RMDL Documentation

Release latest

Mar 08, 2022

Contents

Referenced paper : RMDL: Random Multimodel Deep Learning for Classification

A new ensemble, deep learning approach for classification. Deep learning models have achieved state-of-the-art results across many domains. RMDL solves the problem of finding the best deep learning structure and architecture while simultaneously improving robustness and accuracy through ensembles of deep learning architectures. RDML can accept as input a variety of data to include text, video, images, and symbolic.

Random Multimodel Deep Learning (RDML) architecture for classification. RMDL includes 3 Random models, oneDNN classifier at left, one Deep CNN classifier at middle, and one Deep RNN classifier at right (each unit could be LSTMor GRU).

CHAPTER 1

Installation

There are pip and git for RMDL installation:

1.1 Using pip

```
pip install RMDL
```

1.2 Using git

```
git clone --recursive https://github.com/kk7nc/RMDL.git
```

The primary requirements for this package are Python 3 with Tensorflow. The requirements.txt file contains a listing of the required Python packages; to install all requirements, run the following:

```
pip -r install requirements.txt
```

Or

```
pip3  install -r requirements.txt
```

Or:

```
conda install --file requirements.txt
```

CHAPTER 2

Documentation:

The exponential growth in the number of complex datasets every year requires more enhancement in machine learning methods to provide robust and accurate data classification. Lately, deep learning approaches have been achieved surpassing results in comparison to previous machine learning algorithms on tasks such as image classification, natural language processing, face recognition, and etc. The success of these deep learning algorithms relys on their capacity to model complex and non-linear relationships within data. However, finding the suitable structure for these models has been a challenge for researchers. This paper introduces Random Multimodel Deep Learning (RMDL): a new ensemble, deep learning approach for classification. RMDL solves the problem of finding the best deep learning structure and architecture while simultaneously improving robustness and accuracy through ensembles of deep learning architectures. In short, RMDL trains multiple models of Deep Neural Network (DNN), Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) in parallel and combines their results to produce better result of any of those models individually. To create these models, each deep learning model has been constructed in a random fashion regarding the number of layers and nodes in their neural network structure. The resulting RDML model can be used for various domains such as text, video, images, and symbolic. In this Project, we describe RMDL model in depth and show the results for image and text classification as well as face recognition. For image classification, we compared our model with some of the available baselines using MNIST and CIFAR-10 datasets. Similarly, we used four datasets namely, WOS, Reuters, IMDB, and 20newsgroup and compared our results with available baselines. Web of Science (WOS) has been collected by authors and consists of three sets~(small, medium and large set). Lastly, we used ORL dataset to compare the performance of our approach with other face recognition methods. These test results show that RDML model consistently outperform standard methods over a broad range of data types and classification problems.

CHAPTER 3

Datasets for RMDL:

3.1 Text Datasets:

- IMDB Dataset
  - This dataset contains 50,000 documents with 2 categories.
- Reters-21578 Dataset
  - This dataset contains 21,578 documents with 90 categories.
- 20Newsgroups Dataset
  - This dataset contains 20,000 documents with 20 categories.
- Web of Science Dataset (DOI: 10.17632/9rw3vkcfy4.2)
  - Web of Science Dataset WOS-11967
    - This dataset contains 11,967 documents with 35 categories which include 7 parents categories.
  - Web of Science Dataset WOS-46985
    - This dataset contains 46,985 documents with 134 categories which include 7 parents categories.
  - Web of Science Dataset WOS-5736
    - This dataset contains 5,736 documents with 11 categories which include 3 parents categories.

3.2 Image datasets:

- MNIST Dataset
  - The MNIST database contains 60,000 training images and 10,000 testing images.
- CIFAR-10 Dataset

  - The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images.

