# A DEEP LEARNING APPROACH TO MULTIPLE KERNEL FUSION

*Huan Song*[†], *Jayaraman J. Thiagarajan*[‡], *Prasanna Sattigeri*[⋆],
*Karthikeyan Natesan Ramamurthy*[⋆] *and Andreas Spanias*[†]
[†] SenSIP Center, School of ECEE, Arizona State University, Tempe, AZ
[‡] Lawrence Livermore National Labs, 7000 East Avenue, Livermore, CA
[⋆] IBM T.J. Watson Research Center, 1101 Kitchawan Road, Yorktown Heights, NY

## ABSTRACT

Kernel fusion is a popular and effective approach for combining multiple features that characterize different aspects of data. Traditional approaches for Multiple Kernel Learning (MKL) attempt to learn the parameters for combining the kernels through sophisticated optimization procedures. In this paper, we propose an alternative approach that creates dense embeddings for data using the kernel similarities and adopts a deep neural network architecture for fusing the embeddings. In order to improve the effectiveness of this network, we introduce the kernel dropout regularization strategy coupled with the use of an expanded set of composition kernels. Experiment results on a real-world activity recognition dataset show that the proposed architecture is effective in fusing kernels and achieves state-of-the-art performance.

***Index Terms*—** Kernel fusion, Deep learning, Dropout regularization, Activity recognition

## 1. INTRODUCTION

Kernel methods provide a powerful framework to extend several machine learning formulations since they enable the design of effective non-linear models. For example in Support Vector Machines (SVM), the problem of building binary classifiers to obtain non-linear decision boundaries can be reposed into a dual problem in terms of the kernel similarity matrix. Referred to as the kernel trick, this approach has been successfully applied to a wide range of supervised and unsupervised learning problems [1, 2]. A valid positive semidefinite kernel inherently defines a lifting (transformation) to a Reproducing Kernel Hilbert Space (RKHS), thereby enabling efficient approximation of any function of interest in the transformed space. Another important property of kernel methods is that fusing kernels from multiple sources (e.g. different feature descriptors or sensing modalities) is straightforward. A commonly adopted strategy is to consider a convex combination of the kernels. The process of simultaneously inferring the weights for the convex combination and minimizing the structural risk (SVM objective) is referred to as Multiple Kernel Learning (MKL) [3, 4]. The idea is to effectively exploit the complementary nature of the different features [5] and the representation power of different kernel functions. Despite their wide-spread use, as pointed out by [6], MKL algorithms may suffer when solving for global weights and the most critical support vectors, since the weight for a kernel is restricted to be the same over the whole input space. This challenge is alleviated by Localized MKL (LMKL) [6], which introduces a gating function for each kernel. By treating the input data sample as the independent variable, the gating function characterizes the underlying localities in data and reduces the number of support vectors.

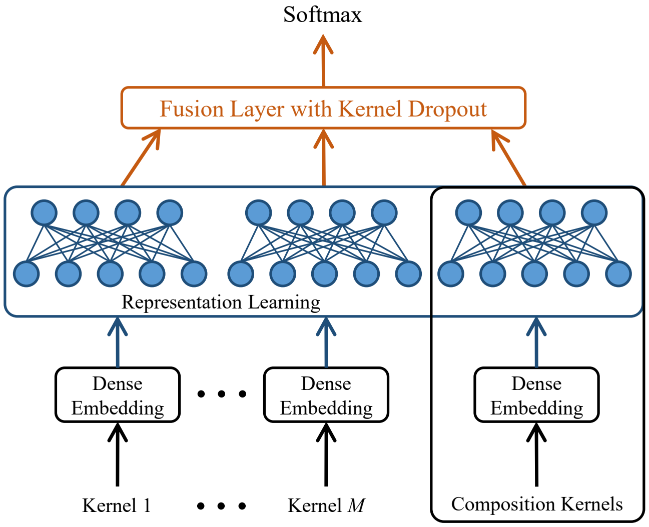

**Fig. 1**: Proposed approach for multiple kernel fusion. We use the kernel matrices to create dense embeddings for data and fuse them in a fully connected deep network. The kernel dropout regularization at the fusion layer and the use of different composition (sum) kernels improves the optimization.

