Modout: Learning Multi-modal Architectures by Stochastic Regularization

Fan Li[1], Natalia Neverova[2], Christian Wolf[3], and Graham Taylor[1]
[1] School of Engineering, University of Guelph, Guelph, ON, Canada
[2] Facebook, Paris, France
[3] LIRIS, INSA-Lyon, Lyon, France

***Abstract*— Model selection methods based on stochastic regularization have been widely used in deep learning due to their simplicity and effectiveness. The well-known Dropout method treats all units, visible or hidden, in the same way, thus ignoring any *a priori* information related to grouping or structure. Such structure is present in multi-modal learning applications such as affect analysis and gesture recognition, where subsets of units may correspond to individual modalities. Here we describe Modout, a model selection method based on stochastic regularization, which is particularly useful in the multi-modal setting. Different from other forms of stochastic regularization, it is capable of learning whether or when to fuse two modalities in a layer, which is usually considered to be an architectural hyper-parameter by deep learning researchers and practitioners. Modout is evaluated on two real multi-modal datasets. The results indicate improved performance compared to other forms of stochastic regularization. The result on the Montalbano dataset shows that learning a fusion structure by Modout is on par with a state-of-the-art carefully designed architecture.**

I. INTRODUCTION

Multi-modality is a common setting in the fields of gesture, activity, and emotion recognition, as cues captured from multiple sensors, such as color and depth video, audio, mocap, physiological data, etc., are often useful for making predictions. Recently, deep learning methods have proven effective on various multi-modal learning problems due to their ability to learn complex and useful representations in a domain-agnostic way [16], [17], [22]. However, fusing multiple modalities effectively is an unsolved problem. It is already well-known that good results are not likely to be achieved by simply concatenating features belonging to different modalities into a single "fully-connected" layer [16]. Previous work has primarily focused on multi-modal analysis of RGB-D action videos [20], [27], [28]. For example, [27] proposed carefully designed multi-modal layers for RGB-D object recognition, which fuses color and depth information by enforcing the transformed features to share a common part. [20] also attempted to discover the shared and informative components of RGB-D signals using a deep autoencoder-based nonlinear common component analysis. But the generalized performance of these methods to modalities beyond RGB-D videos is unknown. [16] explored the fusion of multiple modalities including RGB-D videos, mocap, and audio. They used a carefully-designed network architecture to gradually fuse modalities, and they found empirically that it is better to fuse modalities that have higher correlation (e.g., visual modalities first, then motion capture, then audio).

Learning what and when to fuse is a specific instance of the more general problem of learning model structure. Global aspects of structure, such as the number of layers or number of units in a layer, are typically treated as model hyper-parameters, and are found via techniques like grid search, random search, or more recently, model-based optimization [19]. Unfortunately, this separates learning a model architecture from learning its parameters, which can convolute the process of model search. There are a few recent cases in which researchers have considered learning model structure and parameters jointly. [3] showed how depth and width can be gradually added to Inception networks by transferring knowledge from one trained network to another. [14] proposed "Blockout", a method for simultaneous regularization and model selection via structured (masked) weight matrices. Blockout, and its predecessors [29] are motivated by the desire to learn separate features for subsets of related categories for recognition problems. In other words, they are focused on the "top" of the network, towards the classifier. In this work, we are primarily interested in the way that modality-specific representations are fused, so we start from the "bottom". What is common to these approaches, and ours, is a desire to perform structure learning via backpropagation.

In the context of multi-modal gesture recognition, [16] introduced a Dropout-like regularization scheme called ModDrop. During training, ModDrop randomly removes the input from one or more modalities. This was shown, at test time, to improve robustness to corruption or loss of modalities. Our proposed Modout algorithm in this paper takes a different approach than ModDrop [16]. Instead of dropping the units belonging to a modality, in Modout the connections between the units in two adjacent layers are dropped with prior knowledge of modality-specific groupings. It has mainly two advantages over ModDrop. First, Modout can learn whether and when to fuse two modalities by optimizing the probabilities of dropping the connections between the two modalities. Second, Modout can be applied to any layer – not just the input layer. Although outside the scope of this paper, Modout could, in theory, apply to other types of known groupings beyond modalities.

This work originally appeared as an extended abstract in [12].

II. RELATED WORK

Regularization is a crucial component of training large neural networks, and advances in regularization have played a role in deep learning's advancement across large-scale applications. Traditional methods of regularization such as early stopping, weight decay, weight constraints, or addition of noise during training can be viewed as a means of limiting the capacity within a model and therefore its ability to overfit.

A new class of regularization methods that are stochastic have been widely used in deep learning due to their simplicity and effectiveness. At training time, these methods randomly remove certain structural elements of the network for each presented example, or collection of examples. The elements can be hidden or visible units (Dropout [21]), connections (DropConnect [26]), or even layers (stochastic depth [8]). At test time, the original network is used for prediction with a rescaling factor to compensate for the absence of elements during training. By pruning the network in a stochastic manner, stochastic regularization methods can be considered as a kind of ensemble that improves generalization via model averaging.

