A Robust Deep Model for Improved Classification of AD/MCI Patients

Feng Li, Loc Tran, Kim-Han Thung, Shuiwang Ji, Dinggang Shen, and Jiang Li

***Abstract*—Accurate classification of Alzheimer's disease (AD) and its prodromal stage, mild cognitive impairment (MCI), plays a critical role in possibly preventing progression of memory impairment and improving quality of life for AD patients. Among many research tasks, it is of a particular interest to identify noninvasive imaging biomarkers for AD diagnosis. In this paper, we present a robust deep learning system to identify different progression stages of AD patients based on MRI and PET scans. We utilized the dropout technique to improve classical deep learning by preventing its weight coadaptation, which is a typical cause of overfitting in deep learning. In addition, we incorporated stability selection, an adaptive learning factor, and a multitask learning strategy into the deep learning framework. We applied the proposed method to the ADNI dataset, and conducted experiments for AD and MCI conversion diagnosis. Experimental results showed that the dropout technique is very effective in AD diagnosis, improving the classification accuracies by 5.9% on average as compared to the classical deep learning methods.**

***Index Terms*—Alzheimer's disease (AD), deep learning, early diagnosis, magnetic resonance imaging (MRI), positron emission tomography (PET).**

I. Introduction

ALZHEIMER'S disease (AD) is the sixth leading cause of death in the United States [1]. AD patients usually undergo progressive stages of cognitive and memory function impairment, including prodromal, MCI, and AD. For each of these stages, significant amount of research has been conducted aiming to understand the underlying pathological mechanisms. In addition, imaging biomarkers have been identified using different imaging modalities, such as magnetic resonance imaging (MRI) [2], positron emission tomography (PET) [3], and functional MRI [4]. Imaging biomarkers are a set of indicators computed from image modalities and can be used for early detection of AD disease. It has been shown that fusing these different modalities may lead to more effective imaging biomarkers [5].

The first successful deep learning framework autoencoder was developed in 2006 [6]. It was subsequently used in other application fields, and achieved state-of-the-art performance in speech recognition, image classification, and computer vision [7]. Deep learning itself also evolves after 2006. For instance, the multimodal deep learning framework boosted speech classification by learning a shared representation between video and audio modalities [8]. A dropout technique further improved zip code recognition, document classification, and image recognition [9], [10].

In this paper, we developed a robust deep learning framework for AD diagnosis by fusing complementary information from MRI and PET scans. These 3-D scans were preprocessed and their features were further extracted. Specifically, we first applied principal component analysis (PCA) to obtain PCs as new features. We then utilized the stability selection technique [11] together with the least absolute shrinkage and a selection operator (Lasso) method [12] to select the most effective features. The selected features were subsequently processed by the deep learning structure. Model weights in the deep structure were first initialized by unsupervised training, and then, fine-tuned by AD patient labels. During the fine-tuning phase, the dropout technique was employed to improve the model's generalization capability. Finally, the learned feature representation was used for AD/MCI classification by a support vector machine (SVM).

In addition to discrete patient labels (AD, MCI, or healthy), there are two additional clinical scores, namely minimum mental state examination (MMSE) and AD assessment scale-cognitive subscale (ADAS-Cog) associated with each patient. MMSE is a 30-point questionnaire widely used to measure cognitive impairment [13]. It is used to estimate the severity and progression of cognitive impairment, instead of providing any AD information. ADAS-Cog is the most popular cognitive testing instrument to measure the severity of the most important symptoms of AD, including the disturbances of memory, language, praxis, attention, and other cognitive abilities, which have been referred as the core symptoms of AD [14]. The information from these scores is related and identifying the commonality among them may help AD diagnosis. We configured the deep learning structure as a multitask learning (MTL) framework, and treated the learning of class label, MMSE, and ADAS-Cog as related tasks to improve the prediction of main task (class label).

We evaluated the proposed method on the ADNI[1] dataset, and compared it with a baseline method and a similar deep

[1]Available at http://www.loni.ucla.edu/ADNI.

Manuscript received January 6, 2015; revised April 3, 2015; accepted April 30, 2015. Date of publication May 4, 2015; date of current version September 1, 2015. This work was partially supported by NIH grants (EB006733, EB008374, EB009634, MH100217, AG041721, and AG042599). *(Corresponding authors: Dinggang Shen and Jiang Li).*

F. Li, L. Tran, and J. Li are with the Department of Electrical and Computer Engineering, Old Dominion University, Norfolk, VA 23529 USA (e-mail: flixx003@odu.edu; ltran004@gmail.com; jli@odu.edu).

