Asynchronous Stochastic Optimization for Sequence Training of Deep Neural Networks: Towards Big Data

Erik McDermott, Georg Heigold, Pedro Moreno, Andrew Senior & Michiel Bacchiani

Google Inc.
Mountain View & New York, USA
{erikmcd,heigold,pedro,andrewsenior,michiel}@google.com

Abstract

Previous work presented a proof of concept for sequence training of deep neural networks (DNNs) using asynchronous stochastic optimization, mainly focusing on a small-scale task. The approach offers the potential to leverage both the efficiency of stochastic gradient descent and the scalability of parallel computation. This study presents results for four different voice search tasks to confirm the effectiveness and efficiency of the proposed framework across different conditions: amount of data (from 60 hours to 20,000 hours), type of speech (read speech vs. spontaneous speech), quality of data (supervised vs. unsupervised data), and language. Significant gains over baselines (DNNs trained at the frame level) are found to hold across these conditions. The experimental results are analyzed, and additional practical details for the approach are provided. Furthermore, different sequence training criteria are compared.
Index Terms: deep neural networks, sequence training, asynchronous stochastic gradient descent, unsupervised training.

1. Introduction

This study significantly expands the evaluation of the proposal in [1] for sequence training based on asynchronous stochastic optimization. The context for this work is as follows.

In the last few years, Deep Neural Networks (DNNs) have replaced Gaussian Mixture Models (GMMs) as the state-of-the-art acoustic model (AM) for automatic speech recognition (ASR) [2]. DNN optimization based on a Cross-Entropy (CE) loss function reflecting classification of temporally local speech frames into context-dependent states for using in a "hybrid" DNN/HMM system [3] is a simple, easily implemented approach used successfully in several studies [4] [5].

Simplicity notwithstanding, frame-discriminative training is not guaranteed to optimize the ultimate target, word accuracy over the entire utterance, decoded with a full language model (LM) [6] [7]. "Sequence" discriminative training, based on cost functions defined at the utterance level and reflecting actual performance of the speech recognition process [8] [9] [10] [11] [12], is therefore still highly relevant. The first description of DNN optimization using an utterance-level Maximum Mutual Information (MMI) criterion incorporating large-vocabulary word recogntion may be that in [13]. More recent work [14] re-ignited the topic of sequence training for DNNs, with active further research [15] [16] [17], among other studies, most confirming the effectiveness of sequence-discriminative training for DNNs.

A focal issue for sequence training of DNNs is the optimization method. Two distinct approaches stand out: Stochastic Gradient Descent (SGD) [18], typically implemented on a single GPU [5] [15], and batch-based second-order optimization [17]. Conventional SGD has remarkable convergence properties [19], converging quickly in terms of number of steps, but must be run on a single processor, limiting scalability, and suffers from limited data shuffling given the utterance-level chunking of sequence-discriminative optimization criteria [15]. The second approach can be parallelized over many machines and does not need data shuffling, but requires far more steps to converge, resulting in better, but still limited, scalability.

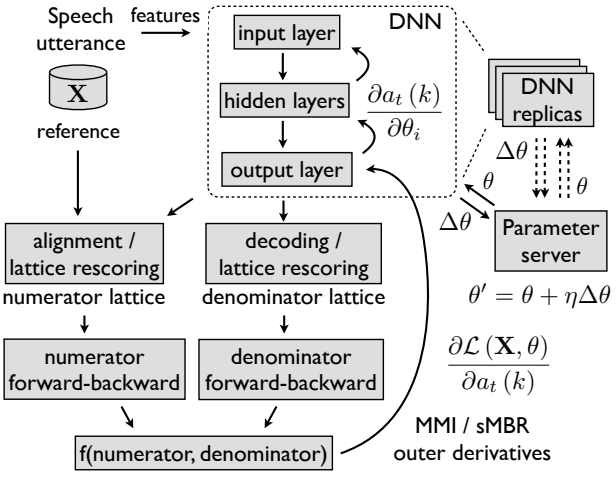

Figure 1: Asynchronous, stochastic sequence training of DNNs using a parameter server and model replication.

