J Digit Imaging (2017) 30:615–621
DOI 10.1007/s10278-016-9910-0

# Machine Learning Interface for Medical Image Analysis

**Yi C. Zhang**[1] · **Alexander C. Kagen**[2]

Published online: 11 October 2016


**Abstract** TensorFlow is a second-generation open-source machine learning software library with a built-in framework for implementing neural networks in wide variety of perceptual tasks. Although TensorFlow usage is well established with computer vision datasets, the TensorFlow interface with DICOM formats for medical imaging remains to be established. Our goal is to extend the TensorFlow API to accept raw DICOM images as input; 1513 DaTscan DICOM images were obtained from the Parkinson's Progression Markers Initiative (PPMI) database. DICOM pixel intensities were extracted and shaped into tensors, or *n*-dimensional arrays, to populate the training, validation, and test input datasets for machine learning. A simple neural network was constructed in TensorFlow to classify images into normal or Parkinson's disease groups. Training was executed over 1000 iterations for each cross-validation set. The gradient descent optimization and Adagrad optimization algorithms were used to minimize cross-entropy between the predicted and ground-truth labels. Cross-validation was performed ten times to produce a mean accuracy of $0.938 \pm 0.047$ (95 % CI 0.908–0.967). The mean sensitivity was $0.974 \pm 0.043$ (95 % CI 0.947–1.00) and mean specificity was $0.822 \pm 0.207$ (95 % CI 0.694–0.950). We extended the TensorFlow API to enable DICOM compatibility in the context of DaTscan image analysis. We implemented a neural network classifier that produces diagnostic accuracies on par with excellent results from previous machine learning models. These results indicate the potential role of TensorFlow as a useful adjunct diagnostic tool in the clinical setting.

**Keywords** Artificial intelligence · Computer vision · Classification · Image analysis

✉ Yi C. Zhang
yi.zhang1@mssm.edu

Alexander C. Kagen
akagen@chpnet.org

[1] Translational and Molecular Imaging Institute, Icahn School of Medicine at Mount Sinai, 1470 Madison Avenue, 1st Floor, New York, NY 10029, USA

[2] Department of Radiology, Mount Sinai West, 1000 10th Ave Rm 4C-12, New York, NY 10019, USA

## Background

Machine learning approaches are increasingly used to perform automated image analysis with the goal of improving the accuracy of visual interpretation. In the imaging evaluation of Parkinson's disease, the European Association of Nuclear Medicine Neuroimaging Committee guidelines recommend concomitant visual and quantitative assessments of striatal signal changes on $^{123}$I-ioflupane (DaTscan) SPECT studies [1].

Computational models such as support vector machines [2–4], multivariate logistic regression [5], and artificial neural networks [6] have correctly classified patients with Parkinson's disease with high accuracies above 90 %. Although machine learning models show excellent potential as diagnostic aids in clinical practice, access to the technology is limited due to institution-specific, localized software implementations.

Google™ TensorFlow is a widely available second-generation machine learning software library for implementing and executing large-scale artificial neural networks [7]. Released under the Apache 2.0 license in November 2015, TensorFlow offers open-source access with documentation for general public use. TensorFlow has a

potential to advance radiological image analysis in a similar way that other Google algorithms have made progress in image classification and object detection [8–10] and in speech recognition [11–13].

TensorFlow features artificial neural networks that train on labeled input data for classification tasks with the goal of correctly assigning an unknown into a predefined class through iterative updates of weights and biases associated with each input [7]. Training is a search process for sets of weights and biases that minimize the squared errors between the predicted output and ground-truth data.

The TensorFlow application program interface (API) allows researchers to train and test neural networks by constructing computational graphs using one of the supported front-end languages such as Python or C++ [7]. Input image datasets are in the form of tensors, or *n*-dimensional arrays with prespecified element types, that flow between nodes during each training iteration. TensorBoard is a visualization tool for the TensorFlow computational graph where execution sequences and performance metrics are displayed via a graphical interface [14].