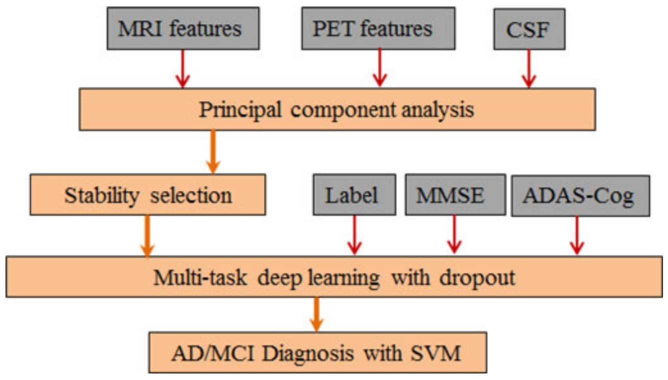
K.-H. Thung is with the Department of Radiology, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599 USA (e-mail: henrythung@gmail.com).

S. Ji is with the Department of Computer Science, Old Dominion University, Norfolk, VA 23529 USA (e-mail: sji@cs.odu.edu).

D. Shen is with the Department of Radiology, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599 USA, and also with the Department of Brain and Cognitive Engineering, Korea University, Seoul 136-701, Korea (e-mail: dinggang_shen@med.unc.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JBHI.2015.2429556



MRI features
PET features
CSF
Principal component analysis
Stability selection
Label
MMSE
ADAS-Cog
Multi-task deep learning with dropout
AD/MCI Diagnosis with SVM

Fig. 1. Diagram of the proposed multitask deep learning framework.

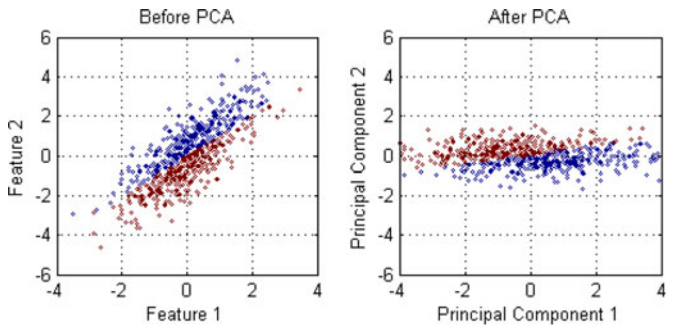


Fig. 2. PCA example. PC 1 contains the most energy of the data but does not have any discrimination information for the "red" and "blue" classes.

learning system, where the autoencoder was used as a feature extractor for AD diagnosis [5]. The baseline method contains feature selection and SVM steps, but does not use deep learning. We also evaluated the impact on performance of each of the components in the proposed system. A brief version of this paper was published in [15].

II. Materials and Methods

The proposed system consists of multiple components, including PCA, stability selection, unsupervised feature learning, multitask deep learning, and SVM training, as shown in Fig. 1. We detail each of these components in the following sections.

A. Data Preprocessing

We utilized the public ADNI dataset to validate our proposed deep learning framework. The dataset consists of MRI, PET, and CSF data from 51 AD patients, 99 MCI patients [43 MCI patients who converted to AD (MCI.C), and 56 MCI patients who did not progress to AD in 18 months (MCI.NC)], as well as 52 healthy normal controls. In addition to the crisp diagnostic result (AD or MCI), this dataset contains two additional clinical scores MMSE and ADAS-Cog for each patient. A typical procedure of image processing was applied to the 3-D MRI and PET images [2], [16], [17] including anterior commissure-posterior commissure correction, skull stripping, cerebellum removal, and spatially normalization. Finally, we extracted 93 region-of-interest-based volumetric features from MRI and PET images, respectively, which together with three CSF biomarkers, i.e., Aβ_{42}, $t-$tau, and p-tau, sum up to 189 features for each subject.

B. Principal Component Analysis

PCA is a linear orthogonal transformation that converts a set of features into linearly uncorrelated variables in which each of the new variable is a linear combination of all original features [18]. The first principal component (PC) is defined as the one that can explain the largest variance in the original dataset. The second PC has the second largest variance under the constraint that it is orthogonal to the first component. If correlations exist among features, the number of PCs that can be found is usually less than the number of features in the original data. PCA is optimal for preserving energy, and it is often used for dimensionality reduction by just keeping the first few PCs.

Let $\mathbf{F}$ denotes a feature dataset with a size of $n \times p$, where n is the number of data samples and p is the number of features in the data, and each column in $\mathbf{F}$ is centered. PCA can be achieved by performing the singular value decomposition on $\mathbf{F}$ as

$$\mathbf{F} = \mathbf{U}\Sigma\mathbf{V}^T \tag{1}$$

where $\mathbf{U}$ is an $n \times n$ matrix with orthogonal unit columns (left singular vectors of $\mathbf{F}$), Σ is an $n \times p$ diagonal matrix consisting of singular values of $\mathbf{F}$ from the largest to the least, and $\mathbf{V}$ is an $p \times p$ matrix whose columns are orthogonal unit vectors (right singular vectors of $\mathbf{F}$).

To achieve dimensionality reduction, the first l columns in $\mathbf{V}$ corresponding to the first l largest singular values of $\mathbf{F}$ can be used as a transformation matrix to be applied on $\mathbf{F}$

$$\mathbf{x} = \mathbf{F}\mathbf{V}_l \tag{2}$$

where $\mathbf{V}_l$ consists of the first l columns of $\mathbf{V}$.