In contrast, the approach described in [1] for sequence training via *Asynchronous Stochastic Gradient Descent (ASGD)* attempts to bring the best of both worlds to sequence training: rapid convergence *and* scalability through parallel processing [20]. The study in [1] offers both theoretical and empirical support for the approach's salient features such as (1) superior data shuffling, (2) tolerance of additional sources of asynchrony arising for sequence training and (3) good resulting scalability and rapid convergence. The first evaluation focused on a small, supervised, read speech task, and the use of just MMI as the objective function. Here, the evaluation is substantially expanded to assess whether the approach scales to additional languages and much larger datasets (both supervised and unsupervised). Results are also presented for additional optimization criteria, state-level Minimum Bayes Risk (sMBR) [14] [21] and the MMI variant proposed in [16].

ASGD is well suited to the use of very large data sets.

Copyright © 2014 ISCA

Learning is incremental, and *convergence doesn't depend on data set size* [19] [22]. For small data sets, training tokens are typically recycled, possibly leading to generalization issues (that would arise for batch training too), but for large datasets, learning keeps going, often resulting in good convergence without full use of all training data. The training set is flexibly leveraged, as needed. This is a highly desirable property for the Big Data era of speech recognition.

Tackling truly "Big" datasets inevitably raises the topic of unsupervised training, which in turn raises the question of whether sequence training can be effective in that context [23]. Here this issue is addressed with evaluation on 20,000 hour unsupervised datasets for two languages. This may relate to the choice of optimization criterion. Given the difficulty of guaranteeing high quality transcripts for large training datasets, the ability of a criterion to filter out utterances with poor transcripts, far away from decision boundaries, is of great interest.

Paper organization: The asynchronous optimization framework for sequence training introduced in [1] is outlined, focusing on key features and advantages and providing additional practical details. Objective functions for both the frame-level CE baseline and sequence training criteria are summarized. Experimental results for 4 datasets of different quality, type and quantity are then presented and analyzed.

2. Sequence training using asynchronous stochastic optimization

In essence, [1] adapts the general asynchronous optimization framework described in [24] to the specific case of sequence training. A brief overview follows.

2.1. Asynchronous stochastic optimization

Though the effectiveness of SGD has been known for decades [18][19], schemes to scale it via parallelization have run into the fundamental problem of memory-locking and synchronous model updates, which drastically slows down learning performance. It was shown in [20] that for sparse problems, where only a fraction of the coordinates of the gradient vector are nonzero for any given training example, SGD can be implemented as ASGD, i.e. without memory-locking during model updates, outperforming other approaches by an order of magnitude. This was one of the motivations for the large-scale optimization software framework based on ASGD described in [24]. This framework was successfully applied to frame-level, CE based optimization of DNNs for ASR [24] [25]. The key features are:

- A *parameter server* that (1) Holds a snapshot of the current parameter set, which it can communicate on request; (2) Updates the parameter set, given incoming parameter gradients and a learning rule;
- A set of model *replicas* that (1) Make requests to the parameter server to obtain the latest version of the parameter set; (2) Update a local representation of the cost function gradient according to their local parameter set and the data shard being processed; (3) Communicate their local gradient back to the parameter server.

Crucially, the parameter set updating is lock-free, i.e. asynchronous. The replicas are not made to wait for the latest version to be updated; rather, they compute derivatives using a model that is typically out of date. The asynchrony allows for faster optimization, at the risk of divergence if the parameter updates are too big [20].

2.2. Extension to sequence training

It has up to now been an open question whether this optimization strategy would work for sequence training, with its by-definition utterance-level chunking of data. In this context, sequence training poses significant new challenges, in particular, (1) How frequently to update the "decoder" parameters, used to compute the sequence training outer derivatives, and (2) How to shuffle the data properly.

Concerning the first point, [1] uses a weak-sense auxiliary function to model the situation, and reports an empirical comparison of different decoder parameter update schedules. Regarding shuffling, each replica in [1] manages a pool of randomized utterances for which all frame-wise outer derivatives are computed. The replica then selects a batch of N frames, one frame per utterance, and aggregates a corresponding total gradient, that is then sent to the parameter server for model updating. As each replica uses the same procedure, the overall shuffling achieved is tremendously amplified by the number of replicas. This property, enabled by the asynchrony of the approach, is particularly beneficial to sequence training, as it circumvents the utterance-level chunking of the objective function.