In the standard Dropout method, all units in a layer are dropped at the same rate, and therefore it ignores any structuring of the inputs which may result in more correlation among certain inputs. For example, pixels in an image are more correlated if they are spatially adjacent to each other. Also, for multi-modal learning, there are more correlations for features within a modality. Recently, several variants of Dropout have been proposed which aim to exploit this correlation. Tompson et al. [25] proposed SpatialDropout for convolutional layers, in which adjacent pixels in the drop-out feature maps are either all dropped-out or all preserved. Neverova et al. [16] proposed ModDrop for multi-modal learning, in which the input features belonging to the same modalities are either all dropped-out or all preserved. These methods have been shown to outperform standard Dropout, while their drop-out rates are pre-defined hyper-parameters.

[10] have proposed a method of learning the structure of deep neural networks via deterministic regularization. They insert a sparse diagonal matrix between each pair of fully connected layers, with entries of l_1 penalized. This implicitly defines the size of the effective weight matrices at each layer, and has a similar effect to Dropout.

An exception to the Dropout-variants is Blockout [14], which is also strongly relevant to our work. Blockout generalizes Dropout by introducing cluster assignments for each unit. Both the (implicit) Dropout rates and the parameters are learned using backpropagation. Similar to Dropout and DropConnect, Blockout does not use the information regarding structural groupings among units, and the number of clusters needs to be set and tuned. Instead, at every layer, Modout ties the clusters to the modalities, and only learns the probability of fusion between each pair of modalities. The result is a substantial reduction in number of free parameters and one less hyperparameter to tune.

III. MODEL DESCRIPTION

In this section, we give an overview of the Modout method. We first introduce its formulation as a modality-aware, weight masking stochastic regularizer. We then deal with the non-differentiability of the sampling step as it relates to gradient backpropagation.

A. Parameter Regularization via a Stochastic Mask

Most stochastic regularization methods can be considered as applying a stochastic mask to the weight matrix. A standard feed-forward network layer j with n_j units and stochastic weight-masking can be written as:

$$\mathbf{x}_j = \sigma(W'_j \mathbf{x}_{j-1}) = \sigma((M_j \circ W_j)\mathbf{x}_{j-1}) \tag{1}$$

where $\mathbf{x}_j$ denotes layer output, $\sigma(\cdot)$ is the sigmoid activation function, $\circ$ denotes the Hadamard product (i.e., elementwise multiplication), W is a unconstrained weight matrix, W' is a masked weight matrix, and M is a stochastic mask. The bias term is included in the weight matrix to simplify notation.

Different masks for different stochastic regularization methods are shown in Fig. 1. For Dropout,

$$M_j = \mathbf{m}_j \mathbf{m}_{j-1}^T \tag{2}$$

where $\mathbf{m}_j$ represents a binary vector of n_j, $\mathbf{m}_j \sim \text{Bernoulli}\,(p_j)$, where p_j is the drop-out rate in layer j; for DropConnect, $M_j \sim \text{Bernoulli}\,(p_j)$. For Blockout,

$$M_j = \frac{1}{K} C_j C_{j-1}^T. \tag{3}$$

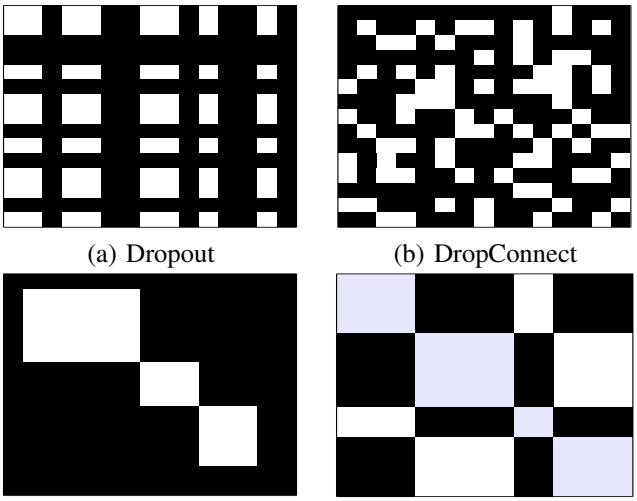
K is the predefined number of clusters, C_j is a $n_j \times K$ binary cluster assignment matrix, and $C_j \sim \text{Bernoulli}(P_j)$, where P_j is a probability matrix of the same size of C_j. So for a connection between unit s in layer $j-1$ and unit t in layer j, the probability of being dropped is

$$p_{s,t} = \frac{1}{K} \mathbf{p}_{j-1,s}^T \mathbf{p}_{j,t}. \tag{4}$$

Therefore, the mask generated by Blockout shown in Fig. 1(c) is the same as DropConnect, but generated using different probabilities for different units and different clusters.

The proposed Modout method is similar to Blockout in the sense that units are assigned to clusters. But instead of generating the cluster assignments randomly, the clusters in Modout are assigned based on knowledge of modalities. We assume that N_m modalities (paths) are input to the network, as illustrated by the example given in Fig. 3 showing different paths for gesture recognition. The Modout layers fuse the different paths – in Fig. 3 they replace the handcrafted and manually optimized network structure shown in the red dashed rectangle.