Geometrically, PCA analysis rotates data to align its maximum variance direction of the data with the coordinate system as illustrated in Fig. 2. PCA is an effective tool for dimensionality reduction but the preserved PCs may not be useful for classification. The 2-D artificial dataset in Fig. 2 consists of "blue" and "red" classes. After PCA, the whole dataset was rotated, and its main axis was aligned with the coordinate system. However, even though PC 1 has the largest variance, it does not contain any discriminating information for the two classes. For the purpose of classification, PC 2 is preferred, and a feature selection step is necessary. This example shows that the feature selection may be applied after PCA to retain discriminating information for classification.

C. Stability Selection

In this paper, we first applied PCA to the 189 features, and used the resulting PCs as new features. We then applied Lasso [12] to identify the most effective features for AD diagnosis. Lasso tries to minimize the following cost function for feature selection

$$\min_{\mathbf{s}} ||\mathbf{t} - \mathbf{x}\mathbf{s}||_2^2 + \lambda||\mathbf{s}||_1 \tag{3}$$

where $\mathbf{t} \in \{+1, -1\}^n$ is a class label vector of size $n \times 1$ associated with the feature matrix $\mathbf{x}$ of size $n \times l$, where l is the number of features (PCs) found in PCA, $\mathbf{s} = [s_1, s_2, \ldots, s_l]^T$ is the weight vector associated with the l features (columns in $\mathbf{x}$), λ is a regularization parameter, and $||\cdot||_2$ and $||\cdot||_1$ denote L_2 and L_1 norms, respectively. Because of the L_1 norm constraint on the weight magnitude, the solution minimizing the above cost function is usually sparse, meaning that if a feature is not correlated with the target class label, the feature will have a zero value for its weight. Features having nonzero weights will be selected and otherwise will be excluded.

It is well known that the solution of L_1 norm-based optimizations are sensitive to the choice of λ, and it is difficult to determine how many features should be kept in the model. A recent breakthrough sheds a light on selecting the right amount of regularization for stability selection [11]. The idea is to repeat the feature selection procedure multiple times based on bootstrapped datasets, and compute the probability of the features to be selected. The final selected features are those having probabilities above a predefined threshold t_h. It has been shown experimentally and theoretically that the feature selection results vary little for sensible choices in a range of the cutoff value for t_h [11]. We incorporated the stability selection concept into the AD patient diagnosis in this paper. In particular, we repeated the Lasso procedure 50 times, and each time with a different value for the parameter λ (we used the SLEP toolbox for Lasso[2]). A probability p_i, for the ith feature was computed by counting the frequency of the feature being selected in the 50 experiments. The ith feature was selected if p_i is larger than a predefined threshold t_h.

D. Multitask Deep Learning With Dropout

In contrast to a traditionary three-layer neural network (shallow structure), deep learning is based on a deep architecture consisting of many layers of hidden neurons for modeling. A shallow architecture would involve many duplications of effort to express things, and such a fat architecture has been shown to suffer from the problem of overfitting, which leads to a poor generalization capability. Instead, a deep architecture could more gracefully reuse previous computations, and discover complicated relations of input [19].

To train a deep architecture, the standard Backpropagation (BP) algorithm did not work well with randomly initialized weights because the error feedback becomes progressively noisier as it goes back to lower levels (closer to inputs) making the low-level weight updates less effective. Even though experiments have shown that if top layers have enough units, the deep structure can still bring down training errors small enough, it cannot generalize well to new data [20]. This is because the top layers can be effectively trained by gradient based algorithms but low levels cannot. The randomly initialized low-level layers behave like random feature detectors so good representations for original data were not achieved leading to degraded generalization capability [20]. In 2006, a breakthrough in deep learning

[2]Available at http://www.public.asu.edu/ jye02/Software/SLEP/index.htm.

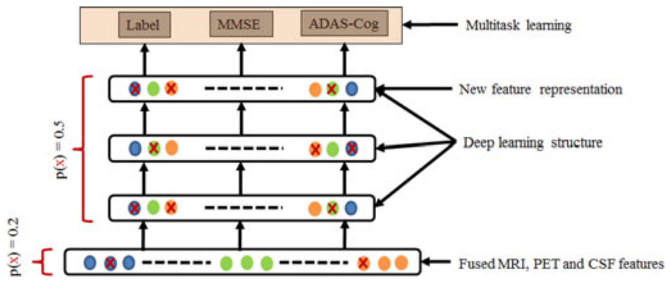


Fig. 3. Multitask deep learning with dropout. "x" denotes a dropped unit.

