2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Dual Deep Neural Network Approach to Matching Data in Different Modes

Mark Eastwood and Chrisina Jayne

***Abstract*—This paper investigates the application of a novel Deep Neural Network (DNN) architecture to the problem of matching data in different modes. Initially one DNN is pre-trained as a feature extractor using several stacked Restricted Boltzmann Machine (RBM) blocks on the entire training data using unsupervised learning. This DNN is duplicated and each net is fine-tuned by training on the data represented in a specific mode using supervised learning. The target of each DNN is linked to the output from the other DNN thus ensuring matching features are learnt which are adjusted to take differing representation into account. These features are used with some distance metric to determine matches. The expected benefit of this approach is utilizing the capability of DNN to learn higher level features which can better capture the information contained in the input data's structure, while ensuring the differences in data representation are accounted for. The architecture is applied to the problem of matching faces and sketches and the results compared to traditional approaches employing Principal Component Analysis (PCA) or Linear Discriminant Analysis (LDA).**

I. Introduction

There has been a significant research interest in recent years related to Deep Architectures, which refer to architectures with a larger number of hidden layers [1]. In 2006 Hinton et al. [1] introduced an unsupervised fast, greedy learning algorithm that can find a fairly good set of parameters quickly in deep networks with millions of parameters and many hidden layers. This involved the unsupervised pre-training of deep supervised multi-layer neural networks using the Restricted Boltzmann Machine (RBM) generative model for each layer [1], [2]. The pre-training serves as an initialization of the neural network which is then fine-tuned with respect to a supervised criteria. The unsupervised pre-training helps with initialization of the net into a more favorable region of the weight space [3] compared to a random initialization. Several studies have show that this leads to better generalization results [3], [4], [5].

The unsupervised learning in the Deep Architectures helps also by extracting salient information about the input distribution in the hidden layers, [2]. The features extracted in the first hidden layer are seen as low-level features. These are then fed as the input in the second layer where the new extracted features are higher-level features [2]. This approach leads to steadily higher-level abstractions of the training data, and the potential to learn complex relationships with much fewer neurons than a network with few hidden layers would be able to [6]. Deep architectures have seen use in many application areas, for example object recognition [7], speech recognition [8], and musical genre categorization [9].

M. Eastwood and C. Jayne are with Coventry University, Priory Street, Coventry CV1 5FB, UK email:ab1527@coventry.ac.uk

This paper investigates the application of a novel Deep Neural Network (DNN) architecture to the problem of matching data in two different modes. There are a number of situations in which we wish to identify matching objects which have been represented in different ways. Common examples are faces (or images in general) viewed in different frequency bands or lighting conditions, or represented as photos or sketches. While the representations differ, there are similarities in the important features that may be exploited. In this paper we will focus on matching faces represented as photos and sketches, an important problem in the context of forensics in particular.

The proposed method first uses a stack of unsupervised RBM blocks to extract features using the entire training data. The parameters found during this step are used as initialization parameters for two initially identical DNNs which are then fine-tuned using supervised learning. The first DNN is trained using as input the training data of one mode, while the second DNN is trained using matching training data of the other mode. During training, the networks are linked so that, at each epoch, the target of the first DNN is updated to be the output of the second DNN and vice-versa. The motivation for this method is that the DNN architecture enables the extracting of relevant features from the input data, with each net fine-tuned on one data representation to tailor the global features to their representation in each data mode. Subsequently the fine-tuned DNN pair can be used to map data into a common feature space in which matching can be performed. We call the proposed approach Dual Matching Deep Neural Network (DMDNN).

The DMDNN approach has conceptual similarities with the Coupled Spectral Regression (CSR) framework [10] applied to solving the problem of matching heterogeneous face images. The CSR initially models the properties of data represented in different modes separately, and then learns two associated projections to project heterogeneous data into a discriminative common space. Classification is then performed in the common space. The motivation for the DMDNN is similar to the one found in [10] for the CSR learning algorithm. Data from the same object in different modes are at different positions in observation space. If the same projection or feature extraction is used for the two types of data the comparison of the obtained projections would not be optimal. Therefore having separate projections tailored to each representation but mapping into a common subspace is more likely to produce better classification/matching results. Linear and Kernel based CSR methods are introduced in [10]. Projections in CSR are found by optimizing an objective function which includes separate terms for each type of data.

