer, there are three hidden layers with 2048-dimension each. The output layer has a dimension of 1509. The DNN system was trained using the senone alignments from the GMM-HMM system.

4.2. Results: Stochastic gradient implementation of the adaptation techniques

Table 1 reports the WERs by GMM-HMMs and DNNs. The baseline GMM-HMM has WER of 43.6% on average of the test set. Using DNN, the WER is reduced to 34.1%, corresponding to 21.8% relative WER reduction (WERR). The table also shows results obtained using a stochastic gradient implementation of the affine transformations. Adapting on the top layer by the bias shift $\mathbf{c}$ reduces WERs to 29.4%, a 13.9% relative WERR compared to the speaker-independent DNN system. Using fDLR, the WERs are reduced to 28.5%, a 16.8% relative WERR. We also combine updating $\mathbf{c}$ on the top layer with fDLR. Using fDLR on top of updating $\mathbf{c}$, the WER is reduced to 28.3%.

Table 2. Word error rates on the top layer adaptation using batch updates implementations

Methods	WER
DNN + $\mathbf{c}$ (Eq. 10))	30.9
DNN + $diag\mathbf{M} + \mathbf{c}$ (Eqs. 9 and 10)	30.8
DNN + $\mathbf{a}$ (Eq. 14)	31.9

4.3. Results: Batch-update implementation of the adaptation techniques

Table 2 reports the WERs achieved by adapting the top layer. These results were obtained using batch updates[2]. The step sizes for estimating $\mathbf{c}$ and $\mathbf{M}$ are set to 0.1 and 0.01 respectively. The algorithms update estimates of affine transformations in 20 iterations. The results show that two variations of affine transformations achieve almost the same performances, reducing WERs approximately by 9.7% compared against the speaker-independent DNN system. This indicates that much of the performance gains are obtained by the bias shift $\mathbf{c}$. Adaptation on the log-linear model bias $\mathbf{a}$ directly reduces WERs to 31.9%, a relative 6.4% WERR. We conjecture that this is because when using affine transformations of the top hidden layer activations, the bias shift $\mathbf{c}$ is shared by all the senones and thus can be more reliably estimated than the log-linear model bias $\mathbf{a}$, whose element can be robustly estimated only if the corresponding senone is observed many times in the adaptation data. However, numerical effects can also cause performance differences so further experiments on other datasets may be necessary.

5. CONCLUSIONS AND DISCUSSIONS

Adaptation of neural networks is known to be more difficult than the GMM counterpart for ASR. In this paper we report our recent investigation on several adaptation techniques for CD-DNN-HMM based ASR. The techniques we have developed work by transforming the outputs of the final hidden layer of the DNN, amounting to modifying parameters of the log-linear (softmax) model at the final layer of the DNN. We have developed two types of implementation for the above adaptation techniques. Both implementation types lead to significant reduction of ASR word error rates on top of a baseline DNN-HMM system. The batch update of the adaptation method achieves 10% relative WER reduction. A stochastic gradient ascent implementation of the method reduces the WER by 14% relatively. We also evaluated the fDLR method, leading to 17% relative WERR. These results seem to suggest that adaptation on CD-DNN-HMMs not only is possible but also can be very effective at least on some LVSR tasks.

Note that the results may be further improved by sharing the affine transformations based on regression trees. We also plan to investigate the effectiveness of affine transformations at other layers in the DNN than have reported in this paper.

[2] A pilot experiment indicated that, to estimate affine transformations, silence segments should not be excluded. Notice that this is different from fMLLR estimation that usually excludes silence segments.

6. REFERENCES

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

[2] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Processing Magazine*, 2012.

[3] X. Huang, A. Acero, and H.-W. Hong, *Spoken Language Processing: a guide to theory, algorithm, and system development*, Prentice Hall, 2001.

[4] B. Li and K. C. Sim, "Comparison of discriminative input and output transformations for speaker adaptation in the hybrid NN/HMM systems," in *INTERSPEECH*, 2010, pp. 526–529.

[5] R. Gemello, F. Mana, S. Scanzio, P. Laface, and R. D. Mori, "Adaptation of hybrid ANN/HMM models using linear hidden transformations and conservative training," in *ICASSP*, 2006, pp. 1189–1192.

[6] S. M. Siniscalchi, J. Li, and C.-H. Lee, "Hermitian based hidden activation functions for adaptation of hybrid HMM/ANN models," in *INTERSPEECH*, 2012.

[7] J. Neto *et al*, "Speaker-adaptation for hybrid HMM-ANN continuous speech recognition system," in *EUROSPEECH*, 1995.

[8] Arthur Earl Bryson and Yu-Chi Ho, *Applied optimal control: optimization, estimation, and control*, p. 481, Blaisdell Publishing Company or Xerox College Publishing, 1969.

[9] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *ASRU*, 2011.

[10] P. Zhan *etal*, "Vocal tract length normalization for LVCSR," in *Tech. Rep. CMU-LTI-97-150*. Carnegie Mellon University, 1997.

[11] M. J. F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer, Speech and Language*, vol. 12, pp. 75–98, 1998.

[12] V. Abrash, H. Franco, A. Sankar, and M. Cohen, "Connectionist speaker normalization and adaptation," in *EUROSPEECH*, 1995.

[13] A. Mohamed, G. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 14–22, 2012.

[14] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, pp. 1771–1800, 2002.

[15] G. Heigold, H. Ney, P. Lehnen, T. Gass, and R. Schluter, "Equivalence of generative and log-linear models," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 19, no. 5, pp. 1138–1148, 2011.

[16] X. Li and J. Bilmes, "Regularized adaptation of discriminative classifiers," in *ICASSP*, 2006.

[17] N. Kumar and A. G. Andreou, "A generalization of linear dis-criminant analysis in maximum likelihood framework," in *Tech. Rep. JHU-CLSP Technical Report*. Johns Hopkins University, Aug 1996, vol. 16.