# Deep Learning of Neuromuscular Control for Biomechanical Human Animation

Masaki Nakada(✉) and Demetri Terzopoulos

Computer Science Department, University of California, Los Angeles, USA
nakada@cs.ucla.edu

**Abstract.** Increasingly complex physics-based models enhance the realism of character animation in computer graphics, but they pose difficult motor control challenges. This is especially the case when controlling a biomechanically simulated virtual human with an anatomically realistic structure that is actuated in a natural manner by a multitude of contractile muscles. Graphics researchers have pursued machine learning approaches to neuromuscular control, but traditional neural network learning methods suffer limitations when applied to complex biomechanical models and their associated high-dimensional training datasets. We demonstrate that "deep learning" is a useful approach to training neuromuscular controllers for biomechanical character animation. In particular, we propose a deep neural network architecture that can effectively and efficiently control (online) a dynamic musculoskeletal model of the human neck-head-face complex after having learned (offline) a high-dimensional map relating head orientation changes to neck muscle activations. To our knowledge, this is the first application of deep learning to biomechanical human animation with a muscle-driven model.

## 1 Introduction

The modeling of graphical characters based on human anatomy is becoming increasingly important in the field of computer animation. Progressive fidelity in biomechanical modeling should, in principle, result in more realistic human animation. Given realistic biomechanical models, however, we must confront a variety of difficult motor control problems due to the complexity of human anatomy.

In conjunction with the modeling of skeletal muscle [1,2], existing work in biomechanical human modeling has addressed the hand [3–5], torso [6–8], face [9–11], neck [12], etc. These prior efforts demonstrated different control schemes for individual parts of the human body. Further research is needed to achieve a fully robust control system for muscle-actuated biomechanical human animation.

The challenge in controlling biomechanical human models stems from anatomical intricacy, which complicates the kinematics, dynamics, and actuation of the character. For example, the neck-head biomechanical model developed in [12] has 10 bones and 72 muscle actuators (Fig. 2), while the full-body biomechanical model presented in [13] includes 103 bones and 823 muscle actuators,

G. Bebis et al. (Eds.): ISVC 2015, Part I, LNCS 9474, pp. 339–348, 2015.
DOI: 10.1007/978-3-319-27857-5_31

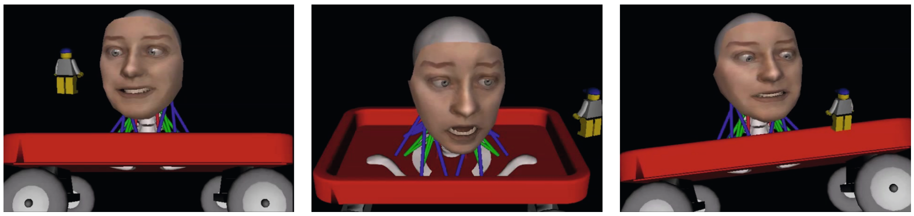

**Fig. 1.** Snapshots from an animation demonstrating robust online neuromuscular control of head orientation in a muscle-actuated biomechanical neck-head-face system. Despite the continuously varying orientation of its shoulders that are fixed to the unsteady wagon base, the character can visually track a moving target (doll) while naturally supporting the mass of its head in gravity atop its anatomically accurate flexible neck. Facial expressions are automatically produced in response to the observed movement of the doll—the character expresses awe when the doll is raised above the head, anger when the doll is shaken, and pleasure when the doll is held calmly at eye level.

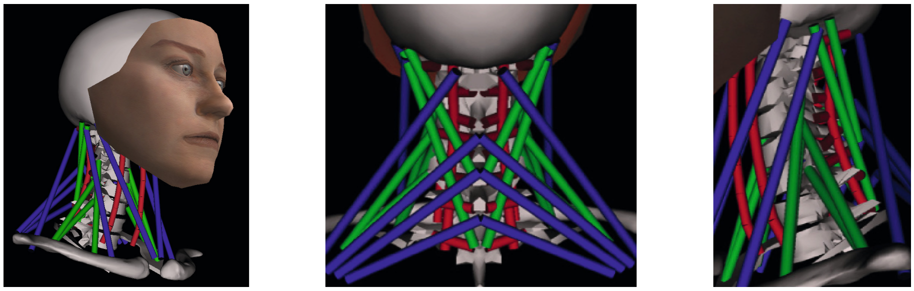

**Fig. 2.** Close-up views of the musculoskeletal structure of the biomechanical neck-head model, comprising 7 cervical vertebra and 72 actuators: 48 deep muscles (red), 12 intermediate muscles (green), and 12 superficial muscles (blue) (Color figure online).

an order of magnitude greater complexity. Controlling such models requires the specification of muscle innervations (i.e., activation signals) at every time step of the biomechanical simulation.

