Proceedings of the Twenty-Ninth International
Florida Artificial Intelligence Research Society Conference

A Scalable Unsupervised Deep Multimodal Learning System

Mohammed Shameer Iqbal and **Daniel L. Silver**
Jodrey School of Computer Science
Acadia University
Wolfville, Nova Scotia, Canada B4P 2R6
111271i@acadiau.ca, danny.silver@acadiau.ca

Abstract

We present an unsupervised multimodal learning system that scales linearly in the number of modalities. The system uses a generative Deep Belief Network for each modality channel and an associative network layer that relates all modalities to each other. The system is trained so as to reconstruct the output at all missing channels given input on one or more channels. The system uses a derivation of the back-fitting algorithm to fine-tuning just those weights leading to the associative layer from each channel. This allows the system to generate an appropriate representation at the associative layer and to scale linearly with the number of modalities. An experiment learning the numeric digits 0 through 9 from four sensory channels - audio, visual, motor and classification - demonstrates that the generative system can accurately reconstruct any channel from as few as one other channel.

Introduction

Humans receive several types of sensory data through different channels or modalities and are able to associate one modality with another. This fusion of sensory information channels allow us to learn concepts from a variety of features and to recognize objects even when information from one or more channels is missing. There are two significant challenges that must be faced when creating a machine learning system that can associate multiple modalities: (1) overcoming the differences in signal complexity of the channels, and (2) scaling the system up in terms of training time and memory with respect to the number of channels. We propose a generative Deep Learning Architecture (DLA) approach composed of Deep Belief Network (DBN) channels and an associative memory layer that is used to associate the channels. The weights of each channel to associative memory layer are fine-tuned using a back-fitting algorithm which allows the system to equalize the impact of the various channel signals and to scale linearly in the number of channels. Given the task of recognizing numeric digits 0 through 9 and the sensory channels audio, visual, motor and classification, we demonstrate that this generative system can accurately reconstruct any channel from as few as one other channel.

Background

The multimodal approach has been adopted by several deep learning researchers such as (Srivastava and Salakhutdinov 2012) and (Ngiam et al. 2011). Recently, development of multimodal systems that use images and text have become very active (Socher et al. 2014) (Karpathy and Fei-Fei 2015) (Kiros, Salakhutdinov, and Zemel 2014). However, these systems tend to associate only two modalities (such as shown in Figure 3). This allows the methods to fine-tune the internal representations and associate one layer with the other using techniques such as back-propagation. Unfortunately, such supervised techniques do not scale well to three or more modalities, because fine-tuning is needed for all possible input to output modality combinations. This problem is specifically recognized in (Ngiam et al. 2011). The authors propose a fine-tuning solution that converts the multimodal DLA into a deep autoencoder that has two input modalities (audio, video) and two matching output modalities (audio, video). The network is trained on a series of examples that contains either one or the other modalities or both modalities as inputs. This allows the fine-tuned model to generate appropriate reconstructions for both modalities when only one is provided or both are provided. Unfortunately, this solution has a scaling problem because the deep autoencoder is twice the size of the original multimodal DLA, and the number of training examples will grow exponentially with the number of modalities in order train on all combinations of input modalities.

A second problem that we have encountered when using strictly unsupervised learning methods for multimodal leaning is that one channel will dominate over the others in terms of developing the associative memory portion of the network. A channel that has a simple, noise free signal per class will provide the dominate signal to the associative layer. When this occurs, it is difficult for the network to properly generate the correct features at the associative layer when input is provided on one or more of the other channels.

We seek to overcome these two problems by using a variation of a *back-fitting* algorithm originally developed by Hinton et al. (Hinton et al. 1995).

Theory and Approach

We propose a multimodal deep learning neural network architecture that consists of multiple channels connected to a shared associative memory layer in the same manner as (Ngiam et al. 2011). Figure 1 shows an example of such a network. Training of the weights in this network is completely unsupervised.

Each sensory channel or modality is a generative Deep Belief Network (DBN) composed of stacked Restricted Boltzman Machine (RBM) layers whose weights are trained using the contrastive divergence algorithm (Hinton 2007). Each DBN channel learns multiple levels of abstraction of its input layer and is generative in nature. The number of hidden layers in a channel varies according to the complexity of the sensory data for that channel, as deeper architectures are necessary to learn higher order, non-linear transformations (Bengio 2009).