Several existing approaches for feature fusion begin by building compact and effective representations from raw data since they help the fused result to be robust to noise and outliers. In particular, sophisticated representation learning paradigms such as deep learning have shown exceptional power when dealing with complex, high-dimensional data. In

This research was supported in part by the SenSIP center.

[7], the authors focused on feature learning in different multimodal settings and showed that in the multimodal fusion case, the fused feature exploits complementary information from each modality. In [8], Zhao *et.al.* built sub-networks for each heterogeneous feature and relied on the Stacked Denoising Autoencoders to learn high-level homogeneous representations for feature integration. Note that, both methods start with the raw features directly and did not exploit any use of similarity kernels. Existing works on incorporating the advantages of deep learning into kernel methods either develop new kernel constructions to mimic the large neural computation [9] or apply similar neural network structure to combine kernels and optimize at each layer [10].

In this paper, we propose to exploit the advantages of deep architectures in feature learning to build a new approach for multiple kernel fusion. First, we adopt a novel viewpoint of kernels by treating the similarities encoded in the kernel matrix as a valid embedding of the data. This is similar to the approaches in the natural language processing literature, wherein relevance measures such as Pointwise Mutual Information (PMI) of a word with respect to other words in the vocabulary is treated as a word embedding [11]. Since the kernel matrix can be inherently sparse and low-rank, we propose to apply a dense embedding procedure (e.g. Singular Value Decomposition) to the columns of the kernel matrix. Consequently, the problem of kernel fusion is transformed to fusing their dense embeddings. To this end, we build a deep architecture for kernel fusion, coupled with novel training strategies: (a) to emulate the convex combination approach in MKL, we expand the set of kernels by including combinations of different subsets of the base kernels, and (b) we perform kernel dropout in the fusion layer for improved regularization. The proposed architecture replaces the complex optimization procedure in MKL by efficient representation learning and straightforward feature merging. This makes our fusion approach easily scalable to a large number of kernels.

## 2. PROPOSED APPROACH

### 2.1. Architecture

The proposed approach considers the similarity information encoded in a kernel as an embedding of the data, and poses the problem of MKL as fusing these embeddings in a deep learning architecture. In this section, we start by presenting the general architecture and then describe strategies for improving the performance.

As shown in Figure 1, the proposed architecture consists of three components: (a) dense embeddings computation using kernel similarities, (b) representation learning using a deep architecture, and (c) feature fusion. Let us denote the kernel Gram matrix as $\mathbf{K}$, where $\mathbf{K}_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$. Each column $\mathbf{s}_j$ of the Gram matrix encodes the relevance between sample $\mathbf{x}_j$ and all other samples $\mathbf{x}_i$ and it can be treated as an embedding for $\mathbf{x}_j$. This viewpoint is very similar to the construction of dense word embeddings using the PMI in text processing [12]. In the ideal case, $\mathbf{s}_j$ has large values at the samples coming from the same class with $\mathbf{x}_j$ and zeros at others. The sparsity in these embeddings makes them unsuitable for inference tasks [11]. To alleviate this, we obtain a dense embedding of the kernel similarities using Principle Component Analysis (PCA), which projects the original kernel feature to a low-dimensional space. Note that, this can be easily replaced by other embedding techniques including manifold learning [13], Word2Vec [14] or random projection [15]. Besides providing dense embeddings, this step also helps to significantly improve the network training speed.

On top of each dense feature set obtained by PCA, we build a fully connected neural network. The goal is to use back-propogation in a large network to learn a concise representation which are more effective for inference tasks. To achieve this, the size of the network needs to be adequately large. In our application, we build a $4-$layer network separately for each embedding. At each hidden layer, dropout regularization [16] is used to prevent overfitting and batch normalization [17] to accelerate training. After the representations are learned, we stack another layer which is responsible for fusing the features and obtaining the classification result with a softmax activation. The most straightforward approach for the feature merging is to simply concatenate all the inputs to the layer. However various other merge modes can be easily applied too including summation, averaging, multiplication etc. The flexibility of the merge layer facilitates a wide range of kernel combination forms.