Some simple heuristics are applied to filter frames before passing them to the optimization process. *Minimum posterior filtering* (with a threshold of 0.01) is applied to reject states where numerator and denominator posteriors have essentially canceled out. *Missing reference filtering* [16] can also be applied to MMI ("MMI-FR"). *Silence filtering* [15] was not found to be necessary.

The lattices used for MMI and sMBR are generated *on the fly*. This offers two important advantages:

- *Freshness:* barring search errors, the lattices reflect the current AM. This reduces the need for heuristics to cope with mismatches between model and lattices [15].
- *Processing of Big Data:* On very large data sets, ASGD may converge well before all training tokens are used. There is then no advantage to pre-generating lattices for all training data, and in fact doing so would be wasteful.

2.3. ASGD via Adagrad

Given the total gradient, the parameter set can be updated using a number of techniques. The classic SGD update [18][19] is

$$\theta_{u+1} = \theta_u - \eta_u \nabla \mathcal{L}\left(\mathbf{x}_u, \theta_u\right), \tag{1}$$

with a number of different variants for setting the learning rate η_u possible both in theory [18] and in practice [24]. One technique found to increase the convergence speed and possibly the robustness of ASGD is the Adagrad [26] adaptive learning rate procedure. Rather than use a single fixed learning rate on the parameter sever (in Fig. 1), Adagrad uses a separate adaptive learning rate for each parameter. Let $\eta_{i,u}$ be the learning rate of the i-th parameter θ_i at update step u and $\Delta\theta_{i,u}$ its gradient, then the rule is to set: $\eta_{i,u} = \gamma / \sqrt{\sum_{v=1}^{u} \left(\Delta\theta_{i,v}\right)^2}$. Adagrad can converge prematurely, as the learning rate decreases with the number of steps. This was not found to be an issue for sequence training when starting from a CE-trained DNN.

2.4. Chaining of outer & inner derivatives

The essential component of all gradient-based optimization approaches [27] [28] is the construction of the total gradient of the cost function $\mathcal{L}\left(x, \theta\right)$ w.r.t. individual model parameters θ_i, for a given network input x.

The popular "hybrid" approach for using DNNs as the AM in an HMM-based recognition system [1] [3] [14] [15] [16] is to define the posterior for an HMM state k, given any one of T feature vectors $\mathbf{x}_t$ from an utterance $\mathbf{X}$, in terms of the last layer activations $a_t(k)$ and resulting network outputs $y_t(k)$:

$$P(k|\mathbf{x}_t) \triangleq y_t(k) = \frac{\exp(a_t(k))}{\sum_{k'=1}^{N} \exp(a_t(k'))}, \quad (2)$$

where N is the number of states. Normalization of the posterior by the state prior is then used to define state log-likelihood:

$$\log p(\mathbf{x}_t|k) \triangleq \log P(k|\mathbf{x}_t) - \log P(k). \quad (3)$$

The criteria considered here make different uses of the same underlying, DNN-based $\log p(\mathbf{x}_t|k)$. The total gradient can be decomposed into inner and outer derivatives via the $a_t(k)$:

$$\frac{\partial \mathcal{L}(\mathbf{X},\theta)}{\partial \theta_i} = \sum_{t=1}^{T}\sum_{k}^{N} \frac{\partial \mathcal{L}(\mathbf{X},\theta)}{\partial a_t(k)} \frac{\partial a_t(k)}{\partial \theta_i}. \quad (4)$$

Different optimization criteria will induce different outer derivatives $\partial \mathcal{L}(\mathbf{X},\theta)/\partial a_t(k)$, while preserving the same inner derivative, $\partial a_t(k)/\partial \theta_i$ [29]. The chaining of outer and inner derivatives to produce the total parameter gradient that is sent to the parameter server is illustrated in Fig. 1.