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

In the DMDNN separate DNN architectures are trained for each type of data to map matching data to the same point in a unified/common feature space.

The proposed method is applied to the problem of matching digital face and sketch images in which faces are represented in both sketch and photo modality. This problem has been recently investigated by a number of researchers e.g. [11], [12], [13], [14] and it is an important real-life law enforcement application. Sketch recognition algorithms can be classified into generative and discriminative [12] categories. Generative methods transform first a digital photo into a synthesized sketch before matching with a probe sketch [11], while discriminative methods perform feature extraction directly on the given digital photo and sketch images [14], [13] without generating synthesised images, and perform matching in this feature space. The DMDNN method is a discriminative approach, using a novel DNN architecture to extract features in a common feature space and performing matching using these features.

The experiments are conducted using the CUHK student database [15]. The results are compared to traditional feature extraction approaches employing Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA). The initial experiments and results demonstrate the potential of utilizing DNN architecture for matching face and sketch images.

The main contributions of this study can be summarized as follows:

- developing a novel method for matching data in two different modes based on a linked Deep Neural Network architecture;
- utilizing the Enhanced Gradient [16], a recent advancements for improving the training of the RBMs, in this novel context.
- evaluating the efficiency and effectiveness of the proposed method in a real-life application for matching digital face and sketch images;

The remainder of the paper is structured as follows. Section II introduces the Restricted Boltzmann Machine and Deep Networks. The proposed DMDNN approach is presented in Section III. Section IV presents the experiments for evaluating the effectiveness of the DMDNN and the obtained results. Finally, Section V gives our conclusions and directions of future work.

II. The Restricted Boltzmann Machine and Deep Networks

A. Restricted Boltzmann Machine

A boltzmann machine is a system of random variables $\mathbf{v}, \mathbf{h}$ whose joint probability can be described by an energy function as follows:

$$P(\mathbf{v}, \mathbf{h}) = \frac{\exp(-E(\mathbf{v}, \mathbf{h}))}{Z} \tag{1}$$

where Z is a normalisation factor

$$Z = \sum_{\mathbf{h}} \sum_{\mathbf{v}} \exp(-E(\mathbf{v}, \mathbf{h}))$$

.

The $\mathbf{v}$'s represent random variables we can observe, whereas the $\mathbf{h}$'s represent hidden variables that we cannot directly observe, only infer. The probability of a given visible vector $\mathbf{v}$ can be calculated by marginalization over $\mathbf{h}$:

$$p(\mathbf{v}) = \frac{1}{Z} \sum_{\mathbf{h}} \exp(-E(\mathbf{v}, \mathbf{h})) \tag{2}$$

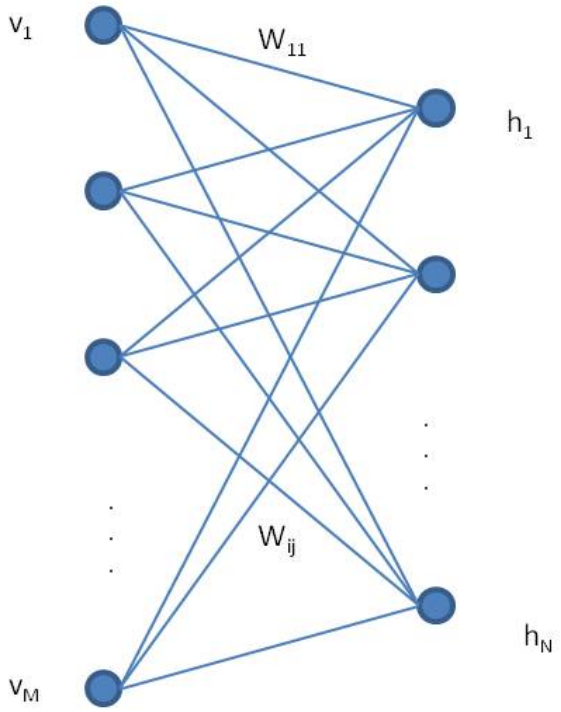

Fig. 1. A graphical illustration of the terms in the RBM energy function

Formally, the Boltzmann Machine belongs to a class of graphical models known as Markov random fields [17]. There are many different types of Boltzmann Machine, distinguished by the particular form of the energy function used; here we will focus on a subclass of the Boltzmann machine called the restricted Boltzmann machine (RBM). This is characterized by an energy function which contains no cross terms, i.e. no terms of the form $f(v_i, v_j)$ or $f(h_i, h_j)$ for $i \neq j$. This can be represented graphically as in figure 1, where a connection represents a term in the energy function involving the variables connected. With this form, the probability distribution for each v_i is independent given $\mathbf{h}$, and similarly for the h_j's given $\mathbf{v}$. This simplifies training significantly, allowing parallel sampling of nodes in one layer given the other layer. The most commonly used form of the energy function is:

$$E(\mathbf{v}, \mathbf{h}) = -\sum_i v_i b_i - \sum_j h_j c_j - \sum_{ij} v_i W_{ij} h_j \tag{3}$$

where both $\mathbf{v}$ and $\mathbf{h}$ are binary random variables. The W_{ij} are the weights as illustrated in figure 1, and the b_i's and c_j's are biases. The sums over i and j run over the M visible and N hidden nodes, respectively. The activation probability of a hidden node under this energy is:

$$P(h_j = 1|\mathbf{v}) = \frac{\exp\left(-c_j - \sum_i v_i W_{ij}\right)}{1+\exp\left(-c_j - \sum_i v_i W_{ij}\right)} \quad (4)$$

$$= sigmoid\left(c_j + \textstyle\sum_i v_i W_{ij}\right) \quad (5)$$

This matches the activation rule for a sigmoid neuron in an MLP, making it suitable for layer wise pre-training of a deep NN, as described in the next section. A number of other energy functions have been proposed for RBMs, to fulfil various roles. The Gaussian-Bernoulli RBM [18] was proposed to deal more naturally with continuous $\mathbf{v}$, and convolutional RBMs [19] have been proposed for image-related applications [20] and for use as pre-training of deep convolutional neural networks. RBMs allowing for hidden nodes to take continuous values were proposed in [21], and energy functions which allow a more direct modelling of the covariance between different visible units have been suggested in [22].

An RBM is typically trained by an approximation to maximum likelihood. The gradient of the log-likelihood L can be expressed [23] as:

$$\frac{\partial L}{\partial \theta} = -\left\langle \frac{\partial E}{\partial \theta} \right\rangle_d + \left\langle \frac{\partial E}{\partial \theta} \right\rangle_m$$

Where θ represents some parameter of the energy function, and $< \cdot >_d$, $< \cdot >_m$ denote averages over the data/model distributions respectively. While the first average is easily calculated, the second in general cannot be computed efficiently. It can be approximated via Gibbs sampling, i.e. alternating samples of $P(\mathbf{h}|\mathbf{v})$ and $P(\mathbf{v}|\mathbf{h})$ which will converge to the required distribution after many iterations, however this is still very costly. In [24] it was proposed to approximate the model distribution by running the Gibbs sampling for only a small number n of iterations (in practice one is often found to be enough), initialized from data samples. This procedure is called contrastive divergence. While it was initially proposed as an approximation to the Gibbs-sampled likelihood gradient, it was also shown to follow more closely the gradient of the difference of two Kullback-Leibler (KL) divergences:

$$CD_n = KL(p_0|p_{\inf}) - KL(p_n|p_{\inf})$$

Where p_n denotes the probability distribution of samples after n steps of Gibbs sampling starting from the training distribution p_0. Thus $p_{\inf}$ is the model distribution.

For the energy function given in equation 3, the CD updates have the form:

$$\begin{aligned} \Delta W_{ij} &= \lambda(\langle v_i h_j \rangle_d - \langle v_i h_j \rangle_m) \\ \Delta b_i &= \lambda(\langle v_i \rangle_d - \langle v_i \rangle_m) \\ \Delta c_j &= \lambda(\langle h_j \rangle_d - \langle h_j \rangle_m) \end{aligned} \quad (6)$$

where λ is a learning rate.

B. The Enhanced Gradient

The CD updates in equations 6 have a number of problems, which were identified in [16]. In this paper, an alternative gradient was also proposed to address these problems, which we will describe here.

There are two main problems with the update gradients 6. The first is that as the covariance of v and h w.r.t. distribution P is

$$cov_P(v_i, h_j) = < v_i h_j >_P - < v_i >_P < h_j >_P$$

we can write the weight update in 6 as

$$\nabla w_{ij} = cov_d(v_i, h_j) - cov_m(v_i, h_j)$$
$$+ < v_i >_{dm} \nabla c_j + < h_j >_{dm} \nabla b_i$$

where $< \cdot >_{dm} = \frac{1}{2} < \cdot >_d + \frac{1}{2} < \cdot >_m$. This means the gradient w.r.t the weights may be correlated with the gradient w.r.t the biases, leading to a problem where many neurons learn features similar to the bias terms.