TensorFlow is commonly used for classification tasks with established computer vision input datasets [15]. However, the DICOM file format for medical imaging is not yet supported in the TensorFlow API. The purpose of this study is to describe a methodological innovation for adapting the TensorFlow API to raw DICOM input in the context of $^{123}$I-ioflupane (DaTscan) classification.

## Methods

### DaTscan Image Collection

The Institutional Review Board of Mount Sinai Hospital System deemed that this study does not require IRB approval or IRB exemption. Data used in the preparation of this article were obtained from the Parkinson's Progression Markers Initiative (PPMI) database (www.ppmi-info.org/data). For up-to-date information on the study, visit www.ppmi-info.org. PPMI is an observational clinical study that collects imaging, biologic sampling, clinical, and behavioral assessments to characterize Parkinson's disease progression. Further details about inclusion and exclusion criteria are available at http://www.ppmi-info.org/wp-content/uploads/2014/01/PPMI-AM7-Protocol.pdf [16].

The dataset retrieved on February 16, 2016 contained 1513 preprocessed $^{123}$I-ioflupane SPECT images. There was no statistically significant difference in age and gender between healthy controls and Parkinson's disease patients in the PPMI database. Initial and follow-up $^{123}$I-ioflupane SPECT images were included in the subsequent analysis.

### DaTscan SPECT Image Processing

$^{123}$I-ioflupane SPECT imaging was acquired at PPMI imaging centers according to PPMI protocol [16, 17]. Briefly, SPECT raw projection data at all imaging centers was imported to a HERMES (Hermes Medical Solutions, Skeppsbron 44, 111 30 Stockholm, Sweden) system for iterative (HOSEM) reconstruction. The HOSEM-reconstructed files were then transferred to the PMOD (PMOD Technologies, Zurich, Switzerland) for subsequent processing. Attenuation correction ellipses were drawn on the images, and a Chang 0 attenuation correction was applied to images utilizing a site-specific mu that was empirically derived from phantom data acquired during site initiation for the trial. A standard Gaussian 3D 6.0-mm filter was applied. PPMI database protocol [17] specified that attenuation-corrected SPECT studies were normalized to the Montreal Neurologic Institute (MNI) space, a standard anatomical coordinate system, such that all scans were in the same anatomic alignment for image registration. Each DaTscan SPECT contains 91 slices, and each slice contains 91 × 109 pixels encoded with 16-bit greyscale.

### TensorFlow Input Dataset Assembly

TensorFlow accepts sets of images and corresponding textual labels as input data. A total of 1513 DaTscan SPECT imaging volumes were included in this study, of which 1171 studies were from the Parkinson's disease group, 211 studies from the control group, and 131 studies from the Scans without Evidence of a Dopaminergic Deficit (SWEDD) group. For each DaTscan study, only a single axial image out of the imaging volume can be selected for analysis because TensorFlow currently supports input in the form of 2D images and not 3D volumetric data. Two radiologists (board-certified attending and board-eligible fellow) reviewed DaTscan axial images and selected slice 41, which contains the highest striatal signal-to-background ratio by visual inspection, as input data for TensorFlow.

The 1513 axial images through the basal ganglia were randomized and split into three parts: 1189 images for the training set, 108 images for the validation set, and 216 images for the test set (Fig. 1). The 11:1:2 ratio of training, validation, and test sample sizes was modeled after the TensorFlow tutorial dataset construction [15]. In addition to analyzing axial slice 41, two more datasets containing 1189 training, 108 validation, and 216 test images were constructed using slice 40 and slice 42 in order to include more imaging volume and capture possible additional discriminating information.