The top level features of each DBN is combined as input to another RBM layer called the *associative memory*. The associative memory layer connects the features of one channel with all others, allowing it to reconstruct the values at the visible nodes of a channel, if they are missing. We say the system learns to *know* a concept based on its ability to reconstruct any modality from one or more of the other modalities. For example, the system comes to know the concept "cat" because of is ability to create the image of a cat, given the sound and feel of a cat.

Consider a four channel DLA that contains four modalities: image, audio, motor and classification as shown in Figure 1. The image channel learns handwritten images (28 x 28 pixels) of the digits 0–9. The image channel contains two hidden RBM layers with 700 and 300 neurons respectively. The audio channel learns representation of the spoken digits 0–9. Similar to the image channel, the audio channel contains two RBM layers with 700 and 400 neurons, respectively. The motor channel learns a vector of 7 coordinates (14 values) that could control a robot arm to draw the digits in 2-D space. The motor channel has one hidden RBM layer with 100 neurons. The classification channel has 10 visible neurons where each acts as the class indicator for one of the digits. No hidden layers are needed. It is important to note that the classification channel is not required for the development of an accurate multimodal DLA using just the remaining channels. Rather, we have included the classification channel to see more directly the state of the representation in the associative layer of the system. The experimentation section will discuss this further.

Each generative channel is trained individually using its corresponding unsupervised examples. Once the individual channels are trained, the features generated by the top level RBM of each channel are used to train the associative memory with the goal of being able to reconstruct all channels given input on any one channel. This is not trivial. If improperly trained, the associative memory can become dependent on dominant features that are present in some channel A. This results in a system that does a poor job of reconstructing missing channels without the features from channel A.

To prevent the associative memory from becoming dependent on dominate features, the weights connecting the individual channels and the associative memory need to be fine-tuned. Typically, in a multimodal DBN of two channels, the back-propagation algorithm is used to fine-tune the weights of an equivalent deep autoencoder (Ngiam et al. 2011). The error is backward propagated through the weights from the output of one channel to the input of another (or both channels) and the weights adjust along the way. However, for three or more channels, this would have to be done for every possible input to output channel combination of the system. The possible number of configurations or examples grows exponentially ($2^n - 2$), with respect to the number of channels n. This is not a practical solution for even $n = 4$. It would mean building 14 separate models each with the same number of weights and fine-tuning each using a separate set of training examples. This would lengthen training times.

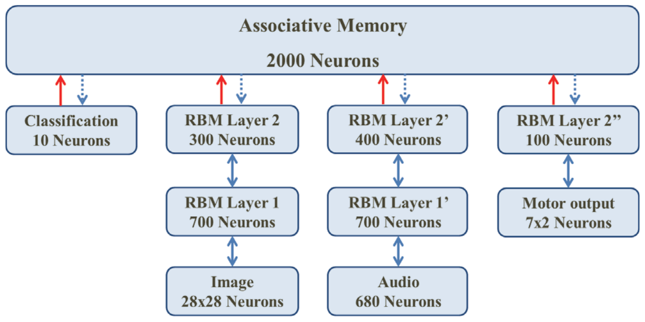

Figure 1: A four-channel multimodal DLA.

Fine-tuning using Back-Fitting

Our objective is to develop a multimodal Deep Learning Architecture (DLA) that can handle missing channels and linearly in the number of modalities. The solution lies in a contrastive version of the wake-sleep algorithm which fine-tunes RBM weights in a fast and greedy fashion (Hinton, Osindero, and Teh 2006; Hinton et al. 1995). This *back-fitting* algorithm can be applied to weights between any two RBM layers. Figure 2 shows the back-fitting algorithm applied to the weights between RBM layer v and RBM layer h. The input is presented at the v layer to produce activations at the h layer, using the RBM weights. These activations are saved and referred to as $\boldsymbol{h}$. The bidirectional weights from the RBM training are split into two unidirectional weights. The weights going up are called *recognition weights*, w_r, and the ones going down are called *generative weights*, w_g. The recognition weights w_r are highlighted by a solid red arrow and generative weights w_g are highlighted by a dotted blue arrow in Figure 2. The back-fitting approach has two steps: bottom-up (bu) and top-down (td). The bottom-up step uses the recognition weights while the top-down step uses the generative weights. The input features of the visible nodes presented at RBM layer v in the bottom-up step are used to stochastically activate the hidden nodes in layer h. These node activations in the top-down step are used to reconstruct the original input features using generative weights. The bottom-up and the top-down steps are alternated until the error between the reconstructed nodes and original input features is minimized. Each recognition