2.5. Optimization criteria

In the following, different optimization criteria and their outer derivatives are briefly summarized. The reader is referred to [1] [14] [15] [16] for more details and derivations.

2.5.1. Frame-level Cross-Entropy (CE)

The baseline in this study is the simple frame-level CE optimization criterion that helped propel DNN recognition accuracies past most GMM baselines [4] [5].

$$\mathcal{L}_{XENT}(\mathbf{X},\theta) = \sum_{t=1}^{T}\sum_{k=1}^{N} \hat{y}_t(k) \log \frac{\hat{y}_t(k)}{y_t(k)} \quad (5)$$

with outer derivative

$$\frac{\partial \mathcal{L}_{XENT}(\mathbf{X},\theta)}{\partial a_t(k)} = y_t(k) - \hat{y}_t(k). \quad (6)$$

The $\hat{y}_t$ values are context-dependent HMM state *targets*, obtained e.g. using a forced-alignment of the utterance transcript.

2.5.2. Maximum Mutual Information (MMI) for sequence training

For DNNs, the log-likelihood for $\mathbf{X}$ given a word string S_j with best Viterbi state sequence $s_j(t)$ is defined as

$$\log p(\mathbf{X}|S_j) = \sum_{t=1}^{T} \log p(\mathbf{x}_t|s_j(t)). \quad (7)$$

Closely related to CE, the MMI criterion [8] can then be defined for $\mathbf{X}$ and the reference word string S_r:

$$\mathcal{L}_{MMI}(\mathbf{X},\theta) = -\log \frac{p(\mathbf{X}|S_r)^{\kappa} P(S_r)}{\sum_j p(\mathbf{X}|S_j)^{\kappa} P(S_j)} \quad (8)$$

It can be shown [11] [14] that the MMI outer derivatives are

$$\frac{\partial \mathcal{L}_{MMI}(\mathbf{X},\theta)}{\partial a_t(k)} = \kappa\left(\gamma_t^{den}(k) - \gamma_t^{num}(k)\right), \quad (9)$$

where $\gamma_t^{den}(k)$ and $\gamma_t^{num}(k)$ are denominator and numerator lattice occupancies for state k, respectively. These statistics can be obtained efficiently from lattices represented as Weighted Finite State Transducers (WFSTs) using the Forward-Backward algorithm on the Log semiring [30].

Boosted MMI [16] was evaluated but not found to give any gains; these results are not detailed here.

2.5.3. MMI with Frame Rejection (MMI-FR) for "missing" reference

The MMI criterion (Equ. (8)) diverges when the reference posterior goes to zero [30]. This means the criterion might be susceptible to instability when trained on poorly transcribed utterances. A simple heuristic is simply to skip frames where the reference state mass in the denominator is too small [16].

2.5.4. State-level Minimum Bayes Risk (sMBR)

The sMBR criterion [21] is in the family of Expected Error cost functions [12]:

$$\mathcal{L}_{sMBR}(\mathbf{X},\theta) = \bar{E}(\mathbf{X}) \triangleq \frac{\sum_j p(\mathbf{X}|S_j)^{\kappa} P(S_j) E(S_j,S_r)}{\sum_{j'} p(\mathbf{X}|S_{j'})^{\kappa} P(S_{j'})}. \quad (10)$$

$E(S_j,S_r)$ is a frame-wise state alignment matching error [14]. It can be shown [14] that the sMBR outer derivatives are

$$\frac{\partial \mathcal{L}_{sMBR}(\mathbf{X},\theta)}{\partial a_t(k)} = \kappa \gamma_t^{den}(k)\left(\bar{E}(\mathbf{X},k,t) - \bar{E}(\mathbf{X})\right), \quad (11)$$

where $\bar{E}(\mathbf{X},k,t)$ is expected error through lattice state k at time t, and $\bar{E}(\mathbf{X})$ is overall expected error for the utterance. These statistics (for both state occupancy and expected error) can be obtained efficiently from lattice WFSTs using the Forward-Backward algorithm on the Expectation semiring [30]. sMBR is less prone to the "missing" reference problem than MMI, as the derivative goes to zero away from decision boundaries [30] [31] [32].