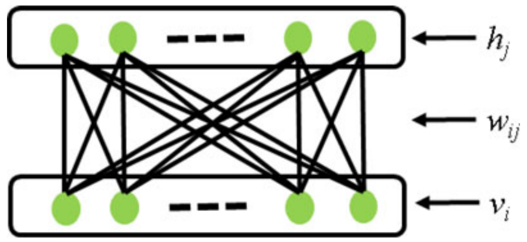


Fig. 4. Basic RBM model.

has made the deep architecture training possible by utilizing the restricted Boltzmann machine (RBM) to initialize multiple hidden layers one layer at a time in an unsupervised manner [6]. With the unsupervised learning, deep learning tries to understand data first, i.e., to obtain a task specific representation from data so that a better classification can be achieved. It has experimentally proven that the unsupervised learning step plays a critical role in the success of deep learning [7]. The proposed deep model shown in Fig. 3 consists of several components that will be described below.

1) Pretraining With RBM: Each layer in the proposed deep model is an RBM, and the deep model used in this paper consists of a stack of RBMs. RBM is an energy-based model, in which a scalar energy is associated with each configuration of the variables in the model, and a probability distribution function (PDF) through the energy function is defined. The purpose of learning is to modify the energy function so that a desirable PDF can be achieved, i.e., to have low energy. A basic RBM model having a visible (input) layer and a hidden (output) layer is shown in Fig. 4. The visible layer of the bottom RBM contains real-valued units (receiving data) and all other RBM layers have binary units. Let $v \in R^M$ represents input data (visible units), and $h \in 0, 1^N$ denotes binary hidden units for the bottom RBM, we used Gaussian–Bernoulli RBMs to train it [20], [21]. All other RBMs were trained by utilizing Bernoulli–Bernoulli distribution. Variables v and h have a joint probability distribution defined as

$$p(v, h) = \frac{1}{Z} exp^{-E(v,h)} \quad (4)$$

where $E(v, h)$ is an energy function and Z is a normalization constant. For real-valued visible layer RBMs, $E(v, h)$ is defined as

$$E(v,h) = \frac{1}{2\sigma^2}\sum_i v_i^2 - \frac{1}{\sigma^2}\left(\sum_i c_i v_i + \sum_j b_j h_j + \sum_{i,j} v_i w_{ij} h_j\right) \quad (5)$$

where c_i and b_j are biases of the ith and jth units in the visible and hidden layers, respectively. w_{ij} is the weight connecting v_i and h_j, and σ^2 is the variance of v. The conditional probability distributions are

$$P(h_j = 1|v) = \text{sigmoid}\left(\frac{1}{\sigma^2}\left(\sum_i w_{ij} v_i + b_j\right)\right) \quad (6)$$

$$P(v_i|h) = \mathcal{N}\left(\sum_j w_{ij} h_j + c_i, \sigma^2\right). \quad (7)$$

If both visible and hidden layers are binary, the energy function and conditional probability distributions are defined as

$$E(v,h) = -\left(\sum_i c_i v_i + \sum_j b_j h_j + \sum_{ij} v_i w_{ij} h_j\right) \quad (8)$$

$$P(h_j = 1|v) = \text{sigmoid}\left(\sum_i w_{ij} v_i + b_j\right) \quad (9)$$

$$P(v_i = 1|h) = \text{sigmoid}\left(\sum_j w_{ij} h_j + c_i\right). \quad (10)$$

Model parameters w, b, and c are updated using contrastive divergence [22]. For RBM having a real-valued visible layer, the formulas for updating those parameters during each iteration are

$$\Delta w_{ij}^{t+1} = \eta \Delta w_{ij}^{t} - \epsilon\left(< \frac{1}{\sigma^2} v_i h_j >_d - < \frac{1}{\sigma^2} v_i h_j >_m\right) \quad (11)$$

$$\Delta b_i^{t+1} = \eta \Delta b_i^{t} - \epsilon\left(< \frac{1}{\sigma^2} v_i >_d - < \frac{1}{\sigma^2} v_i >_m\right) \quad (12)$$

$$\Delta c_j^{t+1} = \eta \Delta c_j^{t} - \epsilon\left(< h_j >_d - < h_j >_m\right) \quad (13)$$

where $< \cdot >_d$ and $< \cdot >_m$ denote the expectation computed over data and model distributions accordingly, t is the iteration index, η is the momentum, and ϵ is the learning rate. For binary RBM, (11) and (12) become

$$\Delta W_{ij}^{t+1} = \eta \Delta W_{ij}^{t} - \epsilon\left(< v_i h_j >_d - < v_i h_j >_m\right) \quad (14)$$

$$\Delta b_i^{t+1} = \eta \Delta b_i^{t} - \epsilon\left(< v_i >_d - < v_i >_m\right). \quad (15)$$

Note that the pretraining of an RBM is unsupervised, i.e., class label (classification task) or desired output (regression) is not needed in the training. After the pretraining, we attached the class label on top of the stacked RBMs, and utilized an adaptive BP algorithm to fine tune the weights in the model. All binary layers were also converted to real-valued units by using their continuous activities. Thus, the deep learning model turned to be a traditional multilayer perceptron but its weights were initialized by RBM.