Therefore, in our case, each unit is assigned a unique cluster label, while in Blockout units can be assigned to more than one cluster. The number of units that belong to each modality in a hidden layer can be set to be proportional to the number of features in the input layer if it is not otherwise specified. During the entire process, the cluster



(a) Dropout (b) DropConnect

(c) Blockout (d) Modout

Fig. 1. Stochastic weight masks (M) for different stochastic regularization methods. Black regions mean the weights in the same location are made zero in the forward pass, and white (stochastic) or blue (deterministic) means the weights are preserved. Note that (c) only shows a special case of Blockout when each unit belongs to no more than one cluster, and also the rows and columns have been arranged.

assignments for all the units remain the same. Different from Blockout, which learns the probability of assigning units to clusters, Modout learns the probabilities of connecting the units belonging to different modalities.

In Modout, given N_m modalities, the stochastic mask M_j for layer j is defined as

$$M_j = C_j U_j C_{j-1}^T \tag{5}$$

where C_j is a $N_j \times N_m$ binary matrix and U_j is a $N_m \times N_m$ binary matrix, $U_j \sim \text{Bernoulli}(P_j)$. P_j is a modality-wise probability matrix in the same size as C_j, its elements are in the range of $[0, 1]$.

The binary mask C_j is used to control the assignment of units to modalities, and fixed during training. Also, each unit is assigned to exactly one modality, so that $C_j U_j C_{j-1}^T$ gives a binary block-structured matrix as shown in Fig. 1 (d). We note that unlike ModDrop, which is only used for the input layer, we extend the concept of modalities *over the network*, grouping units by "modalities" throughout the network from the input up to the penultimate layer. In our experiment, the number of hidden units assigned to each modality is simply set to be evenly distributed.

In our work, P is trained together with the rest of the network parameters. However, the diagonal elements of P are fixed to unity in order to guarantee that all the signals for a modality can be passed to the units which belong to that modality in the next layer.

The off-diagonal elements of U describe if two modalities need to be fused. If U is an identity matrix, the layer is equivalent to multiple independent layers for different modalities (i.e. no cross-talk); if U is a matrix of ones, all the modalities are fused in this layer; in other cases, only some of the modalities are fused. The corresponding architectures are shown in Fig. 2.

At inference time, it is necessary to average over the stochastic U's which define the connectivity to approximate an ensemble of fusion architectures. The adjusted weight matrix we use is given by:

$$\mathbb{E}\left[W_j \circ C_j U_j C_{j-1}^T\right] = W_j \circ C_j P_j C_{j-1}^T. \tag{6}$$

We note that the number of additional parameters to learn for a mask is only $N_m(N_m - 1)$, which is significantly less than Blockout which requires learning K probabilities for each unit. A summary of ModDrop, Blockout, and Modout is shown in Table I.

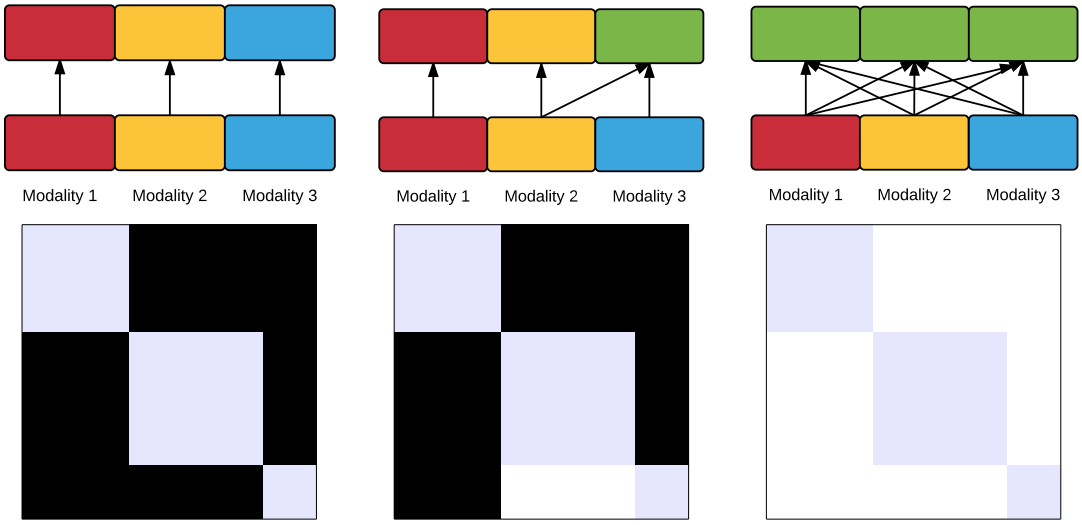
B. Learning Modality Fusion via Gradient Descent

To learn the probability matrix P, ideally we want to update it via gradient descent like the other parameters in the network. However, the gradient of P is not available because P is related to the cost function by sampling, which is not differentiable. [14] addressed a similar problem when attempting to compute the gradient of the loss with respect to the cluster probabilities parameterizing the cluster assignments. They simply used the gradient of the loss with respect to the cluster assignments, masked by the assignment matrix such that the gradient of unselected clusters is zero. Here, we derive an alternative solution using the re-parametrization trick [6]. The purpose of the trick is to make the stochasticity an input to the network instead of an operator, so that the entire network can be considered as deterministic. Here, we define r as a random number generated uniformly from $[0, 1]$. Also, we introduce free variables to be optimized as P' where $P = \sigma(P')$. The P' are unconstrained while $P \in [0, 1]$.