3. Databases

All datasets are anonymized. The following training sets were used: *Icelandic (is_IS)*: 60 hours of read speech, simulating Voice Search queries; *American English (en_US)*: 3000 hours of spontaneous speech from live Voice Search data, human-transcribed; *French (fr_FR)* and *Russian (ru_RU)*: 20,000 hours each of unsupervised speech, described in the following. The test sets for each language are human-transcribed Voice Search datasets of about 25 hours each.

The unsupervised training sets for ru_RU and fr_FR were harvested from an original set of 300M utterances (approximately 30 years worth of audio), selected at random from three months of live Voice Search traffic. The utterances are redecoded with the best acoustic model available, with generous decoding parameters (beam size and maximum number of arcs), and using a larger LM than the systems used in production. In order to avoid too-frequent queries (like "facebook") and to increase triphone coverage and phonetic diversity, at most 50 utterances are kept with the same transcription. The redecoded utterances are sorted according to confidence score (average frame posterior probability, excluding silence segments). Finally, the top 20M utterances (about 20,000 hours audio) according to confidence score are selected.

Table 1: *Word error rates for sequence-trained DNNs on Voice Search tasks in several languages using different optimization criteria.*

Training data				DNN Configuration		Optimization Criterion WER				Training Time	
Lang.	Type	Quality	Quantity	Topology	# Wts.	CE	MMI	MMI-FR	sMBR	CE	Seq.Tr.
is_IS	Read	Sup.	60h	26x40:4x1024:2.5K	7M	13.0%	11.9%	**11.3**%	12.3%	12h	12h
en_US	Spont.	Sup.	3,000h	26x40:8x2560:14K	85M	11.3%	11.7%	10.5%	**10.4**%	4d	4d
ru_RU	Spont.	Unsup.	20,000h	26x40:8x2560:18K	95M	24.8%	23.4 %	**23.3**%	23.6%	9d	4d
fr_FR	Spont.	Unsup.	20,000h	26x40:8x2560:23K	108M	13.6%	**12.2**%	12.3%	12.6%	9d	4d

4. Experiments

Asynchronous stochastic optimization using each criterion was run for each training set. Rectified linear unit activations and ADAGRAD are used throughout. The Adagrad learning rate was initialized to be as large as possible without causing divergence. Fifty multi-core machines were used for is_IS and en_US, 100 multi-core machines for fr_FR and ru_RU. DNN configurations ("CxF:LxN:O", where C, F, L, N and O are number of context frames, features, hidden layers, nodes per hidden layer, and output states, respectively), and corresponding approximate total number of parameters are shown in Table 1.

4.1. Frame-level Cross-Entropy baselines

CE was run on DNNs initialized with random weights and using force-aligned utterances to generate the targets in Equ. (5). For is_IS, several epochs (full presentations of the entire training set) are used; for the much larger training sets, about one epoch is performed. A batch size of 200 frames was used.

4.2. Sequence training

The LM used to make denominator lattices on-the-fly is a unigram (is_IS, ru_RU, fr_FR) or bigram (en_US) LM made from the training utterance transcripts. This LM was also used on a held-out set to help gauge convergence during training.

Training was performed using MMI, MMI-FR, and sMBR. For is_IS, several training epochs are again used. Remarkably, on en_US, training has largely converged after using just 50% of the data; that is the model used in the WER evaluation here. Similarly, for ru_RU and fr_FR, training has largely converged after about 20% of the data is seen; that is the model used in WER evaluation. Note that training beyond these points continues to improve the models on held out data; the generalization of ASGD is excellent.

It was found that minimum posterior filtering (Section 2.2), used with all sequence training criteria), rejects 85-90% of all utterance frames for which numerator and denominator outer derivatives essentially cancel, speeding up training considerably. This sparseness may make sequence training well-suited to optimization using ASGD [20]. A batch size of 32 frames after filtering was used. The overall processing time per learning step for sequence training is about 2X that for CE. Approximate training times and test WER for both CE and sequence training are shown in Table 1.