### 1.1 Neuromuscular Control

As pioneered by Lee and Terzopoulos [12] in the context of neck-head control, neuromuscular learning approaches are the most biomimetic and promising in tackling high-dimensional, muscle-actuated biomechanical motor control problems. To control the neck-head system in gravity, rather than resorting to traditional inverse dynamics control, which is both unnatural and computationally expensive, these authors employed a conventional, artificial neural network with two hidden layers of sigmoidal units, which they trained offline through back-propagation learning. The training data comprised tens of thousands of random target head orientations as network inputs and the muscle activation levels

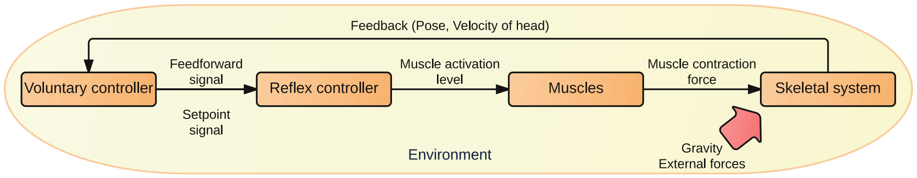

**Fig. 3.** Control flow for the neck-head model.

required to achieve these target orientations as associated network outputs. Their trained neuromuscular controller was capable of efficiently controlling the biomechanical model online, thereby achieving a real-time neuromuscular neck-head control system. The neck-head biomechanical model is actuated by 72 muscles, so their trained neural network controller maps 3 inputs (a target head orientation) to 72 outputs (the muscle activations required to achieve this orientation) with adequate generalization. Their *offline* training procedure took approximately 10 h to complete the backpropagation learning process.

Conventional neural networks with only a few hidden layers have not proven effective at coping with the higher dimensionalities and necessarily much greater quantities of training data required to learn to control biomechanical models of an order of magnitude greater complexity, such as the one presented in [13]. For this reason, Lee and Terzopoulos [12] made no attempt to generalize their neck-head controller to deal with other than horizontally-oriented shoulders and, unsurprisingly, their trained neuromuscular controller fails to maintain satisfactory control unless the shoulders remain nearly horizontal.

## 1.2 Deep Learning of Neuromuscular Control

Deep neural networks with many hidden layers could not at first be trained due to technical difficulties with the backpropagation learning algorithm such as local minima and the vanishing gradient problem. Motivated by recent breakthroughs in "deep learning" that have mitigated these problems, we have applied deep neural network architectures to neuromuscular control. The long-range goal of our research is to overcome the aforementioned challenges in the neuromuscular control of complex biomechanical models.

In this paper, we adapt deep learning techniques to the neuromuscular control of the human neck-head biomechanical model described in [12]. Our contributions include the introduction of a new neuromuscular controller, whose deep architecture differs markedly from the one in [12], with dramatically enhanced learning ability and efficiency even when employing much higher dimensional training data. This enables us to push significantly beyond the prior benchmark, training our neck-head neuromuscular controller to work well not merely for horizontal shoulders, but for a considerable range of shoulder orientations.

### 1.3 Overview

The remainder of this paper is organized as follows: In Sect. 2, we briefly explain the neck-head model and its neuromuscular control system. We then develop our novel neuromuscular learning approach in Sect. 3, and demonstrate the effectiveness of our approach through experiments reported in Sect. 4. Section 5 presents our conclusions and discusses future work.