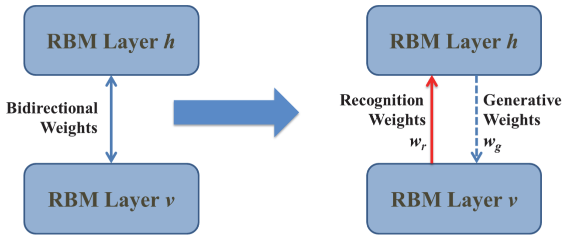

Figure 2: Applying back-fitting between two RBM layers

weight w_{ij} is updated using the equation,

$$\Delta w_{ij} = \epsilon(< v_i h_j >^{bu} - < v_i \boldsymbol{h_j} >^{td}) \qquad (1)$$

where ϵ is the learning rate, $< v_i h_j >^{bu}$ is the fraction of times visible node i and hidden node j are on together during the bottom-up step, and $< v_i \boldsymbol{h_j} >^{td}$ is the fraction of times the reconstruction of the visible node i and the saved activation of the hidden node j are on together during the top-down step. The generative weights w_g remain unaltered. The recognition weight update shown in Equation 1, is similar to the weight update equation in the contrastive divergence algorithm used by the RBM algorithm. However, there are two differences: (1) two set of weights, one for each direction, are used in back-fitting as opposed to one set of RBM weights for both directions in the contrastive divergence algorithm, and (2) $< v_i \boldsymbol{h_j} >^{td}$ is calculated during top-down pass in back-fitting, while $< v_i h_j >^{td}$ is calculated during the reconstruction pass in contrastive divergence. The alternating steps of bottom-up and top-down are continued until the reconstruction error,

$$e = \sum_{l=1}^{m} \sum_{i=1}^{x} (v_{li}^{bu} - v_{li}^{td}) \qquad (2)$$

is minimized to a desired level; where m is the number of examples and x is the number of input nodes in v. In this way, the back-fitting algorithm remains an unsupervised approach. To apply back-fitting to a multimodal DLA of n channels requires a few additions to the above procedure. These are described in the following section.

Iterative Back-Fitting for a Multi-Channel DLA

At the end of RBM training, the associative memory has learned a representation that can reconstruct all channels given input features from all channels. The representation does poorly at generating the features needed if input is provided on only one channel. To fine-tune the weights of the DLA we use an iterative version of the back-fitting algorithm described above. Algorithm 1 develops a representation in the associative memory layer that can generate the features needed for all channels when input is present on as few as one channel. The algorithm requires that every training example has a value for each modality. Once back-fitting has completed, testing can be done with values missing for one or more channels.

To describe the algorithm, consider a two-channel DLA (n=2) where each channel has two RBM layers, as shown in

Algorithm 1 Back-fitting for a multi-channel DLA

```
Input: $n \leftarrow$ number of channels, $k \leftarrow [1, n]$
    $m \leftarrow$ number of examples, $l \leftarrow [1, m]$
    Input features $v_l^k$ for the top RBM layer of the each channel $k$ and each example $l$
    Bi-directional RBM weights connecting associative memory and each channel
    $\phi$, the required minimum value for the reconstruction error, $e$, over all channels
Output: Recognition weights $w_r$ and generative weights $w_g$ for each channel $k$
    $e$, the reconstruction error over all channels
1:  for $l \leftarrow [1, m]$ do
2:    for $k \leftarrow [1, n]$ do
3:      Present input $v_l^k$
4:    end for
5:    Produce and save the associative memory node activations $\boldsymbol{h}$ for each example $l$ using RBM weights
6:  end for
7:  for $k \leftarrow [1, n]$ do
8:    Split bi-directional RBM weights into two unidirectional weights
9:    $w_r^k \leftarrow$ weights going to associative memory
10:   $w_g^k \leftarrow$ weights coming from associative memory
11: end for
12: while $e \geq \phi$ do
13:   $e \leftarrow 0$
14:   for $k \leftarrow [1, n]$ do
15:     for $l \leftarrow [1, m]$ do
16:       Present input $v_l^k$
17:       $v_l^y \leftarrow 0$, where $y \neq k$
18:       Produce activations $h$ using $w_r^k$
19:       Update $w_r^k$ using Equation (1) which uses the saved values of $\boldsymbol{h}$
20:     end for
21:     Compute $e^k$ for $v^k$ using Equation (2)
22:     $e \leftarrow e + e^k$
23:   end for
24: end while
```