### 2.2. Using Composition Kernels

An important property of MKL is the various parameterization forms for mixing kernels such as convex combination, Hadamard product or mixtures of polynomials [3]. We emulate this property by including all possible combinations of base kernels (namely the composition kernels in Figure 1) to the architecture input. Given the base kernels set $\{\mathbf{K}_1, \mathbf{K}_2, ..., \mathbf{K}_M\}$, the whole input kernel set $\Phi$ to our architecture will have size $\tilde{M}$:

$$\Phi = \{\mathbf{K} \mid \mathbf{K} = \sum_{i=p}^{q} \mathbf{K}_i, \forall p, q \in \{1, 2, 3, ..., M\}, q \geq p\}$$

$$\tilde{M} = \sum_{m=1}^{M} \binom{M}{m}$$

Simple kernel summation proves to be highly effective in practical recognition problems and the derived representation corresponding to the summed kernel is often very different from either alone. Note that other formulations with base kernels can also be used. Paired with the flexible merge and deep feature representation, our architecture covers a large number of kernel combination scenarios without explicitly formulating them.

### 2.3. Kernel Dropout Regularization

In dropout regularization [16] for training large neural networks, neurons are randomly chosen to be removed from the network along with their incoming and outgoing connections. The process can be viewed as sampling a large set of possible network architectures with shared weights. Given the large kernel set $\Phi$, a more effective regularization mechanism is needed to prevent the network training from overfitting certain kernels. More specifically, we propose to regularize the fusion layer by dropping the entire representations learned from some randomly chosen kernels. Denoting the representations learned for all kernels as $\tilde{\Phi} = \{\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_{\tilde{M}}\}$ and a vector $\mathbf{t}$ associated with $\tilde{M}$ independent Bernoulli trials, the representation $\mathbf{r}_m$ is dropped from the fusion layer if $t_m$ is 0. The feed-forward operation can be expressed as:

$$
\begin{gathered}
t_m \sim \text{Bernoulli}(p) \\
\hat{\Phi} = \{\mathbf{r} \mid \mathbf{r} \in \tilde{\Phi} \text{ and } t_m > 0\} \\
\tilde{\mathbf{r}} = (\mathbf{r}_i), \mathbf{r}_i \in \hat{\Phi} \\
z_i = \mathbf{w}_i \tilde{\mathbf{r}} + b_i, y_i = f(z_i)
\end{gathered}
$$

where $\mathbf{w}_i$ are the weights for hidden unit $i$, $(\cdot)$ denotes vector concatenation and $f$ is the softmax activation function.

## 3. SYSTEM SETUP

In this section, we apply the proposed architecture to the important problem of sensor-based activity recognition. Recent advances in activity recognition have shown promising results in the applications of fitness monitoring and assistive living [18]. However, problem still exists on how to effectively deal with the measurement inaccuracy and noise. One popular approach to the problem is utilizing various features and kernels that characterize salient aspects of the data and develop efficient fusion mechanisms to combine them. In this paper, we construct kernels which describe the statistical property, periodic structure and inter-sample relations for the accelerometer signals.

### 3.1. Feature Extraction and Kernel Construction

#### *3.1.1. Statistics Kernel*

Statistical features have been known to be useful for activity recognition [19]. The features we use include mean, median, standard deviation, kurtosis, skewness and total acceleration. In addition, we extract the mean-crossing rate and dominant frequency to capture the frequency-domain information. We construct a Gaussian kernel and the best $\gamma$ parameter is determined by cross validation on the training set.

#### *3.1.2. Shape Kernel*

In lieu of building conventional state-space models, Time Delay Embeddings (TDE) provide an effective way to reconstruct the underlying dynamical system from the observed data. Given a time-series data, the phase space is the set of states which contain all the necessary information to predict the future of the system [20]. The TDEs of a time-series data $\mathbf{x}$ can be defined in matrix form $\mathbf{O}$ whose $i$th column is $\mathbf{o}_t = [x_t, x_{t+\tau}, x_{t+2\tau}, ..., x_{t+(n-1)\tau}]$.