## 2 Biomechanical Model and Control System

In our work, we use the anatomically accurate musculoskeletal model of the human neck-head complex implemented by [12], which is illustrated in Fig. 2. The model comprises a skull, whose mass is commensurate with that of an average adult male head, atop a cervical skeleton comprising 7 vertabrae connected by 3 degree-of-freedom rotational joints, and a clavicle base. This skeletal system is actuated by 72 contractile cervical muscles, modeled as Hill-type actuators, arranged in three layers (Fig. 2). Appendix 1 presents additional details about the biomechanical model.[1] The associated neuromuscular control architecture (Fig. 3) involves a high-level voluntary controller that determines the head pose and a lower-level reflex controller that produces the necessary muscle activation inputs. Appendix 2 presents additional details about the hierarchical neuromuscular control system, as well as the gaze control mechanism.

The specific technical challenge that we address in our work is how best to implement and train the voluntary controller. The considerable kinematic and muscular redundancy of the biomechanical model makes it infeasible to control the cervical muscles online in real time using an inverse dynamics approach. Instead, as in [12], we pursue a biomimetic neuromuscular control approach. A neural network controller is trained offline on a large dataset comprising several tens of thousands of random head orientations (inputs) and the associated muscle activations (outputs). To generate training data, each 72-dimensional muscle activation output, which must achieve the associated target head pose and sustain it against the pull of gravity, is synthesized off-line through (computationally expensive) inverse kinematics/dynamics computations as in [12].

Lee and Terzopoulos [12] used a conventional, shallow neural network for their voluntary controller. It typically took on the order of 10 h to train this network, which imposed practical limits on the dimensionality and size of the usable training dataset. To surmount this obstacle in neuromuscular control, we devise a deep neuromuscular control architecture that proves to be more efficient to train and is capable of learning more proficiently from larger and much higher-dimensional datasets. In principle, this makes it amenable to biomechanical models of much greater complexity—potentially even to a full-body biomechanical model with an order of magnitude more articular degrees of freedom and muscle actuators [8,13].

[1] Due to space limitations, the appendices are found in a supporting document (see, e.g., https://db.tt/wn37j8rg).

At each time-step of the biomechanical simulation, muscle activations are computed by our trained deep neural network in the voluntary controller, and they are input to the reflex controller. The latter adjusts the muscle activations in accordance with the difference between the current state and the desired state of each muscle. The biomechanical simulator computes the muscle forces induced by the muscle activations, applies the muscle forces to the dynamic skeletal system, and numerically updates the accelerations, velocities, and positions of the articulated bones, thereby advancing the simulation in time.

### 2.1 Egocentric Head-Eye Control

Humans obtain visual information about objects from their foveated retinal images. Sensing a target of interest in the periphery of the visual field triggers a rapid eye rotation to foveate the target in conjunction with a comparatively sluggish head rotation (due to the greater mass of the head) toward the target. Lee and Terzopoulos [12] synthesized desired head rotation trajectories through the spherical linear interpolation (slerp) of head Euler angles, which is unnatural.

By contrast, in our neuromuscular control system, we define the angular error vector as the difference of the current head orientation and the target head orientation. In each time step, our neuromuscular controller acts to reduce this 2-D angular error vector (with horizontal angle $\theta$ and vertical angle $\phi$). Our neck-head control system does not require knowledge of the position of the visual target nor of the orientation of the head relative to the world frame. Rather, the relevant variables are represented relative to the egocentric frame whose origin is at the head and which rotates with the head.

## 3 The Deep Neuromuscular Controller

Our neuromuscular controller differs substantially from the shallow (3-input, 72-output, 2-hidden-layer) network described in [12]. We map to the 72 muscle activation outputs not only the target head orientation inputs, but also the 72 current muscle activations, so that the current muscle activation state affects the computation of the output muscle activations. Incorporating the current muscle activations as inputs is necessary in order to reduce the angular error vector in egocentric coordinates while achieving robust control regardless of shoulder orientation. The higher input dimensionality of our controller along with the much greater quantity of data necessary to train it at a variety of shoulder orientations exceeds the capabilities of the shallow network used in [12], and this problem is unlikely to be overcome without the use of a deeper network.

Our proposed deep network architecture includes 74 inputs, comprising the 2 angles, $\theta$ and $\phi$, of the head orientation error vector plus the 72 current muscle activations, along with 72 muscle activation outputs. The intermediate, hidden layers of the network implement an $n$-stage stacked denoising autoencoder (SdA) [14], which will be described in the next section. The bottom layer outputs the modified 72 cervical muscle activations (illustrated in Appendix 3).