Using the re-parametrization trick, the binary matrix U can be re-formulated into

$$U = \frac{1 + \text{Sign}(\sigma(P') - r)}{2} \tag{7}$$

where $\text{Sign}(x)$ is the sign function, which is 1 if $x \geq 0$, or



(a) independent (b) fused (c) fully-connected

Fig. 2. Three typical fusion architectures achievable by Modout, and their corresponding weight masks. Best viewed in colour.

TABLE I

COMPARISONS OF MODDROP, BLOCKOUT, AND MODOUT.

	Blockout	ModDrop	Modout
Cluster assignment	stochastic	static based on knowledge of modalities	same as ModDrop with extension to hidden layers
Num of clusters assigned to each unit	non-deterministic	one and only one	one and only one
Applicable to hidden layers	yes	no	yes
Probabilities to learn	clusters assigned to units	n/a	connections between different modality groups
Num of free parameters to learn	$K \cdot N_j$	n/a	$N_m(N_m - 1)$

otherwise -1, and $\sigma(\cdot)$ is the Sigmoid function[1].

The sign function has zero gradient almost everywhere, making direct gradient descent still not applicable. We instead apply the "straight-through estimator" which has been used for estimating and propagating gradients through stochastic discrete units [1]. This estimator has been used in other problems such as binarized neural networks [9]. Here we use the same notation as [9]. Considering $q = \text{Sign}(t)$, and abbrevating the partial derivates of the loss L with respect to t and q as $g_t = \frac{\partial L}{\partial t}$ and $g_q = \frac{\partial L}{\partial q}$ respectively, the straight-through estimator is defined as

$$g_t = g_q 1_{|t|\leq 1}. \tag{8}$$

It basically preserves the gradient when t is small while ignoring the gradient when t is too large. Since $|\Sigma(P') - 0.5| \leq 0.5 < 1$, g_t can be always replaced by g_q in our problem. Therefore, we have

$$\frac{\partial U}{\partial P} = \frac{1}{2}. \tag{9}$$

Below are the derivatives of the variables to update in Layer j. The gradient of the loss with respect to the unmasked weight matrix W can be computed as

$$\frac{\partial L}{\partial W_j} = \frac{\partial L}{\partial W'_j}\frac{\partial W'_j}{\partial W_j} = \frac{\partial L}{\partial W'_j} \circ C_j U_j C_{j-1}^T, \tag{10}$$

[1]Noting here that we overload the σ notation. In the previous sub-section it was reserved to be a generic element-wise nonlinearity.

where $\frac{\partial L}{\partial W'_j}$ is calculated by the product of all gradients from the loss function backwards down to the jth layer using the chain rule, as in standard weight updates made by backpropagation.

Considering Eq. 9 and using the chain rule, the gradient of P' can be derived as:

$$\frac{\partial L}{\partial P'_j} = \frac{\partial L}{\partial U_j}\frac{\partial U_j}{\partial P_j}\frac{\partial P_j}{\partial P'_j} = \frac{\sigma(P')(1-\sigma(P'))}{2}\frac{\partial L}{\partial U_j}, \tag{11}$$

and

$$\frac{\partial L}{\partial U_j} = \frac{\partial L}{\partial W'_j}\frac{\partial W'}{\partial U_j} = \frac{\partial L}{\partial W'_j}(W_j \circ C_j C_{j-1}^T). \tag{12}$$

This way of updating the probabilities via Eq. 11 is similar to [14] because Eq. 11 can also be considered as updating the probabilities by the gradient of the cluster assignments. The only difference is that Eq. 11 uses *all* the cluster assignments, not only the selected ones, with a rescaling factor of $1/2$. Though there is no significant difference between the two methods according to the experiments, our proposed update has a more principled derivation.

IV. EXPERIMENTS

In this section, we investigate the ability of Modout to learn model structure competitive with other stochastic regularization methods on three datasets.

TABLE II

TEST ERROR (%) ACROSS DIFFERENT REGULARIZERS ON THE MULTI-MODAL MNIST DATASET.

	Backprop	Dropout	Blockout	ModDrop	ModDrop+Dropout	Modout	Modout+Dropout
Valid	1.70 ± 0.04	1.11 ± 0.03	1.25 ± 0.03	1.42 ± 0.06	0.97 ± 0.03	1.05 ± 0.09	0.86 ± 0.02
Test	1.87 ± 0.06	1.16 ± 0.03	1.19 ± 0.07	1.59 ± 0.09	1.08 ± 0.05	1.18 ± 0.08	**0.99 ± 0.04**

A. Multi-Modal MNIST Dataset

First, we compare the related algorithms using a simulated multi-modal dataset created from the MNIST dataset [11], a well-known benchmark popular in the deep learning community. MNIST consists of 28×28 grayscale images of handwritten digits and their associated labels. 60,000 images are used for training and 10,000 images are used for testing.