Figure 3. The algorithm begins after RBM training has completed, including RBM training of the associative memory layer. For each example, the values of layer, v^1 and v^2, are presented at the top of RBM channels 1 and 2 to produce activations, $\mathbf{h}$, at the associative memory layer (lines 1-6). These $\mathbf{h}$ values are saved for each example (line 5). Then, the RBM weights connecting the top RBM layers and the associative memory are split into recognition weights, w_r^k, and generative weights, w_g^k (lines 7-11).

Fine-tuning of the weights begins on line 12 and repeats until the reconstruction error e for all channels is reduced to a value less than ϕ. For each example l, the input, v^1, is presented to the associative network from RBM channel 1 while the input to the associative layer from RBM channel 2, is set to 0 (lines 16-17). The combined inputs are used to produce activation, h, at the associative layer, using the recognition weights of channel 1, as shown in Figure 3. The recognition weights of channel 1, w_r^1, are then updated using the Equation 1 which serves to reduce the reconstruction error for channel 1 by taking into consideration the differ-

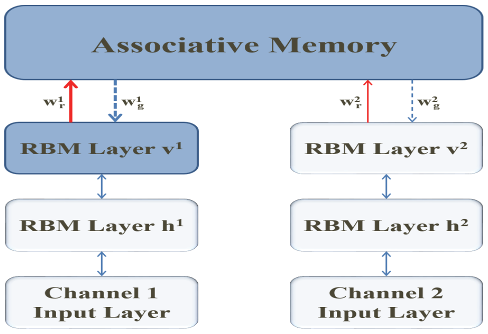

Figure 3: Applying back-fitting to a two-channel DLA

ence between each saved $\mathbf{h}$ and h. This is repeated for every training example. The error for channel 1 is computed and added to e (lines 21-22). Coming back to the top of the for while loop (line 12), an input example, v^2, is presented to the associative network from RBM channel 2 and the input (v^1) at RBM channel 1 is set to 0. The combined inputs are used to produce activation, h, at the associative layer, using the recognition weights of channel 2. The recognition weights of channel 2, w_r^2, are then updated. This is repeated for every training example. The error for channel 2 is computed and added to e.

The variation of the back-fitting algorithm described in Algorithm 1 grows linearly with the number of channels. After back-fitting has been successfully applied to a multi-channel DLA, an input from one channel can produce features at the associative layer that will reconstruct the remaining channels with minimal error.

This technique provides two benefits to our multimodal learning system: (1) The error associated with generative weights of one channel is factored into the recognition weights of other channels. Therefore, by updating the recognition weights of one channel the reconstruction of other channels are improved. (2) As the algorithm is linear in the number of channels, the system will scale up nicely in terms of time, unlike the supervised back-propagation approach that scales exponentially. The back-fitting process yields n sets of recognition weights and n sets of generative weights. These weights can be used to produce all the $2^n - 2$ possible configurations of channel inputs to outputs. This is a substantialsavings in terms of space and structural complexity as compared to the back-propagation solution. These benefits make the unsupervised back-fitting algorithm a more practical solution for fine-tuning a multimodal DLA over a supervised approach.

Experimentation

Objective

The objective of the following experiment is to develop a multimodal DLA and to test its ability to reconstruct all channels given any one channel.

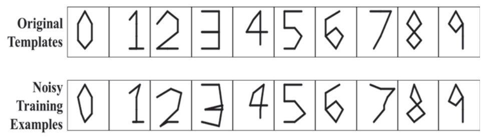

Figure 4: Template digits for the motor channel and an example of noisy training images created from these templates.