A more powerful neuromuscular controller learning method is needed to train our deep network. Our learning method has two phases. The first is a pre-training stage, where we apply unsupervised learning to the SdA. The second is a fine-tuning stage where we apply supervised learning to the multilayer perceptron. The first phase contributes to finding better initial parameters and the second phase tunes those parameters by comparing the output from the network and the training example values.

The overall learning process proceeds as follows: We generate a training dataset comprising many input/output examples by numerically simulating the biomechanical model. Then, we run the SdA pre-training phase on this dataset. After pre-training, we initiate fine-tuning with the multilayer perceptron. Subsequently, we can employ the trained deep network online to control the biomechanical model.

Additional technical details about our deep neuromuscular controller learning process are presented in the following sections.

## 3.1 Stacked Denoising Autoencoder (SdA)

An autoencoder is a multilayer neural network that minimizes the difference between its input and output after encoding/decoding through its hidden layers. The encoding is as follows:

$$\mathbf{y}_e = \boldsymbol{\sigma}(\mathbf{W}\mathbf{x} + \mathbf{b}), \tag{1}$$

where $\mathbf{x}$ is the input, $\mathbf{y}_e$ is the encoded output, $\boldsymbol{\sigma}$ is a nonlinearity with sigmoidal $\sigma(z) = 1/(1+e^{-z})$ components, $\mathbf{W}$ is the weight matrix, and $\mathbf{b}$ is the bias vector. Likewise for the decoder,

$$\mathbf{y}_d = \boldsymbol{\sigma}(\mathbf{W}'\mathbf{y}_e + \mathbf{b}'), \tag{2}$$

where $\mathbf{y}_d$ is the decoded value, and $\mathbf{W}'$ and $\mathbf{b}'$ are the weight matrix and bias vector. After encoding/decoding, the inputs are compared to the outputs, yielding the error function

$$E = \frac{1}{N}(\sum_{i=1}^{N} ||\mathbf{x}_i - \mathbf{y}_{di}||^2), \tag{3}$$

where $i$ indexes over the $N$ training examples. The error gradient $\nabla E$ is used to update the weights and biases through gradient descent (in our work, we set the learning rate to 0.1).

In mini-batch stochastic gradient descent, $\nabla E$ is estimated using a limited number of training examples. We include 32 training examples in each mini-batch. Processing all the mini-batches in this way constitutes a training epoch. We use a tied-weight scheme, $\mathbf{W}' = \mathbf{W}^T$, which has a beneficial regularization effect and requires less memory. Hence, the parameters to be updated are $\mathbf{W}$, $\mathbf{b}$, and $\mathbf{b}'$.

A Denoising Autoencoder (dA) forces its hidden layers to discover more robust features and prevent learning the identity mapping, which the simple autoencoder may uselessly determine as being the zero-error optimal solution.

To this end, one adds noise to the training data, and the dA tries to reduce the noise and recover the original data from the corrupted data. Associating a probability (0 implies certainty) with each unit in each layer, we set the probabilities to 0.1 for the hidden layers and randomly mask some inputs by setting them to zero.

We stack the dAs to form a Stacked Denoising Autoencoder (SdA), where the outputs of each dA are propagated as inputs to the next dA. The advantage of the SdA is that each dA is trained independently and sequentially, and the dAs may be stacked as deeply as necessary to produce a good output.

### 3.2 Pre-training Phase

During the pre-training phase, the training starts at the input layer of the SdA and advances to the output layer, completing the training one layer at a time. We run 300 epochs for each layer sequentially such that each dA can learn a good representation and the input data are mapped to the output data with minimal information loss.

The initial weights are uniformly sampled from the interval $[-v, v]$, where $v = \sqrt{6/(n_i + n_o)}$, with $n_i$ being the number of inputs and $n_o$ the number of outputs in a layer. It is known that uniform sampling from this interval works well with sigmoidal units [15]. This prevents the learning process from becoming trapped in a local minimum early in the pre-training phase and it yields better optimization and faster convergence.

### 3.3 Fine-Tuning Phase

During the fine-tuning phase, we start with the weights and biases obtained in the pre-training phase, and then we update them using the multilayer backpropagation algorithm. This phase is supervised, as we compare the outputs to the labeled training data and reduce the error by updating the weights and biases using mini-batch stochastic gradient descent with (mean sum-of-squared) error backpropagation.