The training set containing images and corresponding labels are used to train the machine learning model. The validation set, which is not seen in the training phase, is used to iteratively assess training accuracy and prevent overfitting, which manifests as increasing errors in prediction. The

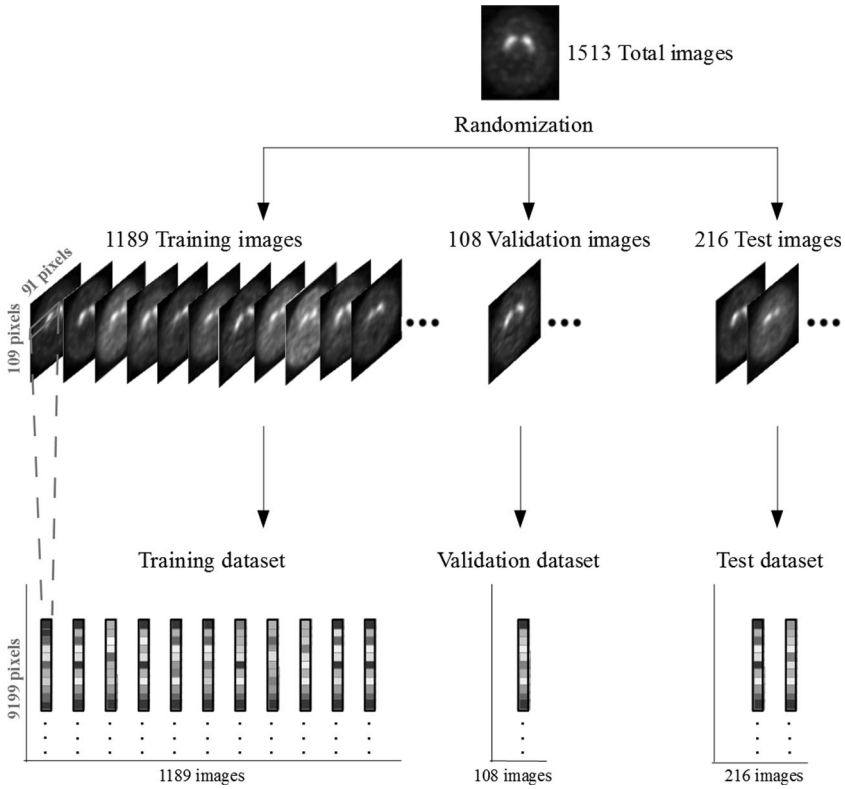


**Fig. 1** Input dataset assembly. Block diagram illustrates a single cross-validation dataset where 1513 individual axial DaTscan images through the basal ganglia are randomized into a training dataset of 1189 images, validation dataset of 108 images, and test dataset of 216 images

validation set gives insight into how the algorithm will generalize to the unknown test dataset.

Images from the Parkinson's disease group were labeled "1" in a TensorFlow input label file. Images in control and SWEDD groups were labeled "0" since both the groups do not demonstrate imaging signs of dopaminergic deficit. The image sets and their corresponding labels were saved to the appropriate TensorFlow input data directories.

We modified the TensorFlow source code for image extraction with a custom Python script (Fig. 2) in order to enable direct extraction of pixel intensities from raw DICOM files. Each DaTscan image is encoded as an array of 16-bit greyscale intensities with dimensions 91 × 109 pixels. This pixel array is reshaped into a vector of size 9919 pixels to match the input format for TensorFlow's linear algebra computations. The training image dataset is a tensor (an *n*-dimensional array of numbers) with shape (1189, 9199), where the first dimension indexes the image number out of 1189 images in the training dataset. The second dimension indexes the vectorized pixels for each image. The corresponding training label dataset is a tensor of shape (1189, 2), where the first dimension indexes the label number out of 1189 labels in the training dataset. The second dimension indexes the value of "0" for normal or "1" for Parkinson's disease.

The validation image dataset is a tensor of shape (108, 9199) where the first dimension indexes the image number out of 108 images in the training set. The second dimension indexes the vectorized pixels for each image. The corresponding validation label dataset is a tensor of shape (108, 2). The test image and label tensors are of shape (216, 9199) and (216, 2) respectively.

## Neural Network Implementation

All implementations of machine learning algorithms were performed with Google™ TensorFlow API [15]. A single-layer artificial neural network was used to create a simple linear model for binary classification of normal versus Parkinson's disease patients. We chose the gradient descent optimization algorithm as previously described [7] to minimize the difference between ground-truth and predicted labels. In such an algorithm, output errors are fed back through the neural network to update weights and biases. A comparison of gradient descent optimization versus Adagrad optimization in the TensorFlow toolkit [18] was also performed. Training was implemented over 1000 iterations using small batches of randomized data for performance optimization. The TensorBoard tool was executed to view a graphical representation of the neural network. Finally, the neural network's diagnostic performance metrics are evaluated using a first-seen test dataset.