Secondly, imagine a transformation of the data representation such that some binary units of the RBM are flipped:

$$\bar{v_i} = v_i^{1-f_i}(1 - v_i)^{f_i}$$
$$\bar{h_j} = h_j^{1-g_j}(1 - h_j)^{g_j}$$

with corresponding transformations of the RBM parameters so that the resulting RBM has an equivalent energy function. The f_i's and g_j's define the transformation, with a 1 or 0 indicating if the corresponding v_i or h_j is flipped (or not). If the model is transformed, updated according to 6, and transformed back, the resulting update depends on the transformation, with each possible transformation yielding a valid ML update. There is no good reason to use the original update over any other possible update.

The idea introduced in [16] is to take a weighted sum of these possible updates, weighting more highly the sparse data representations for which $< \cdot >_{dm} \approx 0$. In these representations, the problem of correlation of weight updates with biases described above is minimized. With this choice of weights, the summation results in the following updates:

$$\nabla_e w_{ij} = cov_d(v_i, h_j) - cov_m(v_i, h_j)$$

$$\nabla_e b_i = \nabla b_i - \sum_j < h_j >_{dm} (\nabla w_{ij} - \nabla b_i - < v_i >_{dm} \nabla c_j)$$

$$\nabla_e c_j = \nabla c_j - \sum_i < v_i >_{dm} (\nabla w_{ij} - \nabla c_j - < h_j >_{dm} \nabla b_i)$$

.

A slightly simplified version of the above bias updates,

$$\nabla_e b_i = < v_i >_d - < v_i >_m - \sum_i < h_j >_{dm} \nabla_e w_{ij}$$

$$\nabla_e c_j = < h_j >_d - < h_j >_m - \sum_i < v_i >_{dm} \nabla_e w_{ij}$$

are recommended in [16] and shown experimentally to perform well, and so are used in this paper also.

C. Deep Neural Networks

A 'deep' neural network gains its name simply by incorporating more than the 1-2 hidden layers commonly seen in NN applications. Typical sizes for a deep neural net are 4-5 layers, in some cases larger. Such networks are desirable for their ability to learn steadily more complex, higher level features, and build high-level representations of objects upon these. However training of these networks purely using back-propagation, or other similar gradient based methods works poorly as the gradient becomes too diffuse when back-propagated over multiple layers [4]. This has been addressed by a greedy layer-wise pre-training procedure, most often treating each layer as an RBM, though similar approaches which treat a layer as an auto-encoder also exist [2]. In order to use this approach a layer model must be defined whose hidden node activation probability matches the activation of a corresponding node in the deep neural network architecture. In the case of MLP and convolutional NNs, an RBM or convolutional RBM respectively satisfies this criterion, and so can be trained to provide a layer wise initialization of a deep NN. As an example, equation 5 matches the activation rule of a sigmoid neuron in a standard MLP.

This layer wise pre-training proceeds as follows. The first layer RBM is trained using the data vectors $\mathbf{v}$, resulting in parameters θ_1. In a subsequent layer k, the activation probabilities Q (calculated as in equation 5) of the hidden units in the $k-1$'th layer are calculated by propagating the data vectors $\mathbf{v}$ through the layers already learnt, and are used as the training vectors for the k'th layer resulting in parameters θ_k. The learnt parameters for each layer may then be used to initialize the deep network, which can then be fine-tuned as a complete net using standard gradient-based techniques.

There are a number of reasons for the effectiveness of pre-training. Intuitively, it ensures that the structure of the input distribution to a layer is reflected in the initial weights [5], meaning less information should be lost in the transformation learnt by the layer in the fine-tuning stage. There is also work [25] that suggests the pre-training has a regularizing effect on the network.

Deep versions of a number of different NN architectures can be found in the literature, for example MLP [5] and convolutional [19] architectures, and have been found to perform exceptionally well in some applications.