2) Multitask Learning: In MTL, related tasks are learned simultaneously by extracting and utilizing appropriate shared information across tasks to improve performance. It has received attention in broad areas recently, such as machine learning, data mining, computer vision, and bioinformatics [23]–[25]. This approach is particularly effective when only limited training data for each task is available. It is worth noting that neural networks can simultaneously model multiple outputs making deep learning a natural MTL framework if multiple tasks share inputs [29]. The proposed multitask deep learning framework is shown in Fig. 3, where we treated the predictions of class label, MMSE, and ADAS-Cog as three different tasks and modeled them simultaneously. MMSE and ADAS-Cog were normalized to the range of [0, 1], and we used the deep structure as a regression model. The class label was coded by the 1-of-k scheme. To classify an input vector, we checked the corresponding k outputs, and assign it to the class having the largest output. One drawback of deep model is overfitting due to large capacity. This is more prominent if training data is limited. To overcome this limitation, we utilized the dropout technique to improve training.

3) Dropout With Adaptive Adaptation: Deep learning achieved excellent results in applications, where training data size is large. For small sized datasets such as the one in this paper, it is still possible for a deep structure to overfit the data given the fact that it usually has tens of thousands or even millions of parameters. To improve the generalization capability of the model, the dropout technique tries to prevent weight coadaptation by randomly dropping out some units in the model during training [9], [10]. We incorporated the dropout technique in the MTL context to improve AD diagnosis as shown in Fig. 3. In the training process, each hidden unit in the model was dropped with a probability of 0.5 when a batch of training cases were present. Previous experiments [9] showed that it is also beneficial if we apply the "dropout" process to the input layer but with a lower probability (i.e., 0.2 in this paper). In the testing procedure, all hidden units and inputs were used to compute model outputs for a testing case with appropriate compensations, i.e., weights between inputs and the first hidden layer were scaled by 0.8 and all other weights were halved.

During the multitask fine-tuning step, the stochastic gradient descent method with a fixed learning factor is usually utilized as [6]

$$w_{ij} = w_{ij} + \Delta w_{ij} = w_{ij} - \alpha \frac{\partial L}{\partial w_{ij}} \quad (16)$$

where $\frac{\partial L}{\partial w_{ij}}$ is the gradient of the cost function L and α is a learning factor. Sometimes, the weights update may contain a momentum term [9]. We proposed an adaptive learning factor to speed up the adaptation. The motivation of the adaptive learning is that the learning factor should be large at locations, where gradient is small and vice versa. Assume the decrease of L due to the change in w_{ij} is approximated by

$$\Delta L^{ij} = L^{ij}_{\text{new}} - L^{ij}_{\text{old}} \approx \frac{\partial L}{\partial w_{ij}} \times \Delta w_{ij} = -\alpha \left[\frac{\partial L}{\partial w_{ij}}\right]^2 \tag{17}$$

then ΔL due to all w_{ij} can be computed as

$$\Delta L = -\alpha \sum_i \sum_j \left[\frac{\partial L}{\partial w_{ij}}\right]^2. \tag{18}$$

Suppose, we want to decrease L by $\beta\%$, then $L_{\text{new}} = (1 - \beta) L_{\text{old}}$, and an adaptive learning factor α can be determined as

$$\alpha = \frac{\beta L_{\text{old}}}{\sum_i \sum_j \left[\frac{\partial L}{\partial w_{ij}}\right]^2}. \tag{19}$$

We set β as 10% in our experiments in this paper. Once the new feature representation is learned, an SVM classifier [26] was trained using the learned feature representation.

E. SVM Classifier

Given a set of data pairs $\{\mathbf{r}_i, t_i\}_{i=1}^n$, where $\mathbf{r}_i \in R^M$ is the learned feature representation from subjects, $t_i \in \{+1, -1\}$ is a class label (e.g., AD versus non-AD) associated with $\mathbf{r}_i$. An SVM defines a hyperplane

$$f(\mathbf{r}) = \mathbf{k}^T \phi(\mathbf{r}) + e = 0 \tag{20}$$

separating the data points into two classes. In (20), $\mathbf{k}$ and e are the hyperplane parameters, and $\phi(\mathbf{r})$ is a function mapping, the vector $\mathbf{r}$ to a higher dimensional space. The hyperplane (20) is determined using the concept of *structural risk minimization* [26] by solving the following optimization problem:

$$\min_{\mathbf{k}, e, \xi} \left(\frac{1}{2}\mathbf{k}^T \mathbf{k} + C \sum_{i=1}^n \xi_i\right) \tag{21}$$

subject to

$$t_i \left(\mathbf{k}^T \phi(\mathbf{r}_i) + e\right) \geq 1 - \xi_i, \xi_i \geq 0 \tag{22}$$

where C is a regularization parameter and ξ_i is a slack variable. After the hyperplane is determined, an AD case is declared if $f(\mathbf{r}_i) > 0$, or otherwise a non-AD case is declared.