## Statistical Analysis

A two-tailed Mann-Whitney *U* test with significance level of 0.05 was performed to compare differences in accuracy, sensitivity, and specificity between the gradient descent optimizer

```
def extract_images(directory, targetlist):
  print('Extracting', directory)

  filelist = targetlist
  filelist.sort()
  num_images = len(filelist)
  counter = 0

  selectedSlice = 41

  filenames = []
  data = numpy.empty((num_images, 109, 91, 1))

  for filename in filelist:
    filenames.append(filename)
    image = dicom.read_file(os.path.join(directory, filename))
    rows = image.Rows
    assert rows == 109
    cols = image.Columns
    assert cols == 91

    for x in range(0, 109):
      for y in range(0, 91):
        data[counter][x][y] = image.pixel_array[selectedSlice-1][x][y]
    counter = counter + 1

  return data, filenames
```

**Fig. 2** Custom Python script for TensorFlow API. The extract_images function reads in a list of DaTscan SPECT volume filenames called targetlist from directory. A single slice (slice 41) is extracted from each volume. The pixel intensities for *x*,*y* coordinates in each selected slice is loaded into the array data

algorithm and the Adagrad optimizer algorithm. All calculations were performed using a web-based calculator [19].

## Results

### Neural Network Implementation

The TensorBoard output is a graphical representation of the simple single-layer neural network used in this study (Fig. 3). The "x-input" node represents the normalized and vectorized DaTscan pixel intensities, and the "y-input" node represents the label vectors. TensorBoard uses arrows to represent data flow originating from "x-input" and "y-input" nodes into the mathematical operation node "Wx_b", which calculates a weighted sum of pixel intensities plus biases. The output is passed into the "xent" or cross-entropy node, which calculates the loss associated with the predicted output when measured against the ground-truth *y*-input labels. In the "train" node, stochastic gradient descent and Adagrad optimization algorithms minimize cross-entropy by shifting weights and biases in small increments in the direction of reducing cost at every training iteration. Upon completion of training, the final instances of weights and biases are multiplied with "x-input" to yield predicted labels. The "test" node compares the prediction with ground-truth "y-inputs" to yield the accuracy of the model.

### Diagnostic Performance Metrics

TensorBoard generates a graphical output of accuracy and cross-entropy as a function of training iteration. Figure 4a is the output of a single cross-validation, which demonstrates

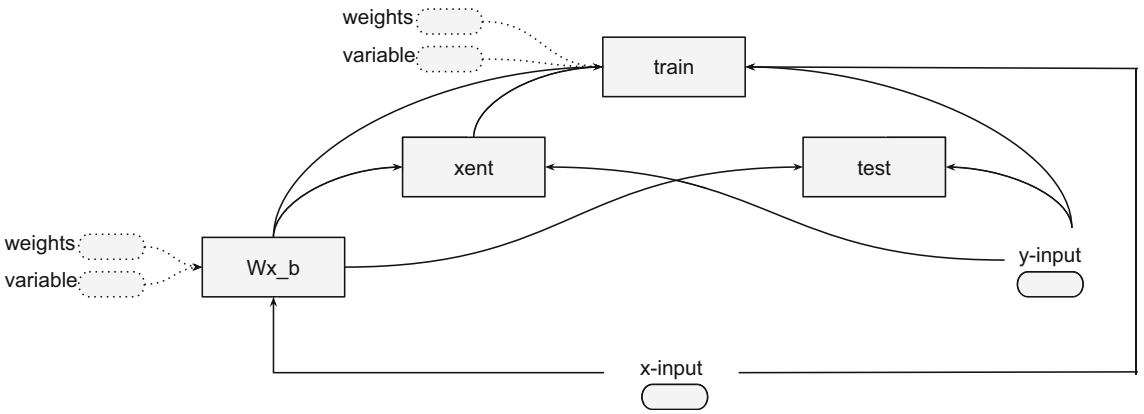


**Fig. 3** Graphical visualization of the neural network. *x*- and *y*-input represent the vectorized DaTscan pixel intensities and corresponding labels, respectively. Wx_b represents a mathematical operation node that applies weights and biases to each *x*-input. The xent node computes the cross-entropy, or cost function, between the predicted and actual labels. Train updates the weights and biases at each training iteration. After completing all training iterations, the test node computes the accuracy, sensitivity, and specificity of the model