The multi-modal setting in our experiment is similar to [16]. All of the images are split evenly into 4 quarters, each of which is treated as one "modality". Compared to the real modalities, features in different simulated modalities are more similar and highly-correlated because they come from the same source.

We fix the architecture to be a multi-layer perceptron (MLP) with three hidden layers, without data augmentation or data preprocessing. The rectified linear unit [15] is used as the activation function for all the units. Different from [16] that tried to optimize the number of units, the numbers of units in the hidden layers are set to $[1200, 1200, 40]$. Each modality is first pretrained using an MLP of two hidden layers (400 units for each) with Dropout. The dropout rates of Dropout, ModDrop, and the Modout + Dropout combination are set to 0.2 for the first layer and 0.5 for the other layers. For Blockout and Modout, the probabilities of dropping are initialized to 0.5 to maximize the uncertainty as suggested in [25]. Note that when we apply Modout + Dropout, there is both a Dropout and Modout rate. The number of clusters in Blockout is optimized to 2. Each test is repeated 10 times with different random initialization of the parameters.

The result is shown in Table II. We see that Dropout with carefully chosen dropout rates achieves significantly better performance than standard back-propagation without using any stochastic regularization, and performs comparably to Blockout, which learns dropout rates automatically. ModDrop has been shown to be robust to missing or corrupted modalities [16], but it is not comparable to Dropout because it is only performed on the input layer. Combining ModDrop with Dropout improves results significantly and performs better using Dropout alone. The combination of Modout and Dropout achieves a state-of-the-art 0.991 test error rate on multi-modal MNIST using a simple MLP structure.

B. CORNELL Activity Dataset (CAD-60)

The second dataset that we considered was the Cornell Human Activity Dataset [23] which is a widely used benchmark dataset for human activity recognition. The dataset consists of RGBD images obtained from a Kinect sensor that recorded activities of 4 different persons performing 12 unique activities (plus one activity defined as "random movement") in unstructured indoor environments, yielding 60 total cases. Activities include actions such as brushing teeth, relaxing on a chair, writing on a white-board and drinking water.

Five different descriptors that captured body pose and motion information as reported in [24] were used as inputs to the multi-modal fusion network. It is trivial to add additional inputs to the multi-modal fusion architecture, for example, adding raw image frames or dense optical flow information as we have designed the multi-modal fusion architecture to be flexible in the number of input modalities that could be fused. The *Body Pose, Hand Position and Motion Information* descriptors were concatenated to form a 459-dimensional skeletal feature. This combined skeletal feature defines one modality. The *SimpleHOG* and *SkeletalHOG* descriptors were computed for both depth and RGB frames, each forming a separate modality, for a total of four visual modalities and one skeletal modality.

In our experiments, we used the so-called "*new person*" setting where the model was trained on three persons, and tested on the fourth. 70% of the data was used for training and 30% was used for testing. There is a total of 907 features from five modalities. A MLP with three fully-connected layers and one softmax layer is adopted for evaluation. The number of neurons in each of the two hidden layers is set to 1500, and the number of output classes is 12. Different stochastic regularization methods are applied to the hidden layers.

The results shown in Table III demonstrate that the use of either Dropout or Blockout does not improve the performance on this dataset. Using ModDrop with Dropout can slightly improve the performance. The combination of Modout and Dropout achieves the best overall result. However, it is still largely behind the reported state-of-the-art result [4]. This is mainly due to our simple, fixed front-end compared to other works that use convolutional architectures.

C. Montalbano Gesture Recognition Dataset

The Montalbano dataset, originally released as part of the Chalearn 2014 Looking at People Challenge (Track 3) [5], is used to evaluate our approach. It consists of 13,858 instances of Italian conversational gestures performed by different people and recorded with a consumer RGB-D sensor. It includes color, depth video and mocap (articulated pose) streams. The gestures are drawn from a large vocabulary, from which 20 categories are identified to be detected and recognized, while the rest are considered as arbitrary movements. Each gesture in the training set is accompanied by a ground truth label as well as information about its start-and-end points. An additional audio modality not in the 2014 competition was re-synced and added in [16], which we also consider.

The input to the network is a so-called *dynamic pose* consisting of synchronized multi-modal measurements concatenated from several frames temporally spaced with a given

TABLE III

CLASSIFICATION ERROR (%) USING DIFFERENT METHODS ON THE CAD-60 DATASET.