Method

We use a dataset of images and audio recordings collected from 20 male students. Each student was asked to write and speak the digits 0 through 9 ten times. This yielded 100 images and 100 audio signals per person and 2000 images and 2000 audio recordings overall. Specifically, the dataset contains 200 images and 200 audio recordings of each of the digits 0 through 9. Each image is re-sized to 28-by-28 pixels and the each audio recording is converted into a Short-Time Fourier Transform (STFT) signal (Allen and Rabiner 1977). These images and STFT signals are used as input for image and audio channels respectively. The classification channel is straightforward. It indicates the probability of the class of the input image or audio signal being 0, 1, 2, 3, 4, 5, 6, 7, 8 or 9. As shown in the top portion of Figure 4, for the motor channel we designed a template for each digit encoded as a sequence of seven Cartesian coordinates. A robotic arm would follow this sequence vector of motor coordinates to draw a digit in a 2-D space. To create a training example, a small amount of random noise is added to each (x,y) coordinate of a digit's template to form a new vector. A noisy training example of each digit is shown in Figure 4.

The RBM layers of the individual channels are of the sizes shown in Figure 1. Each layer is trained using a learning rate of 0.1 for all channels except the audio channel. The STFT signal of the audio channel is represented using real values which requires a smaller learning rate of 0.01 (Hinton 2012). Once the individual channels are trained, the associative memory layer is trained using the features from the top RBM layers of each channel. A 10-fold cross validation approach is taken to build and test the system. Each time we train the system with data from 18 users (1800 examples) and test it with data from the remaining 2 users (200 examples). A portion of the training data (2 users or 200 examples) is used as a validation set to prevent the system from over-fitting. The typical training time for a single cross-validation run was 37.5 hours using two i7 CPU cores.

When the system is tested on an example, input values are presented at the visible layer of each channel and used to activate the hidden RBM layers. Ultimately, the input from each channel produces activation at the top associative layer, which in turn can be used to reconstruct the associated visible neurons of the other three channels. This procedure is repeated for all four input channels. The performance results are calculated for each reconstructed channel based on the desired output at its visible neurons. The classification channel is evaluated based on how accurately it matches the target class for each example. The reconstructed images are evaluated using an image classifier. The image classifier is a DBN developed by Hinton et al. (Hinton and Salakhutdinov 2006) which is retrained on our image dataset to a test set accuracy of 99%. The audio in our system is represented STFT signal, which is an irreversible transformation for actually producing sound. However, as with the image channel, the reconstructed STFT signal can be tested using an independent classifier developed using a random forest algorithm trained to a test set accuracy of 93%. The reconstruction of the motor channel vector is compared with the template of the target digit for each example. The average distance between each coordinate in the reconstructed vector and its corresponding coordinate in the target template is used as the error metric. Based on an experiment with human subjects we determined that a reconstruction error less than 2.2 on the motor channel means that the digit has been correctly drawn for proper classification.

Results

The results are averaged over the 10-fold cross-validation runs. The results are organized by reconstructed channel to more easily compare the performance of the multimodal DLA as a function of the input channel used.

Reconstruction of the Classification Channel. Table 1 shows the percent error in reconstructing each classification channel by presenting input at the other three channels. Input on the image channel reconstructed the classification channel with an average error of 4.1% and the audio channel reconstructed the with an average error of 14%. We observed a positive correlation between length of the utterance and the error. The digits such as "6" and "8" had a shorter utterance period compared to "4" and "5", hence the lower reconstruction error. We believe this effect could be mitigated by choosing a different length of STFT representation or using a more complex encoding technique such as MFCC (Moselhy and Abdelnaiem 2013). The motor channel reconstructed the classification channel with no error. The motor input signal is quite simple when compared to the audio and image inputs; hence, it is able to send a clear and concise reconstruction signal to the system.

Input	0	1	2	3	4	5	6	7	8	9	Overall
Image	3	6	4	8	4	2	4	0	8	2	4.1
Audio	12	26	10	19	16	15	1	14	3	24	14
Motor	0	0	0	0	0	0	0	0	0	0	0

Table 1: Percent error in reconstructing the classification channel from each of the other channels.