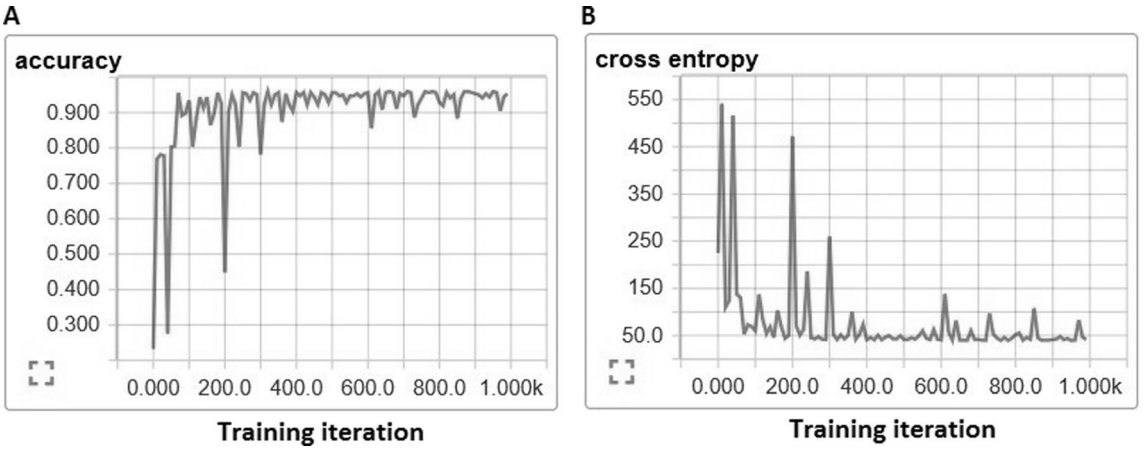


**Fig. 4** Performance summary for a single cross-validation run. Accuracy (**a**) and cross-entropy (**b**) converge with increasing number of training iterations

accuracy convergence for the test dataset using slice 41 after 1000 training iterations. Figure 4b is the output of a single cross-validation, which shows the convergence of cross-entropy towards a minimum for the test dataset using slice 41 after 1000 training iterations.

The accuracy, sensitivity, specificity, false positive fraction, and false negative fraction of the trained neural network are collected for each cross-validation run using slice 41 (Table 1). The gradient descent optimizer algorithm produced a mean accuracy of 0.938 ± 0.047 (95 % CI 0.908–0.967) over a tenfold cross-validation. The mean sensitivity was 0.974 ± 0.043 (95 % CI 0.947–1.00) and the mean specificity was 0.822 ± 0.207 (95 % CI 0.694–0.950). The Adagrad optimizer algorithm produced a mean accuracy of 0.956 ± 0.015 (95 % CI 0.965–0.947) over a tenfold cross-validation. The mean sensitivity was 0.971 ± 0.015 (95 % 0.962–0.981) and the mean specificity was 0.899 ± 0.051 (95 % CI 0.868–0.931). A comparison of gradient descent versus Adagrad optimizers for slice 41 demonstrated no significant difference in accuracy ($p = 1.0$), sensitivity ($p = 0.112$), or specificity ($p = 0.795$).