5. Discussion

The results described here offer strong experimental confirmation for the effectiveness of the asynchronous stochastic optimization approach to sequence training first presented in [1]. The most salient result is simply that the approach scales well to much larger datasets than examined before. These results were obtained with 50-100 multi-core machines, and an efficient parameter communication framework [24]. Further assisted by powerful data shuffling, training converges rapidly, making sequence training on datasets of up to 20,000 hours practical.

A related strong point is that sequence training on the larger datasets converges *without full presentation of the training sets*. This is a good example of the flexible leveraging of redundancy in the training set afforded by stochastic optimization. This supports the use of on-the-fly lattice generation: it would be wasteful to pre-compute lattices for training tokens that will not be needed. Superficially, the finding suggests that we don't need more than 3,000-4,000 hours of training data, but the sequence training results are for CE-initialized models that were trained from scratch using the full training sets, and for specific datasets and model sizes. Sequence training from scratch, use of larger models, and different datasets, may require use of more data, that again should be leveraged flexibly.

Good results were obtained for the large unsupervised datasets; the aggressive filtering and data-balancing heuristics appear to be effective. The results for the different optimization criteria can be related to data filtering. Both sMBR and MMI-FR have built-in data filters, lessening the impact of frames with low reference posteriors, while standard MMI does not. On the unsupervised fr_FR and ru_RU sets, the automatically generated transcripts already pass a strong confidence filter, so one could surmise that there wouldn't be a difference between standard MMI and MMI-FR. It appears it is the human-generated transcripts that need criterion-based filtering. On the supervised en_US data, standard MMI is worse than CE, while both MMI-FR and sMBR yield good results, suggesting some sort of missing reference filter is critical. On the much smaller supervised is_IS data, MMI-FR yields the best result, while here it is sMBR that seems underpowered. Only on en_US does sMBR give the best result. Compared to sMBR, MMI-FR may offer better combination of strong weighting of the reference up to a point on the incorrect side of the decision boundary, and rejection beyond that point (cf Minimum Classification Error [27] and "Differenced MMI" [32]). Further experiments are needed to make firmer conclusions.

6. Conclusions

This paper extended the evaluation of the asychronous stochastic optimization framework for sequence training described in previous work to far larger datasets and additional optimization criteria. The significant gains in word error obtained suggest that not only does the approach scale well to very large datasets, but it is capable of leveraging unsupervised data effectively. Aided by asynchrony- and replica- amplified randomization of training data (particularly difficult to achieve for sequence training), training converges well even without full use of the available set, showing excellent, highly practical use of data redundancy. These are all attractive properties for addressing the challenges of "Big Data" speech recognition.

7. References

[1] G. Heigold, E. McDermott, V. Vanhoucke, A. Senior, and M. Bacchiani, "Asynchronous Stochastic Optimization for Sequence Training of Deep Neural Networks," in *Proc. IEEE ICASSP*, 2014.

[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," in *IEEE Signal Processing Magazine*, 2012.

[3] H. Bourlard and N. Morgan, *Connectionist speech recognition*, Kluwer Academic Publishers, 1994.

[4] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *Proc. ASRU*, 2011, pp. 24–28.

[5] N. Jaitly, P. Nguyen, A. Senior, and V. Vanhoucke, "Application of pretrained deep neural networks to large vocabulary speech recognition," in *Proc. Interspeech*, 2012.

[6] E. McDermott and S. Katagiri, "Prototype-Based Discriminative Training for Various Speech Units," in *Proc. IEEE ICASSP*, March 1992, vol. 1, pp. 417–420.

[7] D. Povey and P.C. Woodland, "Frame Discrimination Training Of HMMs For Large Vocabulary Speech Recognition," in *Proc. IEEE ICASSP*, 1999, pp. 333–336.

[8] P. Brown, *The acoustic-modeling problem in automatic speech recognition*, Ph.D. thesis, Carnegie Mellon University, Department of Computer Science, 1987.

[9] Y. Normandin, *Hidden Markov Models, Maximum Mutual Information Estimation, and the Speech Recognition Problem*, Ph.D. thesis, McGill University, Montreal, Department of Electrical Engineering, 1991.