Reconstruction of the Image Channel. Table 2 shows the percent error in classification of the reconstructed images by presenting class, audio and motor as input. The class channel and motor channel reconstructed images with no error, whereas the audio channel produced the image channel with an average error of 19%. This is due to the difference in the complexity of the signals from the various input channels. The audio channel signal is much more complex compared to the class and motor channels. Figure 5 shows examples of reconstructed images giving input on various channels.

Input	0	1	2	3	4	5	6	7	8	9	Overall
Class	0	0	0	0	0	0	0	0	0	0	0
Audio	26	16	14	10	33	24	2	21	16	24	19
Motor	0	0	0	0	0	0	0	0	0	0	0

Table 2: Percent error in reconstructing the image channel from each of the other channels.

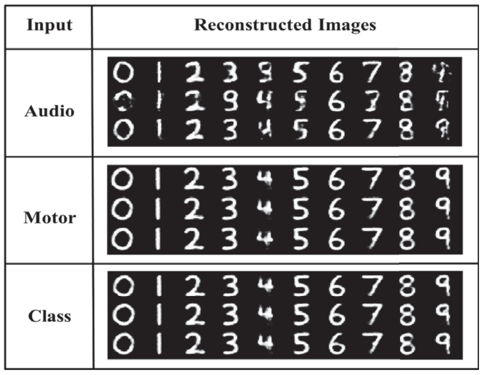

Figure 5: Examples of reconstructed images given each of the other channels.

It is important to note that an additional experiment involving a system with only an image and audio channel has shown that the image channel can be reconstructed with an error of 17% when given the audio channel. Therefore, the success of reconstructing a matching image given an audio signal remains high and independent of the classification and motor channels being present. For further details please see (Iqbal 2015).

Reconstruction of the Audio Channel. Table 3 shows the percent error in reconstructing the audio channel STFT signal given each of the other channels. Again, the classification channel reconstructed the audio channel with no error. The motor channel produced 11% error while the image channel produced 19% error. We remain uncertain why input on the motor channel for the digit 5 did so poorly.

Input	0	1	2	3	4	5	6	7	8	9	Overall
Class	0	0	0	0	0	0	0	0	0	0	0
Image	20	20	5	25	40	10	5	20	5	35	19
Motor	0	0	0	0	20	90	0	0	0	0	11

Table 3: Percent error in reconstructing the Audio channel from each of the other channels.

Reconstruction of the Motor Channel. Table 4 and Figure 6 show the percent error in reconstructing the motor channel given the other three channels. Examples of the reconstructed motor vectors are shown in Figure 7. The audio channel input resulted in the highest average reconstruction error of 1.92, well under the the acceptable error of 2.2. The class channel and the image channel had lower average errors of 1.43 and 1.63, respectively.

Input	0	1	2	3	4	5	6	7	8	9	Overall
Class	0.91	2.05	1.33	1.90	1.40	1.56	1.2	1.64	1.00	1.32	1.43
Image	1.01	2.16	1.55	2.06	1.87	1.69	1.29	1.69	1.64	1.33	1.63
Audio	1.93	2.52	1.84	2.26	2.48	1.84	1.28	2.07	1.39	1.58	1.92

Table 4: Error in reconstructing the motor channel from each of the other channels.

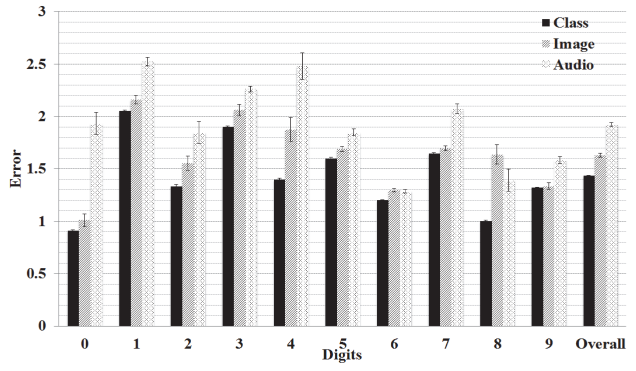

Figure 6: Error in reconstructing the motor channel from each of the other channels.