III. Dual Matching Deep Neural Network Architecture

In this section we will describe our proposed architecture. Conceptually, the proposed method exploits the fact that an image displayed in two different modalities will have similarities in the important features that make up the image. An RBM trained on a mixture of images from two different modes can be expected to output features that are similar when given matching images in two different modes as input. We take RBMs trained in this way to initialize two (initially identical) DNNs as described in section II-C, and fine-tune each net on images from a single mode only. The objective is to train this network pair to map matching images in different modalities to the same point in a unified feature space (and non-matching images to far-apart points in the feature space). We expect images of a certain type to be able to be well represented in a unified feature space if we adjust the mappings to account for the differences in how each feature appears in the different modes in question. We wish ultimately to identify a unified feature space in which we can match an image with a transformed version of itself, by training feature extractors specific to the different modes to give the same feature map. To do this, in each epoch we have two stages of training. In the first stage we link the networks by designating the output of each net to be the target of the other, as in step 2 below. This step aims to train the pair of networks to map matching images in the two modes to the same point in the feature space. In the second stage, mis-matching images are used and the target for each net is generated by adding a small fraction of the difference between the outputs of the two networks to their current outputs, as in step 6 below. This step encourages the network pair to map mis-matching images to far-apart points in the feature space.

More formally, let X_p, X_s be the sets of images in the two modes. Layers of RBMs are trained as in section II-C, using $X = \{X_p, X_s\}$ to train the first layer. These RBMs are used to initialize two initially identical DNNs, N_p and N_s. Training proceeds in the following steps, for each epoch:

1) Outputs of nets calculated as $Y_p = N_p(X_p)$ and $Y_s = N_s(X_s)$
2) Targets of each net set as $T_p = Y_s$ and $T_s = Y_p$.
3) Calculate update for network weights according to chosen gradient calculation
4) Randomly permute the order of examples inX_s, X_p such that images in the same position in X_p and X_s no longer match. That is, $x_{p,i}$ and $x_{s,i}$ are images of different entities for all i. Denote these non-matching sets as $\bar{X_p}, \bar{X_s}$.
5) Output of nets calculated as $\bar{Y_p} = N_p(\bar{X_p})$ and $\bar{Y_s} = N_s(\bar{X_s})$.
6) Calculate $D = \bar{Y_s} - \bar{Y_p}$, and set targets
$$\bar{T_p} = \frac{\bar{Y_p} - D}{||D||}$$
$$\bar{T_S} = \frac{\bar{Y_s} + D}{||D||}$$
.
7) Calculate update according to chosen gradient calculation.

The process then returns to step 1 for the next epoch. The training is illustrated graphically in figure 2. In the matching stage, a probe sketch s is provided to network N_s to give features $y_s = N_s(s)$, and a set of features $\{y_{p,i}\} = \{N_p(p_i)\}$ for all i in the photo library are similarly generated. The 1st, 2nd etc best matching photo given probe sketch are

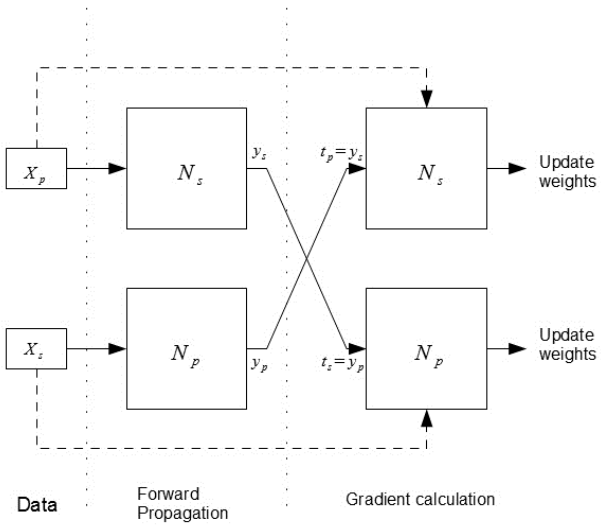

Fig. 2. A graphical illustration of the main phase of a training step in the proposed architecture

then determined according to minimum (euclidean) distance between f_s and the $f_{p,i}$.

At this stage it is worth discussing briefly the issue of stability. As we are modifying the target after each iteration, we must take steps to ensure no unstable behaviour occurs. We implemented two measures to ensure this. Firstly, the signal used to modify the target in step 6 is normalized by the magnitude of the difference between the outputs of the two nets, so that this signal does not grow larger if D grows larger, a situation which could result in instability due to positive feedback. Secondly, we use sigmoid output nodes, which precludes the possibility of the output of some nodes drifting to large positive or negative values during training.