The $n$ time-delayed observation samples can be considered as points in $\mathbb{R}^n$, which is referred to as the delay embedding space. In our application, the delay parameter $\tau$ is fixed to 10 and embedding dimension $n$ to 8. Following the approach in [20], we use PCA to project the embedding to 3-D for noise reduction. We extract a simple shape function based on the geometric distances, and use it to derive our feature. The shape function we consider measures the pairwise distances between samples in the TDE space, calculated as $\mathbf{S}_{ij} = \|\mathbf{o}_i - \mathbf{o}_j\|_2$ [21]. A histogram is calculated using these distances with a pre-specified bin size to build the feature. Following this, we construct an intersection kernel $k(\mathbf{h}, \mathbf{h}') = \sum_i \min(h_i, h'_i)$ [22], where $\mathbf{h}$,$\mathbf{h}'$ are the computed histograms.

#### *3.1.3. Correlation Kernel*

Correlation measures the dependence between two time-series signals and has been widely used in electroencephalogram (EEG) signal analysis. We calculate the absolute value of Pearson correlation coefficient. To account for shift between the two signals, the maximum absolute coefficient for a range of shift values is identified. The correlation matrix $\mathbf{R}$ defined in this way does not guarantee the required positive semi-definite condition of kernel. To correct this, we remove the negative eigenvalues from the matrix. Given the eigen-decomposition of the correlation matrix $\mathbf{R} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$, where $\mathbf{\Lambda} = \text{diag}(\lambda_1, \lambda_2, ..., \lambda_n)$ and $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_r \geq 0 \geq \lambda_{r+1} \geq ... \geq \lambda_n$, the correlation kernel is constructed as $\mathbf{K} = \mathbf{Q}\hat{\mathbf{\Lambda}}\mathbf{Q}^T$, where $\hat{\mathbf{\Lambda}} = \text{diag}(\lambda_1, \lambda_2, ..., \lambda_r, 0, ..., 0)$.

### 3.2. Dataset

The dataset used in our experiments is obtained from [23] and corresponds to 12 different daily activities for 14 subjects. Each activity is repeated in 5 trials for each subject. The 3-axis accelerometer measurements were obtained at a sampling rate of 100Hz. We consider 5 seconds of non-overlapping frames and as a result there are 5353 frames. In our experiment, 80% randomly chosen samples were used for training and the rest for testing. Putting together the 3 base kernels described in Section 3.1 with the combination kernels ($\mathbf{K}_1 + \mathbf{K}_2$, $\mathbf{K}_1 + \mathbf{K}_3$ etc.) makes the total number of input kernels for our architecture to 7.

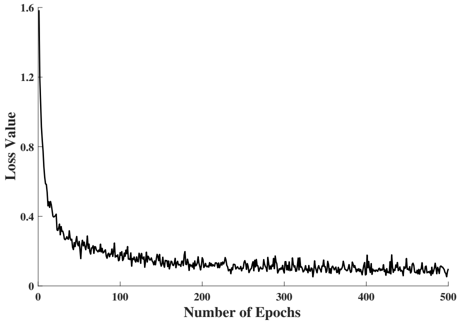

**Fig. 2**: Convergence behavior of the proposed method.

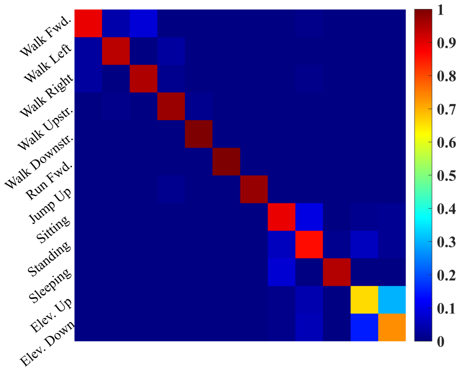

**Fig. 3**: Confusion matrix based on the classification result.

**Table 1**: Classification Performance Comparison

| **Input Kernel** | **Accuracy** (%) |
|---|---|
| Statistics | 79.3 |
| Shape | 73.3 |
| Correlation | 75.3 |
| **Architecture** | **Accuracy** (%) |
| (a): Standard Feature Fusion | 82.3 |
| (b): (a) + Dense Embedding | 86.6 |
| (c): (b) + Composition Kernels | 88.1 |
| **MKL Method** | **Accuracy** (%) |
| UNIFORM | 88.5 |
| SMO-MKL [24] | N/A |
| SwMKL [25] | 88.5 |
| **Proposed**: (c) + Kernel Dropout | **90.2** |