		Backprop	Dropout	Blockout	ModDrop+Dropout	Modout+Dropout
CV-1	valid	17.84	19.48	16.87	18.05	18.18
	test	**29.81**	30.44	32.42	33.75	31.54
CV-2	valid	29.19	29.30	29.79	26.92	29.59
	test	18.22	19.68	17.99	20.19	**17.96**
CV-3	valid	24.26	25.26	23.03	18.45	20.89
	test	43.67	44.32	41.58	**38.67**	42.03
CV-4	valid	28.03	28.03	23.91	23.90	26.56
	test	18.51	16.98	19.03	16.80	**15.68**
average	valid	24.83	25.52	23.40	23.90	23.81
	test	27.55	27.85	27.76	27.35	**26.80**

stride. In [16], different networks are trained on different strides and results are combined. In our experiments, we only use a single temporal stride of 4 for sampling for all the modalities, as the result is very close to using a combination of strides as also shown in [16].

1) Network Architecture and Experimental Setup: The network architectures for testing are based on the one in [16], as shown in Fig. 3. In that work, each modality is pre-trained as an individual classifier (video modalities use two-stage convolutional networks, the audio stream uses a one-stage convolutional network, and the mocap stream uses a MLP with two hidden layers). The penultimate layer of each modality-specific classifier is then connected via a shared hidden layer to a softmax output and the whole system is then trained by a two-stage procedure. First, the weights to and from the shared layer are initialized such that the overall network performs a simple fusion of modalities. Then, this constraint is gradually relaxed to permit a more flexible fusion strategy. In addition to this careful initialization and relaxation process, prior knowledge influences the *order* in which modalities are fused. The depth and intensity channels corresponding to each hand are fused (at HLV1 in Fig. 3), while cross-modality fusion involving the other channels are postponed until the shared layer. This network architecture achieved first place out of 17 teams in the ChaLearn 2014 Looking at People Challenge (gesture recognition track) [5].

Two experiments are conducted for performance evaluation of Modout, where the first aims to compare the performance of Modout with other stochastic regularization methods using a simple network architecture, namely a MLP. Due to the high dimensionality and limited number of training sequences, directly applying a MLP to the raw data leads to poor generalization. Instead, the intermediate outputs from the first fully-connected layer of the pre-trained classifiers are used as the input features for each modality. The number of total input features is 2600, including 800 audio features, 900 mocap features, 450 color features, and 450 depth features. Color and depth features have been concatenated into a single modality to limit the number of modalities (they are split in the second experiment). Methods for comparison include standard (non-regularized) backpropagation, Dropout, Blockout, ModDrop, Modout, and the combination of Modout and Dropout. A MLP with two hidden layers followed by a softmax regression layer is used for all tests. The number of units in each hidden layer is set to 3,000. This experiment was performed on data released early in the challenge. The standard practice of removing frames with no gesture present was applied during preprocessing. Frames are divided into training, validation, and testing data using the same split as in [16].

The second experiment aims to evaluate the performance of Modout by integrating it into the network architecture in [16]. This is done by concatenating each of the last two layers for each modality into a single layer and adding connections which are modulated by Modout. Thus, the network structure in the red box in Fig. 3 becomes a MLP with two hidden layers and one softmax regression layer. The real test data released later in the challenge are used for testing. Similar to [16], we first use a motion detector to remove the frames without motion in the test data, and then use the learned model to classify all the frames. The Jaccard index is used to measure performance:

$$J_{s,n} = \frac{|A_{s,n} \cap B_{s,n}|}{|A_{s,n} \cup B_{s,n}|}, \tag{13}$$

where $A_{s,n}$ and $B_{s,n}$ denote the binary ground truth and predictions for gesture category n and sequence s respectively. The final score is measured by the mean Jaccard index over all categories and sequences (higher is better).

2) Experimental Results and Analysis: Table IV shows the result of different stochastic regularization methods using the pre-trained intermediate representation of each modality as input. The drop-out rate is set to 0.2 for the first hidden layer and 0.5 for the second hidden layer. The number of clusters in Blockout is set to 2 which was found empirically to have the best performance. For both Blockout and Modout, the probabilities are initialized to 0.5 in order to maximize uncertainty at the beginning of training. The results show that there is no significant difference between Dropout and ModDrop, while Modout and its combination with Dropout significantly outperform the other methods which ignore modality information. In this first experiment we report performance on classification of dynamic poses (as opposed to gesture localization), therefore the metric used is classification accuracy. The best test accuracy is achieved by a combination of Modout and Dropout.

Table V gives results on full gesture detection and localization reported as Jaccard Index (Eq. 13). It shows that our approach (Modout + Dropout) achieves a score of 0.888, which is higher than [16] using either Dropout