III. Results and Discussions

A. Experimental Setup

1) Tenfold Cross Validation (CV): We consider four classification tasks including AD patients versus healthy control subjects (AD versus HC), MCI patients versus HC (MCI versus HC), AD patients versus MCI patients (AD versus MCI), and MCI-converted versus MCI-non converted (MCI.C versus MCI.NC). For each task, we utilized a tenfold CV scheme to evaluate the proposed method. In the tenfold CV, we randomly divided the dataset into ten parts and for one run, we separated one part for testing and applied the proposed framework to the remaining data to train a classification model. This procedure was repeated ten times so that each part was tested once. Finally, testing accuracies were computed. To obtain a more reliable estimate of the performance, we repeated the tenfold CV ten times for each task with different random data partitions and computed average accuracy. To compare different classification models, we kept the same data partitions in the tenfold CV, and utilized the paired t-test to evaluate if there is a significant performance difference.

2) Hyperparameter Determination: We did preliminary experiments to determine the structure of the deep learning model. It was found that using three hidden layers with hidden units of 100-50-20 worked the best among the candidate structures considered, and was, thus, utilized in our experiments. For the SVM classifier, we tried different kernels, and a linear kernel was chosen. We also did a grid search for the "soft margin" parameter in the linear kernel SVM model but it did not improve the classification accuracies. Therefore, in all experiments, we utilized a three hidden-layer model with a structure of 100-50-20 for feature learning, and a linear SVM with default soft margin as the classifier.

3) Impact Assessment for Individual Component: There are four components in the proposed framework including PCA, stability selection, dropout, and MTL. Inspired by "sensitivity analysis" and "impact assessment" that analyze inputs of or components in a model and identify their impacts on the model objectives by varying the inputs [27]. We incorporated a similar concept to evaluate the impact of each component on model performance by varying the component (presence versus absence). "Absence" means that the component was not included in the model.

4) Methods for Comparison: We compared the proposed method with a baseline method, and a similar deep learning system proposed in [5]. The baseline method consists of all components in the proposed system except the deep learning step. The work by Suk and Shen in [5] is an autoencoder-based deep learning method in which feature representations for MRI, PET, and CSF from the same dataset were learned separately and combined by a linear SVM classifier. They also combined the learned representations with original features for AD diagnosis.

B. Results

Table I shows the overall performances of the proposed method, and the impact of each component in the framework. The proposed method performed the best in diagnosing AD and MCI patients, and discriminating MCI patients from AD patients with accuracies of 91.4%, 77.4%, and 70.1%, respectively. It is significantly better than the baseline method that obtained accuracies of 86.4%, 72.1%, and 61.5% for the diagnoses. In the MCI conversion diagnosis (MCI.C versus MCI.NC), the PCA component slightly degraded the proposed method (from 58.1% to 57.4%) but it is still significantly better than the baseline method (57.4% versus 50.6%).

Among those components, it is obvious that "dropout" has the most significant impact on the performances. Without "dropout," deep learning did not significantly improve the baseline method (68.2% versus 67.7% in terms of average acc.). The least important component is "PCA," i.e., the average acc.

TABLE I
PERFORMANCE COMPARISON (IN%) OF THE COMPETING METHODS

Tasks	Proposed	-PCA	-Dropout	-SS	-Multitask	Baseline
AD versus HC	**91.4**(1.8)	89.6(1.3)	84.2(3.0)	89.4(1.6)	90.3(1.7)	86.4(2.0)
MCI versus HC	**77.4**(1.7)	76.4(1.5)	73.1(3.1)	74.3(1.6)	75.6(1.7)	72.1(3.0)
AD versus MCI	**70.1**(2.3)	69.5(2.7)	65.1(3.7)	68.7(2.1)	67.1(2.9)	61.5(2.9)
MCI.C versus MCI.NC	57.4(3.6)	**58.1**(1.8)	50.2(3.3)	57.7(1.8)	56.7(3.0)	50.6(4.7)
Average	**74.1**	73.4	68.2	72.5	72.4	67.7

The proposed method consists of four components. "-PCA" stands for "the proposed method without the PCA component" and "SS" stands for stability selection, "baseline" denotes the framework without the deep learning component.

TABLE II
PAIRED t-TEST BETWEEN RESULTS OF THE PROPOSED METHOD VERSUS DEEP LEARNING WITHOUT DROPOUT

Tasks	Proposed	-Dropout	Improvement	p-value	SAEF	LLF+SAEF
AD versus HC	**91.4**(1.8)	84.2(3.0)	7.2	$< 10^{-3}$	83.2(2.7)	85.3(3.2)
MCI versus HC	**77.4**(1.7)	73.1(3.1)	4.3	0.0034	70.1(2.8)	76.9(2.3)
AD versus MCI	**70.1**(2.3)	65.1(3.7)	5.0	0.0017	N/A	N/A
MCI.C versus MCI.NC	57.4(3.6)	50.2(3.3)	7.2	$< 10^{-3}$	58.4(4.1)	**60.3**(2.3)
Overal Average	**74.1**	68.2	5.9	N/A	N/A	N/A
Average w/o AD versus MCI	**75.4**	69.2	6.2	N/A	70.6	74.2

The methods of "SAEF" and "LLF+SAEF" were proposed by Suk and Shen [5]. "SAEF" stands for stacked autoencoder features and "LLF" denotes low-level features.

slightly dropped from 74.1% to 73.4% without the PCA component. Without "stability selection" and "multitask learning," the average accuracy dropped from 74.1% to 72.5% and 72.4%, respectively.