## 4. PERFORMANCE EVALUATION AND CONCLUSION

The proposed approach is tested on the described activity recognition dataset. In the dense embedding stage, the dimension of the kernel feature is reduced to 500. The 4-layer neural network for representation learning has size 256-1024-512-64. At each hidden layer the dropout rate is fixed at 0.5. In the fusion layer, the kernel dropout rate is set to 0.6. We use Keras library with TensorFlow backend to build and perform optimization for the architecture.

The training convergence curve shown in Figure 2 demonstrates that the architecture is able to reduce the loss value and achieve convergence quickly. We report the classification performance in Table 1. In our case, the accuracy is defined as the averaged fraction of correctly predicted labels among all classes. We make comparison of our proposed architecture to other 3 setups: (1) single kernel performance, which is obtained without the fusion layer, (2) different deep architecture settings, and (3) existing MKL methods.

First, we observe that all 3 kernels achieve accuracies in a similar range, while the fusion of them provides a significant improvement. In the best case, the improvement is over 10% compared to the best of a single kernel. Second, we compare each of the proposed training strategies to the standard deep architecture feature fusion (by simple concatenation of learned representations). Dense embedding gives around 4% improvement. This demonstrates the necessity of this preprocessing stage when treating kernel values as embeddings. The inclusion of composition kernels and kernel dropout regularization each provides further improvements. Although the improvement is not tremendous in this case, it is significant. We argue that each step is beneficial and expect much more usefulness of them in more complex problems when a large number of descriptors and kernels are needed. From the visualization of confusion matrix in Figure 3, we can see the overall classification model is highly effective to this problem and most of the confusion happens only between very related activities (e.g. elevator up versus elevator down).

We compare our approach to MKL methods including combination with uniform weights (denoted as UNIFORM), a popular MKL algorithm SMO-MKL [24] and a recent LMKL approach SwMKL [25]. UNIFORM provides a decent performance and this justifies our utilization of the composition kernels. SMO-MKL applies to binary classification natively and as pointed out by [26], the extension to multi-class classification is not trivial. We find this to be true for many existing MKL formulations. SwMKL relies on a regression method to learn the gating function which characterizes the discriminative capabilities of kernels on local data regions. However, in our case each base kernel classifies training data fairly well, causing a highly imbalanced regression problem. This prevents the Support Vector Regressor from obtaining a meaningful gating function, thereby resulting in a performance similar to that of UNIFORM fusion. The proposed approach achieves the best performance. What is more important is that the architecture provides a reliable way of fusing kernels in a multi-class setting using widely available computational backends for generic neural networks. The architecture is also general so that more advanced techniques can be easily incorporated at certain stages.

## 5. REFERENCES

[1] H. Song, J. J. Thiagarajan, K. N. Ramamurthy, and A. Spanias, “Auto-context modeling using multiple kernel learning,” in *IEEE ICIP*, Phoenix, Sept. 2016, IEEE, pp. 1868–1872.

[2] J. J Thiagarajan, K. N. Ramamurthy, D. Rajan, A. Spanias, A. Puri, and D. Frakes, “Kernel sparse models for automated tumor segmentation,” *International Journal on Artificial Intelligence Tools*, vol. 23, no. 03, pp. 1460004, 2014.

[3] A. Jain, S. VN Vishwanathan, and M. Varma, “Spg-gmkl: generalized multiple kernel learning with a million kernels,” in *18th KDD*. ACM, 2012, pp. 750–758.