Conclusions

The results from the experimentation have shown that our unsupervised deep multimodal system is able to overcome the two challenges posed in the introduction of this paper. The iterative back-fitting algorithm that we employ is able to associate channels of varying signal complexity while mitigating the dependence of associative memory on the dominant channels. The approach also scales linearly in the number of DBN channels, because the only added computation for a new channel is updating the recognition weights between that channel and the associative memory. This means that the back-fitting iterations will grow linearly with each new channel. For further details please see (Iqbal 2015).

The results of the experimentation confirm that our unsupervised deep multimodal system is able to accurately reconstruct one channel given input on any other channel. The reconstruction accuracy does vary based upon the complexity of the input signal for a channel. Channels with less complex signals, such as classification and motor channels, produced more accurate class features in the shared associative memory and these features can then more accurately reconstruct the outputs for all other channels. Channels with more complex signals such as image and audio often produce less accurate features in the shared associative memory and these lead to greater reconstruction errors at the visible neurons of the other channels.

In this research, our training examples contain values for all modalities. In reality, such examples are scarce. For instance, consider the process of learning the modal inputs of the concept "cat". At first we might see a cat and hear it meow but not get to touch or smell the cat. We learn the visual features and sound features of the cat and associate these two modalities. At a later point in time, we get to touch and see a cat but the cat makes no sound. Now we can associate the touch of a cat with its most recent image and potentially the prior visual and audio features. In future work we plan to enhance the multimodal DLA system to learn using examples that have modalities that are absent.

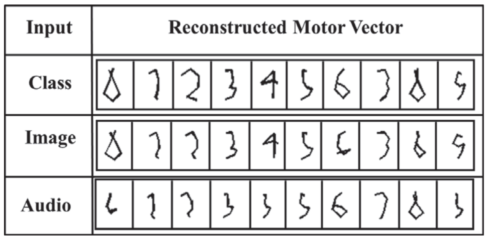

Figure 7: Examples of reconstructed motor vectors given each of the other channels.

References

Allen, J. B., and Rabiner, L. R. 1977. A unified approach to short-time fourier analysis and synthesis. *Proceedings of the IEEE* 65(11):1558–1564.

Bengio, Y. 2009. Learning deep architectures for AI. *Foundations and Trends in Machine Learning* 2(1):1–127. Also published as a book. Now Publishers, 2009.

Hinton, G. E., and Salakhutdinov, R. R. 2006. Reducing the dimensionality of data with neural networks. *Science* 313(5786):504–507.

Hinton, G. E.; Dayan, P.; Frey, B. J.; and Neal, R. M. 1995. The "wake-sleep" algorithm for unsupervised neural networks. *Science* 268(5214):1158–1161.

Hinton, G. E.; Osindero, S.; and Teh, Y.-W. 2006. A fast learning algorithm for deep belief nets. *Neural computation* 18(7):1527–1554.

Hinton, G. E. 2007. Learning multiple layers of representation. *Trends in cognitive sciences* 11(10):428–434.

Hinton, G. E. 2012. A practical guide to training restricted Boltzmann machines. In *Neural Networks: Tricks of the Trade*. Springer. 599–619.

Iqbal, M. S. 2015. Mulit-modal learning using an unsupervised deep learning architecture. Master's thesis, Acadia University, Canada.

Karpathy, A., and Fei-Fei, L. 2015. Deep visual-semantic alignments for generating image descriptions. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kiros, R.; Salakhutdinov, R.; and Zemel, R. 2014. Multimodal neural language models. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 595–603.

Moselhy, A. M., and Abdelnaiem, A. A. 2013. Lpc and mfcc performance evaluation with artificial neural network for spoken language identification. *signal processing* 10:11.

Ngiam, J.; Khosla, A.; Kim, M.; Nam, J.; Lee, H.; and Ng, A. Y. 2011. Multimodal deep learning. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, 689–696.

Socher, R.; Karpathy, A.; Le, Q. V.; Manning, C. D.; and Ng, A. Y. 2014. Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics* 2:207–218.

Srivastava, N., and Salakhutdinov, R. R. 2012. Multimodal learning with deep Boltzmann machines. In *Advances in neural information processing systems*, 2222–2230.