The training process is stopped when it ceases to improve sufficiently on a validation dataset (we set the threshold in each training epoch to 0.005 %). This "early stopping" prevents the training process from overfitting.

## 4 Experiments and Results

### 4.1 Fixed, Horizontal Shoulder Orientation

Our first set of experiments evaluates the performance of our deep neuromuscular controller relative to the shallow one reported in [12] with fixed horizontal shoulder orientation. Figure 4 plots the learning performance of different network structures. The plots indicate the superiority of our deep neuromuscular controller learning method with pre-training. Appendix 4.1 presents the details of this experiment.

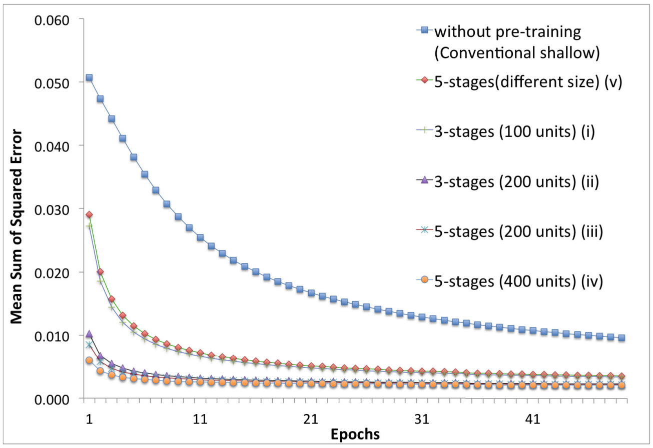

**Fig. 4.** Training error with different network structures when fine-tuning on the validation dataset.

## 4.2 Robust Control with Varying Shoulder Orientation

We verified that the shallow neuromuscular controller developed in [12] fails to maintain control of the head when the shoulders are not in a horizontal orientation. To facilitate comparison, the motion of the doll visual target is the same as the one in the head-eye gaze behavior demonstration illustrated in Fig. 10 of reference [12]. The more the orientation of the shoulders deviates from the horizontal, the more difficulty the shallow neuromuscular controller experiences in tracking the visual target. More strikingly, as the shoulders tilt back moderately, the neck-head skeleton collapses backward in gravity, resulting in catastrophic failure. We will next explain how we train our deep neuromuscular controller to accomplish this more challenging control task.

*Synthesis of Training Data:* First, with horizontal initial shoulder orientation, we synthesize training data by incrementing the head by 1° angles around the $\theta$ and $\phi$ axes in the range $-50° < \theta < 50°$ and $-50° < \phi < 50°$. Then, we repeat the same process with different shoulder orientations, varying the orientation of the shoulders by 5° from $-25°$ to 25° around both the frontal $x$ axis along the shoulders and the sagittal $z$ axis. This yields 1,000,000 example target angular error vectors with associated muscle activations computed through inverse kinematics/dynamics. The training data synthesis took 6.5 days to complete. Finally, we partition the data into training, validation, and testing datasets in the ratio 70 %:15 %:15 %, respectively.

We trained our deep neuromuscular controller with the aforementioned dataset and verified that, unlike the shallow neuromuscular controller, our controller enables the neck-head system to smoothly track the visual target while the orientation of the shoulders varies continuously in the range of $-25°$ to 25° around both the $x$ and $y$ axes. We used the 5-stage SdA network with 400 units in each stage—network (iv) in Fig. 4—since it proves to learn best as is shown in the figure.

Figure 1 shows snapshots from a demonstration video of our neck control system, demonstrating that it copes very well with continuously varying shoulder orientations. Although our training dataset was 33 times larger (and higher-dimensional) than the one used in [12], our off-line training process completed in only 23.0 h, with the pre-training phase taking 19.3 h and the fine-tuning phase taking 2.7 h.

The pre-training phase worked well with our training data, and our deep neural network learned the appropriate initial weights and biases after 300 epochs of training for each autoencoder layer. Subsequently, the fine-tuning phase started with a relatively small 0.021 error (the error would have been much higher had we simply set random initial weights and biases). This reduced the time for the fine-tuning phase, which completed with 0.003 error after satisfying the early stopping condition in 82 epochs.