**Table 1** Quantitative performance measures of two classifier algorithms. The results represent individual statistics for each cross-validation dataset. Note: FP fraction denotes false positive fraction; FN fraction denotes false negative fraction

| Algorithm | Accuracy | Sensitivity | Specificity | FP fraction | FN fraction |
|---|---|---|---|---|---|
| Gradient descent optimizer | 0.95 | 0.97 | 0.90 | 0.02 | 0.02 |
| | 0.88 | 0.86 | 1.00 | 0.00 | 0.12 |
| | 0.84 | 1.00 | 0.38 | 0.16 | 0.00 |
| | 0.90 | 0.99 | 0.52 | 0.09 | 0.00 |
| | 0.99 | 1.00 | 0.94 | 0.01 | 0.00 |
| | 0.95 | 0.99 | 0.83 | 0.04 | 0.01 |
| | 0.97 | 0.97 | 1.00 | 0.00 | 0.03 |
| | 0.95 | 0.98 | 0.88 | 0.03 | 0.02 |
| | 0.96 | 0.99 | 0.86 | 0.04 | 0.00 |
| | 0.97 | 0.99 | 0.91 | 0.02 | 0.00 |
| Adagrad optimizer | 0.97 | 0.99 | 0.91 | 0.02 | 0.01 |
| | 0.94 | 0.98 | 0.80 | 0.04 | 0.01 |
| | 0.95 | 0.97 | 0.90 | 0.02 | 0.02 |
| | 0.95 | 0.98 | 0.87 | 0.03 | 0.01 |
| | 0.95 | 0.94 | 0.96 | 0.01 | 0.04 |
| | 0.95 | 0.97 | 0.90 | 0.02 | 0.02 |
| | 0.97 | 0.98 | 0.93 | 0.01 | 0.01 |
| | 0.94 | 0.95 | 0.85 | 0.03 | 0.04 |
| | 0.94 | 0.96 | 0.89 | 0.02 | 0.03 |
| | 0.98 | 0.98 | 0.98 | 0.00 | 0.01 |

For adjacent striatal slice 40, the gradient descent optimizer algorithm produced a mean accuracy of 0.952 ± 0.040 (95 % CI 0.927–0.977) over a tenfold cross-validation. The mean sensitivity was 0.953 ± 0.053 (95 % CI 0.920–0.986) and the mean specificity was 0.953 ± 0.048 (95 % CI 0.924–0.982). The Adagrad optimizer algorithm produced a mean accuracy of 0.956 ± 0.012 (95 % CI 0.949–0.963) over a tenfold cross-validation. The mean sensitivity was 0.976 ± 0.014 (95 % CI 0.968–0.985) and the mean specificity was 0.887 ± 0.055 (95 % CI 0.853–0.922). A comparison of gradient descent versus Adagrad optimizers for slice 40 input demonstrated no significant difference in accuracy ($p = 0.347$) or sensitivity ($p = 0.150$). However, the gradient descent optimizer produced a significantly greater specificity compared to the Adagrad optimizer ($p = 0.009$) for slice 40 input.

Comparing diagnostic performance using slice 40 versus slice 41 input, the gradient descent optimizer yielded a borderline significant increase in sensitivity with slice 41 input ($p = 0.046$), but produced no differences in accuracy ($p = 0.384$) or specificity ($p = 0.064$). Comparison of slice 40 versus slice 41 input using the Adagrad optimizer showed no significant difference in accuracy ($p = 0.968$), sensitivity ($p = 0.674$), or specificity ($p = 0.624$).

For adjacent striatal slice 42, the gradient descent optimizer algorithm produced a mean accuracy of 0.953 ± 0.023 (95 % CI 0.939–0.968) over a tenfold cross-validation. The mean sensitivity was 0.970 ± 0.019 (95 % CI 0.958–0.982) and the

mean specificity was 0.892 ± 0.144 (95 % CI 0.803–0.981). The Adagrad optimizer algorithm produced a mean accuracy of 0.956 ± 0.016 (95 % CI 0.946–0.966) over a tenfold cross-validation. The mean sensitivity was 0.964 ± 0.023 (95 % CI 0.950–0.979) and the mean specificity was 0.931 ± 0.036 (95 % CI 0.908–0.953). A comparison of gradient descent versus Adagrad optimizers for slice 42 input demonstrated no significant difference in accuracy ($p = 0.912$), sensitivity ($p = 0.849$), or specificity ($p = 0.912$).