[4] J. J Thiagarajan, K. N. Ramamurthy, and A. Spanias, “Multiple kernel sparse representations for supervised and unsupervised learning,” *IEEE Trans. on Image Processing*, vol. 23, no. 7, pp. 2905–2915, 2014.

[5] J. J Thiagarajan, K. N. Ramamurthy, and A. Spanias, “Learning stable multilevel dictionaries for sparse representations,” *IEEE Trans. on Neural Networks and Learning Systems*, vol. 26, no. 9, pp. 1913–1926, 2015.

[6] M. Gönen and E. Alpaydın, “Localized multiple kernel learning,” in *Proceedings of the 25th ICML*. ACM, 2008, pp. 352–359.

[7] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Ng, “Multimodal deep learning,” in *Proceedings of the 28th ICML*, 2011, pp. 689–696.

[8] L. Zhao, Q. Hu, and Y. Zhou, “Heterogeneous features integration via semi-supervised multi-modal deep networks,” in *International Conference on Neural Information Processing*. Springer, 2015, pp. 11–19.

[9] Y. Cho and L. K. Saul, “Kernel methods for deep learning,” in *Advances in neural information processing systems*, 2009, pp. 342–350.

[10] E. V. Strobl and S. Visweswaran, “Deep multiple kernel learning,” in *12th ICMLA*. IEEE, 2013, vol. 1, pp. 414–417.

[11] O. Levy and Y. Goldberg, “Neural word embedding as implicit matrix factorization,” in *Advances in neural information processing systems*, 2014, pp. 2177–2185.

[12] G. Bouma, “Normalized (pointwise) mutual information in collocation extraction,” *Proceedings of GSCL*, pp. 31–40, 2009.

[13] A. Elgammal and C. Lee, “Inferring 3d body pose from silhouettes using activity manifold learning,” in *IEEE Conference on CVPR*. IEEE, 2004, vol. 2, pp. II–681.

[14] T. Mikolov and J. Dean, “Distributed representations of words and phrases and their compositionality,” *Advances in neural information processing systems*, 2013.

[15] E. Bingham and H. Mannila, “Random projection in dimensionality reduction: applications to image and text data,” in *7th KDD*. ACM, 2001, pp. 245–250.

[16] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting.,” *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[17] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *arXiv preprint arXiv:1502.03167*, 2015.

[18] H. Song, J. J. Thiagarajan, K. N. Ramamurthy, A. Spanias, and P. Turaga, “Consensus inference on mobile phone sensors for activity recognition,” in *IEEE ICASSP*, Shanghai, March 2016, IEEE, pp. 2294–2298.

[19] M. Zhang and A. A. Sawchuk, “Human daily activity recognition with sparse representation using wearable sensors,” *IEEE journal of Biomedical and Health Informatics*, vol. 17, no. 3, pp. 553–560, 2013.

[20] J. Frank, S. Mannor, and D. Precup, “Activity and gait recognition with time-delay embeddings.,” in *AAAI*. Citeseer, 2010.

[21] V. Venkataraman and P. Turaga, “Shape descriptions of nonlinear dynamical systems for video-based inference,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. PP, no. 99, pp. 1–1, 2016.

[22] S. Maji, A. C. Berg, and J. Malik, “Classification using intersection kernel support vector machines is efficient,” in *IEEE Conference on CVPR*. IEEE, 2008, pp. 1–8.

[23] M. Zhang and A. A. Sawchuk, “Usc-had: a daily activity dataset for ubiquitous activity recognition using wearable sensors,” in *Proceedings of the ACM Conference on Ubiquitous Computing*. ACM, 2012, pp. 1036–1043.

[24] Z. Sun, N. Ampornpunt, M. Varma, and S. Vishwanathan, “Multiple kernel learning and the smo algorithm,” in *Advances in neural information processing systems*, 2010, pp. 2361–2369.

[25] R. Kannao and P. Guha, “TV commercial detection using success based locally weighted kernel combination,” in *MultiMedia Modeling*. Springer, 2016, pp. 793–805.

[26] A. Zien and C. S. Ong, “Multiclass multiple kernel learning,” in *Proceedings of the 24th ICML*. ACM, 2007, pp. 1191–1198.