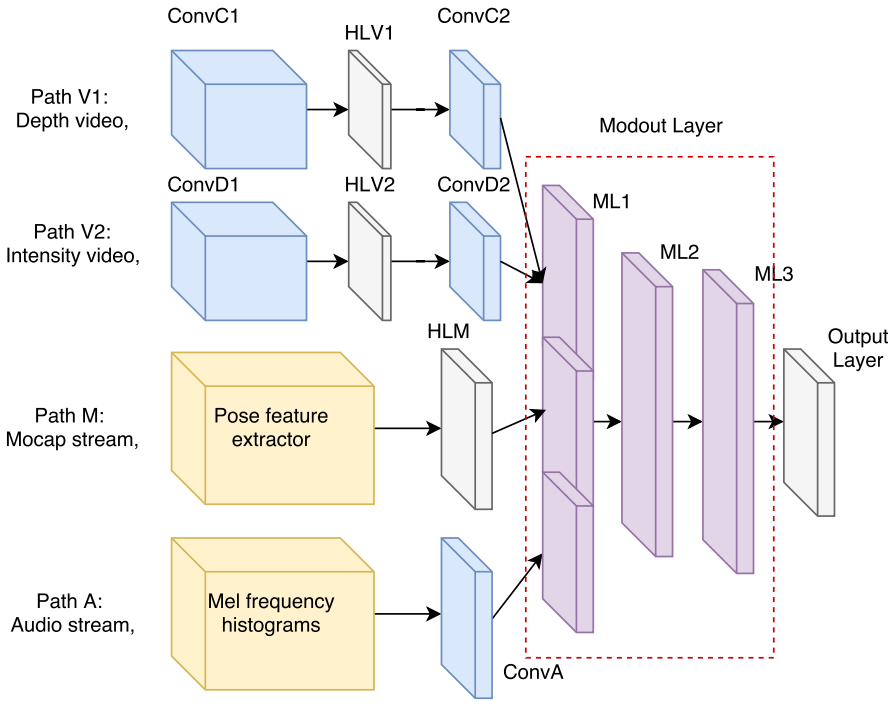


Fig. 3. Network architecture for multi-modal gesture recognition (reproduced from [16]). The structure in the red box is modified into a MLP in the second experiment.

TABLE IV

CLASSIFICATION ACCURACY(%) OF DIFFERENT STOCHASTIC REGULARIZATION METHODS ON MONTALBANO. NOTE THAT WE FOLLOW THE SAME EVALUATION PROCEDURE AS [16] AND [18].

	BackProp	Dropout	Blockout	ModDrop +Dropout	Modout	Modout +Dropout
Validation accuracy	91.1	91.5	91.7	92.1	91.6	**92.9**
Test accuracy	92.0	92.5	92.6	92.4	93.6	**93.8**

or ModDrop + Dropout on a similar but carefully chosen fusion architecture. Compared to previously reported results, our result is only eclipsed by the state-of-the-art, [18] which uses a combination of temporal convolution layers and Long Short-Term Memory (LSTM). One possible reason is that the temporal correlation between two adjacent spatio-temporal blocks is not considered in our approach. Our result could be further improved by using a simple 1-D Markov Random Field model as a post-processing step.

V. USING MODOUT FOR STRUCTURE PRUNING

To demonstrate that the probabilities learned by Modout are non-trivial, we show the improvement of using the probabilities to prune the network structure. A binary mask is created using the probabilities, e.g., the connections with a probability of lower than 0.5 are removed. The pruned structure is compared with early fusion and late fusion, which are the two most common fusion strategies. They are essentially two extremes of network structures: early fusion corresponds to the network structure without any pruning, and late fusion corresponds to the structure where the off-diagonal connections of all but the last layer are pruned.

The experiment is performed on two datasets: the multi-modal MNIST dataset and the Montalbano dataset considered above. Similar to the previous experiment, four modalities of evenly-split sub-images from the MNIST dataset, and intermediate features of three modalities including video, skeleton, and audio from the Montalbano dataset are used. The network is first trained with Modout. Then the network is pruned using the learned probabilities, and the model is trained again. A similar idea of pruning the network structure followed by re-training can be found in [7].

The results are provided in Table VI. We see that early fusion is better than late fusion for the MNIST dataset, while late fusion is better for the Montalbano dataset. The rationale behind this pattern is that the modalities of the MNIST dataset are highly correlated because they are from the same source, while the audio, video, and skeleton modalities in the Montalbano dataset are less correlated. We can also see that the structure learned by Modout outperforms both early fusion and late fusion. For the case of Montalbano, we also see that binarizing the probabilities and fine-tuning deterministically learns a superior model.

VI. CONCLUSIONS

We have presented Modout, a generalization of ModDrop, which is useful for multi-modal learning. It can be applied to multiple layers, and is capable of learning modality fusion.

TABLE V

COMPARISON OF STATE-OF-THE-ART RESULTS RECENTLY PUBLISHED FOR THE MONTALBANO TASK.

Approach	Jaccard index
Wu et al. (2016) (HMM, DBM, 3DCNN) [30]	0.809
Chang (2014) (MRF, KNN, HoG) [2]	0.827
Monnier et al. (2014) (AdaBoost, HoG) [13]	0.834
Neverova et al. (2016) (Dropout) [16]	0.876
Neverova et al. (2016) (ModDrop + Dropout) [16]	0.880
Pigou et al. (2015) (Temp Conv + LSTM) [18]	**0.906**
Ours (Modout + Dropout)	0.888

TABLE VI

COMPARISON WITH EARLY FUSION AND LATE FUSION (ERROR RATES IN PERCENTAGES).

	MNIST	Montalbano
Modout	**1.03**	6.44
Early fusion	1.19	7.23
Late fusion	1.88	6.94
Re-trained using learned structure	1.04	**6.01**

While motivated by the challenge of learning fusion structure, Modout can leverage any known grouping of inputs.