Comparing diagnostic performance using slice 41 versus slice 42 input, neither the gradient descent optimizer nor the Adagrad optimizer produced significant differences in accuracy ($p = 0.569$ and 0.704, respectively), sensitivity ($p = 0.187$ and 0.384, respectively), or specificity ($p = 0.327$ and 0.211, respectively).

## Discussion

Artificial neural networks are a well-established class of machine learning approaches for automated image classification. In this study, we implemented a simple single-layer neural network as a proof-of-concept study to interface TensorFlow with raw DICOM inputs at the pixel level.

Our implementation of a neural network in TensorFlow produces diagnostic accuracies on par with excellent results from previous machine learning models. These results indicate TensorFlow's potential role as a useful adjunct diagnostic tool in the clinical setting. Furthermore, the diversity of DaTscan images in the Parkinson's Progression Markers Initiative database, acquired on a variety of scanners across multiple international centers, contributes to the robustness of the machine learning model.

Limitations of our study include single-slice processing of DaTscan SPECT volumes. TensorFlow currently supports analysis of single images and not volumetric data. Consequently, a single axial slice 41 at the basal ganglia with maximal uptake was chosen to populate the training, validation, and test datasets. This technique creates a potential selection bias by effectively eliminating contribution of other axial slices for any given training session. We reduced this bias by separately analyzing adjacent slices 40 and 42 in the basal ganglia to incorporate more imaging volume. The results indicate that TensorFlow input with slice 41 produces a borderline significant increase in sensitivity over slice 40 input (97.4 versus 95.3 %, respectively), but yields no significant differences in accuracy or specificity. Input with slice 41 versus slice 42 produced no significant differences in accuracy, sensitivity, and specificity. Although a congruence in diagnostic performance is reassuring for adjacent slices within the basal ganglia, this method nonetheless represents a departure from human expert reads that evaluate all slices in the imaging volume.

In the absence of a definitive diagnostic test for Parkinson's disease, machine learning models depend on clinical evaluations to accurately label normal subjects versus those with Parkinson's disease during the training process. Therefore, classification accuracy may be affected by variations in clinical confidence. Furthermore, since imaging findings of patients with Parkinson's disease may precede clinical manifestations, the "ground truth" classification labels derived from clinical exams may be a source of additional bias.

The relatively small sample size is a third limitation of our study. State-of-the-art image classification algorithms in computer science utilize tens of thousands of images. For instance, the established computer vision dataset CIFAR-10 contains 50,000 training images and 10,000 testing images for classification into one of ten labeled categories [20]. CIFAR-10 is a subset of an even larger research dataset containing 80 million images for object and scene recognition [21, 22]. Although the PPMI DaTscan database contains a significantly smaller sample size, TensorFlow nevertheless achieved an excellent accuracy and thus merits follow-up studies using its more sophisticated multi-layered models.

Future directions of research may include analyzing radiology images with convolutional neural networks to account for translational variations of abnormal structures. Potential use cases include lung nodule detection or breast cancer screening. The ability to submit DICOM images to TensorFlow for implementing convolutional neural networks opens an exciting next step for machine learning in radiology.

## Conclusion

We extended the TensorFlow API to enable DICOM compatibility as a framework for implementing machine learning classifiers in medical imaging.

**Acknowledgments** PPMI—a public-private partnership—is funded by the Michael J. Fox Foundation for Parkinson's Research and funding partners, including Abbvie, Avid, Biogen, Bristol-Myers Squib, Covance, GE Healthcare, Genentech, GlaxoSmithKline, Eli Lilly & Co, Lundbeck, Merck, Meso Scale Discovery, Pfizer, Piramal, Roche, Servier, and UCB.