[10] S. Katagiri, C-H. Lee, and B.-H. Juang, "New discriminative training algorithms based on the generalized descent method," in *Proc. IEEE Worshop on Neural Networks for Signal Processing*, 1991, pp. 299–308.

[11] P.C. Woodland and D. Povey, "Large scale discriminative training of hidden Markov models for speech recognition," *Computer Speech and Language*, vol. 16, pp. 25–47, 2002.

[12] D. Povey and P.C. Woodland, "Minimum Phone Error and I-smoothing for improved discriminative training," in *Proc. IEEE ICASSP*, 2002, vol. 1, pp. 105–108.

[13] P. Haffner, "Connectionist speech recognition with a global MMI algorithm," in *EUROSPEECH*. 1993, ISCA.

[14] Brian Kingsbury, "Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling," in *Proc. IEEE ICASSP*, 2009, pp. 3761–3764.

[15] H. Su, G. Li, D. Yu, and F. Seide, "Error back propagation for sequence training of context-dependent deep networks for conversational speech transcription," in *Proc. IEEE ICASSP*, 2013, pp. 6664–6668.

[16] K. Vesely, A. Ghoshal, L. Burget, and D. Povey, "Sequence-discriminative training of deep neural networks," in *Proc. Interspeech*, 2013, pp. 2345–2349.

[17] B. Kingsbury, T. Sainath, and H. Soltau, "Scalable minimum bayes risk training of deep neural network acoustic models using distributed hessian-free optimization," in *Proc. IEEE ICASSP*, 2012.

[18] S. Amari, "A Theory of Adaptive Pattern Classifiers," *IEEE Trans. on Electronic Computers*, vol. EC-16, no. 3, pp. 299–307, 1967.

[19] Léon Bottou, "Online algorithms and stochastic approximations," in *Online Learning and Neural Networks*, David Saad, Ed. Cambridge University Press, Cambridge, UK, 1998, revised, oct 2012.

[20] F. Niu, B. Recht, C. Ré, and S. J. Wright, "Hogwild! a lock-free approach to parallelizing stochastic gradient descent," in *Advances in Neural Information Processing Systems (NIPS)*, 2011.

[21] J. Kaiser, B. Horvat, and Z. Kačič, "A novel loss function for the overall risk criterion based discriminative training of HMM models," in *International Conference on Spoken Language Processing*, 2000.

[22] N. Srebro, "SVM optimization: Inverse dependence on training set size," in *International Conference on Machine Learning(ICML)*, 2008.

[23] Y. Huang, D. Yu, Y. Gong, and C. Liu, "Semi-Supervised GMM and DNN Acoustic Model Training with Multi-system Combination and Confidence Re-calibration," in *Proc. Interspeech*, 2013.

[24] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, Q. Le, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, and A. Ng, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[25] G. Heigold, V. Vanhoucke, A. Senior, P. Nguyen, M. Ranzato, M. Devin, and J. Dean, "Multilingual acoustic models using distributed deep neural networks," in *Proc. IEEE ICASSP*, Vancouver, Canada, Apr. 2013, vol. 1.

[26] J.C. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

[27] E. McDermott, T.J. Hazen, J. Le Roux, A. Nakamura, and S. Katagiri, "Discriminative training for large vocabulary speech recognition using Minimum Classification Error," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 1, pp. 203–223, January 2007.

[28] V. Valtchev, *Discriminative Methods in HMM-based Speech Recognition*, Ph.D. thesis, University of Cambridge, 1995.

[29] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[30] Georg Heigold, *A log-linear discriminative modeling framework for speech recognition.*, Ph.D. thesis, RWTH Aachen University, 2010, http://d-nb.info/1007980575.

[31] D. Povey, *Discriminative Training for Large Vocabulary Speech Recognition*, Ph.D. thesis, University of Cambridge, 2004.

[32] E. McDermott, S. Watanabe, and A. Nakamura, "Discriminative training based on an integrated view of MPE and MMI in margin and error space," in *Proc. IEEE ICASSP*, 2010, pp. 4894–4897.