For a fair comparison, we replicated the shallow neural network presented in [12] and tested it with our larger dataset. Appendix 4.2 includes a plot of the training progress. It took 2.5 days to complete 500 training epochs, and the error fluctuated between 0.0525 and 0.0545 without decreasing. None of the different parameter settings and shallow network structures that we tried resulted in successful convergence.

## 5 Conclusion and Future Work

We have introduced a new approach to the design and training of neuromuscular controllers of complex biomechanical human models for use in computer graphics animation. To our knowledge, this is the first application of deep neural networks to the problem. Our approach proves to be superior to previous methods both in terms of the efficiency of the learning process and, potentially more importantly, in the ability to learn effectively from much larger quantities of higher-dimensional training data. These benefits enabled us to train a deep neuromuscular neck-head controller to work well over a continuous range of shoulder orientations.

Our experiments demonstrated that our deep network, which incorporates significantly more units in its hidden layers, can learn effectively from our largest training datasets (in our case, one million input/output examples), reinforcing the notion that deep networks are generally more suitable than shallow networks for the regression, with good generalization, of massive quantities of training data.

We expect our deep learning approach to architecting and training neuromuscular controllers to work for other, similarly complex biomechanical models, such as those for arms and hands. Furthermore, our approach seems promising for the neuromuscular control of full-body human models with a dramatically higher dimensional muscle activation space, on the order of 1,000 muscle actuators.

## References

1. Ng-Thow-Hing, V.: Anatomically-based models for physical and geometric reconstruction of humans and other animals. Ph.D. thesis, University of Toronto, Computer Science Department (2001)
2. Irving, G., Teran, J., Fedkiw, R.: Invertible finite elements for robust simulation of large deformation. In: Proceedings of the ACM SIGGRAPH/EG Symposium on Computer Animation, p. 131. ACM Press, New York (2004)
3. van Nierop, O.A., van der Helm, A., Overbeeke, K.J., Djajadiningrat, T.J.: A natural human hand model. Vis. Comput. **24**, 31–44 (2007)
4. Tsang, W., Singh, K., Fiume, E.: Helping hand: an anatomically accurate inverse dynamics solution for unconstrained hand motion. In: Proceedings of the ACM SIGGRAPH/EG Symposium on Computer Animation, pp. 319–328 (2005)
5. Sueda, S., Kaufman, A., Pai, D.K.: Musculotendon simulation for hand animation. ACM Trans. Graph. **27**, 83 (2008)
6. Sifakis, E., Neverov, I., Fedkiw, R.: Automatic determination of facial muscle activations from sparse motion capture marker data. ACM Trans. Graph. **24**, 417–425 (2005)
7. DiLorenzo, P., Zordan, V., Sanders, B.: Laughing out loud: control for modeling anatomically inspired laughter using audio. ACM Trans. Graph. **27**, Article no. 125 (2008). ACM SIGGRAPH Asia 2008 Proceedings
8. Lee, S.H., Sifakis, E., Terzopoulos, D.: Comprehensive biomechanical modeling and simulation of the upper body. ACM Trans. Graph. **28**(99), 1–17 (2009)
9. Lee, Y., Terzopoulos, D., Waters, K.: Realistic modeling for facial animation. In: Proceedings of ACM SIGGRAPH 1995, pp. 55–62 (1995)
10. Kähler, K., Haber, J., Yamauchi, H., Seidel, H.: Head shop: generating animated head models with anatomical structure. In: Proceedings of the ACM SIGGRAPH/EG Symposium on Computer Animation, pp. 55–63 (2002)
11. Sifakis, E., Neverov, I., Fedkiw, R.: Automatic determination of facial muscle activations from sparse motion capture marker data. ACM Trans. Graph. **1**, 417–425 (2005)
12. Lee, S.H., Terzopoulos, D.: Heads up! Biomechanical modeling and neuromuscular control of the neck. ACM Trans. Graph. **23**, 1188–1198 (2006)
13. Si, W., Lee, S.H., Sifakis, E., Terzopoulos, D.: Realistic biomechanical simulation and control of human swimming. ACM Trans. Graph. **34**(10), 1–15 (2014)
14. Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H.: Greedy layer-wise training of deep networks. In: Advances in Neural Information Processing Systems 19 (NIPS 2007), pp. 153–160 (2007)
15. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: International Conference on Artificial Intelligence and Statistics, vol. 9, pp. 249–256 (2010)