## References

1. Darcourt J, Booij J, Tatsch K, et al: EANM procedure guidelines for brain neurotransmission SPECT using 123I-labelled dopamine transporter ligands. Eur J Nucl Med Mol Imaging 37:443–450, 2010. doi:10.1007/s00259-009-1267-x
2. Haller S, Badoud S, Nguyen D, et al: Individual detection of patients with Parkinson disease using support vector machine analysis of diffusion tensor imaging data: initial results. AJNR Am J Neuroradiol 33:2123–2128, 2012. doi:10.3174/ajnr.A3126

3. Singh G, Samavedham L: Unsupervised learning based feature extraction for differential diagnosis of neurodegenerative diseases: A case study on early-stage diagnosis of Parkinson disease. J Neurosci Methods 256:30–40, 2015. doi:10.1016/j.jneumeth.2015.08.011
4. Salvatore C, Cerasa A, Castiglioni I, et al: Machine learning on brain MRI data for differential diagnosis of Parkinson's disease and Progressive Supranuclear Palsy. J Neurosci Methods 222: 230–237, 2014. doi:10.1016/j.jneumeth.2013.11.016
5. Huertas-Fernández I, García-Gómez FJ, García-Solís D, et al: Machine learning models for the differential diagnosis of vascular parkinsonism and Parkinson's disease using [(123)I]FP-CIT SPECT. Eur J Nucl Med Mol Imaging 42:112–119, 2015. doi:10.1007/s00259-014-2882-8
6. Hamilton D, List A, Butler T, et al: Discrimination between parkinsonian syndrome and essential tremor using artificial neural network classification of quantified DaTSCAN data. Nucl Med Commun 27:939–944, 2006
7. Abadi M, Agarwal A, Barham P et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. Available at http://download.tensorflow.org/paper/whitepaper2015.pdf. Accessed 14 April 2016.
8. Frome A, Corrado GS, Shlens J et al. DeVISE: A deep visual-semantic embedding model. Available at http://research.google.com/pubs/archive/41473.pdf. Accessed 15 April 2016.
9. Szegedy C, Liu W, Jia Y et al. Going deeper with convolutions. Available at http://arxiv.org/abs/1409.4842. Accessed 15 April 2016.
10. Rosenberg C. Improving Photo Search: A step across the semantic gap. Available at http://googleresearch.blogspot.com/2013/06/improving-photo-search-step-across.html. Accessed 15 April 2016.
11. Zeiler MD, Ranzato M, Monga R et al. On rectified linear units for speech processing. Available at http://research.google.com/pubs/archive/40811.pdf. Accessed 15 April 2016.
12. Heigold G, Vanhoucke V, Senior A et al. Multilingual acoustic models using distributed deep neural networks. Available at http://research.google.com/pubs/archive/40807.pdf. Accessed 15 April 2016.
13. Hinton GE, Deng L, Yu D, et al: Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Signal Process Mag 29:82–97, 2012
14. TensorBoard. Available at http://www.tensorflow.org/versions/r0.7/how_tos/summaries_and_tensorboard/index.html. Accessed 25 April 2016.
15. Tensorflow. Available at http://www.tensorflow.org. Accessed 25 April 2016.
16. The Parkinson's Progression Markers Initiative (PPMI) Protocol. Available at http://www.ppmi-info.org/wp-content/uploads/2014/01/PPMI-AM7-Protocol.pdf. Accessed 20 August 2016.
17. PPMI Imaging Core. Available at http://www.ppmi-info.org/wp-content/uploads/2011/05/Imaging-Core-Update.pdf. Accessed 20 August 2016.
18. Duchi J, Hazan E, Singer Y: Adaptive subgradient methods for online learning and stochastic optimization. J Mach Learn Res 12: 2121–2159, 2011
19. Mann-Whitney U Test Calculator. Available at http://www.socscistatistics.com/tests/mannwhitney/Default.aspx. Accessed 25 April 2016.
20. Krizhevsky A. Learning multiple layers of features from tiny images. Available at http://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf. Accessed 25 April 2016.
21. The CIFAR-10 dataset. Available at http://www.cs.toronto.edu/~kriz/cifar.html. Accessed 25 April 2016.
22. Torralba A, Fergus R, Freeman WT: 80 Million Tiny Images: A Large Data Set for Nonparametric Object and Scene Recognition. IEEE Trans Pattern Anal Mach Intell 30:1958–1970, 2008