This architecture is quite general, in that the specifics of the two networks used can be very different while still using the same top-level methodology of training each network on an individual mode, and updating the targets of each network every iteration.

In the next section we will apply the proposed architecture to a real world matching problem.

IV. Face/Sketch Matching

In this section, we apply the architecture introduced in the previous section to the problem of facial photo/sketch matching. This problem is predominant in the context of forensics, when a forensic sketch of a suspect must be matched to a library of photos of previously known criminals, for example.

We use the CUHK face-sketch database [15], in which we have 188 photo/sketch pairs from 188 individuals both male and female. Faces are originally of 250x200 pixel size, but are cropped in an oval based on eye co-ordinated for use in our method. The resulting images consist of 14019 pixels. An example cut-out photo/sketch pair is shown in Table I. We use 100 individuals for training, split 80/20 into training and validation sets. The remaining 88 are used as the testing set.

TABLE I
Illustration of cropping process

Original	Cropped
	

In addition to results for single runs of the DMDNN, we also test the performance of an ensemble of these nets. In order to create the diversity between individuals that an ensemble approach requires, in addition to the natural diversity arising from the random aspects of the deep net training/pre-training, we train each member of the ensemble on a different, randomly generated training/validation split.

Individuals provide an output in the form of a ranking for all the library photos in response to a probe sketch, according to how well each matches the probe sketch. The individual rank outputs are combined in the following way. The ranks for all individuals for library photo i are summed $s_i = \sum_k r_i^{(k)}$ to give a score s_i. The library matches are then ranked according to minimum score. As an example, a library photo ranked 5,2,1,4,4 (summed score 16) by 5 individuals in response to a probe sketch would be ranked above a library photo ranked 3,4,2,3,5 (summed score 17) when determining the ensemble rankings.

We will compare results from individual and ensemble pair nets with those of standard methods to be found in the literature, namely PCA and LDA.

For each experiment, 10 repetitions are run in which the 188 faces are randomly split into a 100 face training set and an 88 face testing set. We used deep networks with two hidden layers with 600 neurons each, and an output layer of 600 neurons. Networks were trained for 1000 epochs with early stopping if no improvement is made on the validation set within a 250 epoch interval. Learning rate used was 0.04. The ensemble size, where relevant, was 10. The results are shown in table II. Recognition rates are shown for best 1,3, and 5 matches, where recognition is deemed successful if the correct match to the probe image appears in the top 1,3 or 5 of the returned matches, respectively.

As can be seen, while the DMDNN method performed worse than the LDA method, it performed significantly better than the PCA method. This is promising, as in the work presented here we use a fairly simple Deep net as the basis network for our top level architecture, with which we may

Method	1 Best	3 Best	5 Best
DMDNN	57.5	79.7	87.3
DMDNN Ens	64.1	85.6	91.0
PCA	37.5	56.8	64.8
LDA	93.2	97.7	97.7

TABLE II
RECOGNITION RATES (%) IN TOP 1,3 AND 5 MATCHES ON FACE/SKETCH MATCHING TASK

also use more sophisticated architectures such as convolutional nets, which are more suited to image recognition problems, to achieve better performance. Other methods in the literature have been applied to this dataset, though a direct comparison is difficult as some have combined the dataset we used with one or more other datasets which were not available to us. In [11], sketches are synthesised patch wise for matching with a probe sketch, with Markov random fields used to maximise the probability that adjacent patches match, reporting an accuracy of 96% on a dataset that includes roughly 600 examples. In [13], genetic algorithms are used to optimize a matching procedure on features extracted at multiple resolutions, reporting an accuracy of 94% on a dataset of roughly 300 examples. With a more sophisticated base neural network, and a potentially extended dataset, we hope to be able to match or improve on these results.

The ensemble of DMDNNs, as expected, performed much more consistently than an individual net. The performance gap between a DMDNN ensemble and LDA closes significantly when the top few results are considered, which is very often the case in practical use where a shortlist of matches would be delivered for human inspection to determine if the match exists.

V. CONCLUSION

We have introduced a novel method in which neural network feature maps are fine-tuned on images in different modes to map these representations into a common feature space for matching, and illustrated its use in a face/sketch matching context.