We presented experimental results on three multi-modal datasets, which show that Modout outperforms other stochastic regularization methods, and achieves close to the state-of-the-art for gesture recognition. Future work includes applying the approach to other types of neural network structures and validating on other multi-modal datasets, specifically those with a high number of modalities.

REFERENCES

[1] Y. Bengio, N. Léonard, and A. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[2] J. Y. Chang. Nonparametric gesture labeling from multi-modal data. In *Computer Vision-ECCV 2014 Workshops*, pages 503–517, 2014.

[3] T. Chen, I. Goodfellow, and J. Shlens. Net2Net: Accelerating learning via knowledge transfer. *arXiv preprint arXiv:1511.05641*, 2015.

[4] E. Cippitelli, S. Gasparrini, E. Gambi, and S. Spinsante. A human activity recognition system using skeleton data from RGBD sensors. *Computational Intelligence and Neuroscience*, 2016.

[5] S. Escalera, X. Baró, J. Gonzalez, M. A. Bautista, M. Madadi, M. Reyes, V. Ponce-López, H. J. Escalante, J. Shotton, and I. Guyon. Chalearn looking at people challenge 2014: Dataset and results. In *ECCV 2014 Workshops*, pages 459–473. Springer, 2014.

[6] I. Goodfellow, Y. Bengio, and A. Courville. Deep learning. Book in preparation for MIT Press, 2016.

[7] S. Han, J. Pool, J. Tran, and W. Dally. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1135–1143, 2015.

[8] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Weinberger. Deep networks with stochastic depth. *arXiv preprint arXiv:1603.09382*, 2016.

[9] I. Hubara, D. Soudry, and R. E. Yaniv. Binarized neural networks. *arXiv preprint arXiv:1602.02505*, 2016.

[10] P. Kulkarni, J. Zepeda, F. Jurie, P. Pérez, and L. Chevallier. Learning the structure of deep architectures using L1 regularization. In *Proceedings of the British Machine Vision Conference*, 2015.

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[12] F. Li, N. Neverova, C. Wolf, and G. W. Taylor. Modout: Learning to fuse modalities via stochastic regularization. In *2nd Annual Conference on Vision and Imaging Systems (CVIS) - Extended Abstract*, 2016.

[13] C. Monnier, S. German, and A. Ost. A multi-scale boosted detector for efficient and robust gesture recognition. In *ECCV 2014 Workshops*, pages 491–502, 2014.

[14] C. Murdock, Z. Li, H. Zhou, and T. Duerig. Blockout: Dynamic model selection for hierarchical deep networks. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2016.

[15] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

[16] N. Neverova, C. Wolf, G. Taylor, and F. Nebout. ModDrop: adaptive multi-modal gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 38(8):1692–1706, 2016.

[17] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng. Multimodal deep learning. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, pages 689–696, 2011.

[18] L. Pigou, A. v. d. Oord, S. Dieleman, M. Van Herreweghe, and J. Dambre. Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video. *arXiv preprint arXiv:1506.01911*, 2015.

[19] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.

[20] A. Shahroudy, T.-T. Ng, Y. Gong, and G. Wang. Deep multimodal feature analysis for action recognition in RGB+D videos. *arXiv preprint arXiv:1603.07120*, 2016.

[21] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[22] N. Srivastava and R. Salakhutdinov. Multimodal learning with deep boltzmann machines. In *Advances in Neural Information Processing Systems (NIPS) 25*, pages 2222–2230. 2012.

[23] J. Sung, C. Ponce, B. Selman, and A. Saxena. Human activity detection from RGBD images. In *In AAAI workshop on Pattern, Activity and Intent Recognition (PAIR*, 2011.

[24] J. Sung, C. Ponce, B. Selman, and A. Saxena. Unstructured human activity detection from RGBD images. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 842–849, May 2012.

[25] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler. Efficient object localization using convolutional networks. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, pages 648–656, 2015.

[26] L. Wan, M. Zeiler, S. Zhang, Y. LeCun, and R. Fergus. Regularization of neural networks using DropConnect. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pages 1058–1066, 2013.

[27] A. Wang, J. Cai, J. Lu, and T.-J. Cham. MMSS: Multi-modal sharable and specific feature learning for RGB-D object recognition. In *Proceedings of the International Conference on Computer Vision*, pages 1125–1133, 2015.

[28] Z. Wang, R. Lin, J. Lu, J. Feng, et al. Correlated and individual multimodal deep learning for RGB-D object recognition. *arXiv preprint arXiv:1604.01655*, 2016.

[29] D. Warde-Farley, A. Rabinovich, and D. Anguelov. Self-informed neural network structure learning. In *International Conference on Learning Representations (ICLR) Workshop Track*, 2014.

[30] D. Wu, L. Pigou, P.-J. Kindermans, N. Le, L. Shao, J. Dambre, and J.-M. Odobez. Deep dynamic neural networks for multimodal gesture segmentation and recognition. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 38(8):1583–1597, 2016.