We conducted a paired t-test between results by the proposed method and those from classical deep learning ("-Dropout"). Table II lists the improvements and p-values. The average improvement is 5.9%, and the improvements for all the four classification tasks are significant.

The work by Suk and Shen [5] on the same dataset is also shown in Table II, where "SAEF" corresponds to the method using features learned by a deep autoencoder, and "LLF+SAEF" represents the method that combines original features with the SAEF features for AD diagnosis. The AD versus MCI classification experiment was not conducted in [5]. The proposed method (75.4%) outperformed the SAEF method (with an average accuracy of 70.6%). By combining SAEF with LLF (LLF+SAEF), the average accuracy was increased to 74.2% (see last column in Table II).

C. Discussions

There are usually two ways to increase the generalization capability of a model, adding regularization (L_1 or L_2 norm) on weights or using a committee machine. However, solving the regularization problem is usually challenging especially in the deep learning context. In addition, the committee machine technique requires averaging many separately trained models to compute a prediction for a testing case, which is time consuming for deep learning. The dropout procedure does the both (constraint and committee machine) simultaneously in a very efficient way. 1) Each submodel in training is a sampled model from all possible ones and all submodels share weights. The weight sharing property is equivalent to the L_1 or L_2 norm constraint on weights, and 2) the testing procedure is an approximation of averaging all trained submodels for a testing case but it does not separately store them because they share weights. This is an extremely efficient and smart implementation of a committee machine [9], [10].

The impact evaluation method was inspired by the "sensitivity analysis" and "impact assessment" [27]. We were aiming to identify the impact on performance of each component in the model by excluding the component from the pipeline. Note that we did not try to decouple the component from the system. This evaluation method may not be a strict sensitivity analysis or impact assessment by means of their definitions, but we can verify each component if it can improve the AD diagnosis when it is included in the proposed system. Our experiments showed that the dropout component has the largest impact on the performance, MTL ranked the second, stability selection the third, and PCA has the least impact on the performance.

In terms of stability selection and computational efficiency, there were usually around 40 features left after the stability selection, and it took about 1 h for a personal computer to conduct a tenfold CV evaluation for one task. The number of features that were chosen was determined by stability selection, in which the Lasso algorithm ran 50 times with different values of regularization parameter (λ). In each run, Lasso chose different features, and a probability of being chosen for each feature was computed in the 50 runs. Finally, a feature was chosen if its probability is larger than 0.5.

It is worth to note that the results by the proposed method in Tables I and II only used the new representations learned by the deep model. We tried to combine the new representations with the original features but the combination did not improve the performance. In [5], new representations learned from autoencoder did not perform well unless they were combined with the original features. Our experiment also showed that the deep model without dropout just performed comparably as the baseline method. It seems that the traditional deep learning cannot extract information effectively from small datasets unless it is regularized by techniques such as dropout.

In [28], a multikernel SVM (MK-SVM) method was applied to the same dataset to combine the original LLF features for AD diagnosis, and achieved 93.2% and 76.4% for AD versus HC and MCI versus HC classifications, respectively. The MCI conversion diagnosis and AD versus MCI classification were not conducted. In [5], utilizing the MK-SVM method to combine SAEF features from MRI, PET, and CSF boosted the performances to 95.9%, 85.0%, and 75.8% for the three tasks (AD versus MCI classification was not performed), respectively. Since the dropout technique improved upon the basic deep learning, we are currently investigating if the MK-SVM method can further boost the performance of the proposed system.

We did not attempt to perform a comprehensive comparison study of the proposed method with others that have been applied to this dataset in the literature. Instead, we have evaluated some recently proposed advanced machine learning techniques for AD diagnosis including Lasso, stability selection, MTL, deep learning, and dropout. The dropout technique seems to be an effective method of regularization for learning with small datasets. Without dropout, deep learning has no advantage over the baseline method on ANDI dataset (68.2% versus 67.7%). Note that dropout is computationally very efficient as compared to either L_1 norm-based regularization or committee machine, and it can be extended to many models other than the deep model as discussed in this paper.

IV. Conclusion

Our proposed method achieved 91.4%, 77.4%, 70.1%, and 57.4% accuracies for AD versus HC, MCI versus HC, AD versus MCI, and MCI.C versus MCI.NC classifications, respectively. The framework consists of multiple components including PCA, stability selection, dropout, and multitask deep learning. We showed that dropout is the most effective one. This is not surprising because the size of ADNI data is relatively small compared to that of the deep structure utilized in this paper. Classical deep learning does not perform well on this small dataset, but with the dropout technique, the average accuracy was improved by 5.9% on average. We plan to incorporate MK-SVM [5] into our method for further improving AD diagnosis.

References

[1] Alzheimer's Association: 2012, "Alzheimer's disease facts and figures," *Alzheimer's Dementia*, vol. 8, no. 2, pp. 131–168, 2012.