The method is a promising approach that can be used as a general top level architecture incorporating many different neural network types as base components. Our implementation produced good results using a simple Deep feed-forward network as the base component, and although some other standard methods can achieve better performance, we expect that the use of more advanced component architectures which are better suited to image recognition, for example a deep convolutional network, will significantly improve recognition rate. This will be the focus of future work.

REFERENCES

[1] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006. [Online]. Available: http://dx.doi.org/10.1162/neco.2006.18.7.1527

[2] Y. Bengio, "Learning deep architectures for ai," *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 1–127, Jan. 2009. [Online]. Available: http://dx.doi.org/10.1561/2200000006

[3] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, U. D. Montral, and M. Qubec, "Greedy layer-wise training of deep networks," in *In NIPS*. MIT Press, 2007.

[4] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS10). Society for Artificial Intelligence and Statistics*, 2010.

[5] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin, "Exploring strategies for training deep neural networks," *J. Mach. Learn. Res.*, vol. 10, pp. 1–40, Jun. 2009. [Online]. Available: http://dl.acm.org/citation.cfm?id=1577069.1577070

[6] N. Le Roux and Y. Bengio, "Representational power of restricted boltzmann machines and deep belief networks," *Neural Comput.*, vol. 20, no. 6, pp. 1631–1649, Jun. 2008. [Online]. Available: http://dx.doi.org/10.1162/neco.2008.04-07-510

[7] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, P. Bartlett, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., 2012, pp. 1106–1114.

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

[9] X. Yang, Q. Chen, S. Zhou, and X. Wang, "Deep belief networks for automatic music genre classification." in *INTERSPEECH*. ISCA, 2011, pp. 2433–2436.

[10] Z. Lei and S. Z. Li, "Coupled spectral regression for matching heterogeneous faces," in *CVPR*, 2009, pp. 1123–1128.

[11] X. Wang and X. Tang, "Face photo-sketch synthesis and recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 11, pp. 1955–1967, 2009.

[12] H. Bhatt, S. Bharadwaj, R. Singh, and M. Vatsa, "Memetically optimized mcwld for matching sketches with digital face images," *Information Forensics and Security, IEEE Transactions on*, vol. 7, no. 5, pp. 1522–1535, 2012.

[13] H. S. Bhatt, S. Bharadwaj, R. Singh, and M. Vatsa, "On matching sketches with digital face images," in *Biometrics: Theory Applications and Systems*, 2010, pp. 1–7.

[14] P. Yuen and C. H. Man, "Human face image searching system using sketches," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 37, no. 4, pp. 493–504, 2007.

[15] X. Wang and X. Tang, "Face photo-sketch synthesis and recognition," 2009, dataset available from: http://mmlab.ie.cuhk.edu.hk/archive/facesketch.html.

[16] T. R. K. H. Cho and A. Ilin, "Enhanced gradient for training restricted boltzmann machines," *Neural Computation*, vol. 25:3, pp. 805–831, 2013.

[17] R. Kindermann and J. L. Snell, *Markov Random Fields and Their Applications*. AMS, 1980.

[18] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006.

[19] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09. New York, NY, USA: ACM, 2009, pp. 609–616. [Online]. Available: http://dx.doi.org/10.1145/1553374.1553453

[20] M. Norouzi, M. Ranjbar, and G. Mori, "Stacks of convolutional Restricted Boltzmann Machines for shift-invariant feature learning," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, jun 2009, pp. 2735–2742.

[21] H. Luo, P. L. Carrier, A. C. Courville, and Y. Bengio, "Texture modeling with convolutional spike-and-slab rbms and deep extensions," *CoRR*, vol. abs/1211.5687, 2012.

[22] M. Ranzato, A. Krizhevsky, and G. E. Hinton, "Factored 3-Way Restricted Boltzmann Machines For Modeling Natural Images," in *Proc. Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.165.2726

[23] Y. Bengio and O. Delalleau, "Justifying and generalizing contrastive divergence," *Neural Comput.*, vol. 21, no. 6, pp. 1601–1621, Jun. 2009. [Online]. Available: http://dx.doi.org/10.1162/neco.2008.11-07-647

[24] G. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, p. 2002, 2000.

[25] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio, "Why does unsupervised pre-training help deep learning?" *J. Mach. Learn. Res.*, vol. 11, pp. 625–660, Mar. 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1756006.1756025