[2] C. Davatzikos, P. Bhatt, L. M. Shaw, K. N. Batmanghelich, and J. Q. Trojanowski, "Prediction of MCI to AD conversion, via MRI, CSF biomarkers, and pattern classification," *Neurobiol. Aging*, vol. 32, no. 12, pp. 2322.e19–2322.e27, 2011.

[3] A. Nordberg, J. O. Rinne, A. Kadir, and B. Langstrom, "The use of PET in Alzheimer disease," *Nature Rev. Neurol.*, vol. 6, no. 2, pp. 78–87, 2010.

[4] M. D. Greicius, G. Srivastava, A. L. Reiss, and V. Menon, "Default-mode network activity distinguishes Alzheimer's disease from healthy aging: Evidence from functional MRI," in *Proc. Nat. Acad. Sci. United States Amer.*, vol. 101, no. 13, pp. 4637–4642, 2004.

[5] H. Suk and D. Shen, "Deep learning-based feature representation for AD/MCI classification," in *Proc.Med. Image Comput. Comput.-Assisted Intervention Conf.*, 2013, pp. 583–590.

[6] G. E. Hinton, S. G. Osindero, and Y. W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[7] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[8] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Ng, "Multimodal deep learning," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 689–696.

[9] G. E. Hinton, N. Srivastave, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," arXiv:1207.0580, 2012.

[10] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, 2014.

[11] N. Meinshausen and P. Buhlmann, "Stability selection," *J. Roy. Statist. Soc. B*, vol. 72, pp. 417–473, 2010.

[12] R. Tibshirani, "Regression shrinkage and selection via the Lasso," *J. Roy. Statist. Soc., Ser. B*, vol. 58, no. 1, pp. 267–288, 1996.

[13] V. C. Pangman, J. Sloan, and L. Guse, "An examination of psychometric properties of the mini-Mental state examination and the standardized mini-Mental state examination: Implications for clinical practice," *Appl. Nursing Res.*, vol. 13, no. 4, pp. 209–213, 2000.

[14] E. Kolibas, V. Korinkova, V. Novotny, K. Vajdickova, and D. Hunakova, "ADAS-cog (Alzheimer's disease assessment scale-cognitive subscale)—validation of the Slovak version," *Bratisl Lek Listy*, vol. 101, no. 11, pp. 598–602, 2000.

[15] F. Li, L. Tran, KH. Thung, S. Ji, D. Shen, and J. Li, "Robust deep learning for improved classification of AD/MCI patients," *Mach. Learn. Med. Imag.*, vol. 8679, pp. 240–247, 2014.

[16] N. Kabani, D. MacDonald, C. Holmes, and A. Evans, "A 3D atlas of the human brain," *NeuroImage*, vol. 7, no. 4, p. S717, 1998.

[17] C. Hinrichs, V. Singh, G. Xu, and S. C. Johnson," Predictive markers for AD in a multi-modality framework: An analysis of MCI progression in the ADNI population," *NeuroImage*, vol. 55, no. 2, pp. 574–589, 2011.

[18] I. T. Jolliffe, *Principal Component Analysis* (Springer Series in Statistics), 2nd ed. New York, NY, USA: Springer, 2002.

[19] 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, 2010.

[20] Y. Bengio, "Learning deep architectures for AI," *Found. Trends Mach. Learning*, vol. 2, no. 1, p. 1–127, 2009.

[21] K. Cho, A. Ilin, and T. Raiko, "Improved learning of Gaussian-Bernoulli restricted Boltzmann machines," in *Proc. Artif. Neural Netw. Mach. Learn. Conf.*, 2011, pp. 10–17.

[22] G. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 1527–1554, 2006.

[23] B. Heisele, T. Serre, M. Pontil, T. Vetter, and T. Poggio, "Categorization by learning and combining object parts," in *Proc. Neural Inf. Process. Syst.*, 2001, pp. 1239–1245.

[24] S. Ji and J. Ye, "An accelerated gradient method for trace norm minimization," in *Proc. 26th Ann. Int. Conf. Mach. Learn.*, 2009, pp. 457–464.

[25] Y. Xue, X. Liao, L. Carin, and B. Krishnapuram, "Multi-task learning for classification with Dirichlet process priors," *J. Mach. Learn. Res.*, vol. 8, pp. 35–63, 2007.

[26] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[27] A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli,. M. Saisana, and S. Tarantola, *Global Sensitivity Analysis. The Primer*. New York, NY, USA: Wiley, 2008.

[28] D. Zhang, Y. Wang, L. Zhou, H. Yuan, and D. Shen, "Multimodal classification of Alzheimers disease and mild cognitive impairment," *NeuroImage*, vol. 55, no. 3, pp. 856–867, 2011.

[29] R. Caruana, "Multitask learning: A knowledge-based source of inductive bias," *Mach. Learn.*, vol. 28, pp. 41–75, 1997.

Authors' photographs and biographies not available at the time of publication.