3.3 Face Recognition

The Database of Faces (The Olivetti Faces Dataset)

- The The Database of Faces dataset consists of 400 92x112 colour images and grayscale in 40 person

CHAPTER 4

Requirements for RMDL :

4.1 General:

- Python 3.5 or later see Instruction Documents
- TensorFlow see Instruction Documents.
- scikit-learn see Instruction Documents
- Keras see Instruction Documents
- scipy see Instruction Documents

4.2 GPU (if you want to run on GPU):

- CUDA® Toolkit 8.0. For details, see NVIDIA's documentation.
- The NVIDIA drivers associated with CUDA Toolkit 8.0.
- cuDNN v6. For details, see NVIDIA's documentation.
- GPU card with CUDA Compute Capability 3.0 or higher.
- The libcupti-dev library,

CHAPTER 5

Text and Document Classification

- Download GloVe: Global Vectors for Word Representation Instruction Documents
  - Set data directory into Global.py
  - if you are not setting GloVe directory, GloVe will be downloaded

CHAPTER 6

Parameters:

6.1 Text_Classification

```
from RMDL import RMDL_Text
```

```
Text_Classification(x_train, y_train, x_test,  y_test, batch_size=128,
                    EMBEDDING_DIM=50,MAX_SEQUENCE_LENGTH = 500, MAX_NB_WORDS = 75000,
                    GloVe_dir="", GloVe_file = "glove.6B.50d.txt",
                    sparse_categorical=True, random_deep=[3, 3, 3], epochs=[500, 500,␣
↪500],  plot=True,
                    min_hidden_layer_dnn=1, max_hidden_layer_dnn=8, min_nodes_dnn=128,␣
↪max_nodes_dnn=1024,
                    min_hidden_layer_rnn=1, max_hidden_layer_rnn=5, min_nodes_rnn=32, ␣
↪max_nodes_rnn=128,
                    min_hidden_layer_cnn=3, max_hidden_layer_cnn=10, min_nodes_cnn=128,␣
↪max_nodes_cnn=512,
                    random_state=42, random_optimizor=True, dropout=0.05):
```

6.1.1 Input

- x_train
- y_train
- x_test
- y_test

6.1.2 batch_size

- batch_size: Integer. Number of samples per gradient update. If unspecified, it will default to 128.

6.1.3 EMBEDDING_DIM

- batch_size: Integer. Shape of word embedding (this number should be same with GloVe or other pre-trained embedding techniques that be used), it will default to 50 that used with pain of glove.6B.50d.txt file.

6.1.4 MAX_SEQUENCE_LENGTH

- MAX_SEQUENCE_LENGTH: Integer. Maximum length of sequence or document in datasets, it will default to 500.

6.1.5 MAX_NB_WORDS

- MAX_NB_WORDS: Integer. Maximum number of unique words in datasets, it will default to 75000.

6.1.6 GloVe_dir

- GloVe_dir: String. Address of GloVe or any pre-trained directory, it will default to null which glove.6B.zip will be download.

6.1.7 GloVe_file

- GloVe_dir: String. Which version of GloVe or pre-trained word emending will be used, it will default to glove.6B.50d.txt.
- NOTE: if you use other version of GloVe EMBEDDING_DIM must be same dimensions.

6.1.8 sparse_categorical

- sparse_categorical: bool. When target's dataset is (n,1) should be True, it will default to True.

6.1.9 random_deep

- random_deep: Integer [3]. Number of ensembled model used in RMDL random_deep[0] is number of DNN, random_deep[1] is number of RNN, random_deep[0] is number of CNN, it will default to [3, 3, 3].

6.1.10 epochs

- epochs: Integer [3]. Number of epochs in each ensembled model used in RMDL epochs[0] is number of epochs used in DNN, epochs[1] is number of epochs used in RNN, epochs[0] is number of epochs used in CNN, it will default to [500, 500, 500].

6.1.11 plot

- plot: bool. True: shows confusion matrix and accuracy and loss

6.1.12 min_hidden_layer_dnn

- min_hidden_layer_dnn: Integer. Lower Bounds of hidden layers of DNN used in RMDL, it will default to 1.

6.1.13 max_hidden_layer_dnn

- max_hidden_layer_dnn: Integer. Upper bounds of hidden layers of DNN used in RMDL, it will default to 8.

6.1.14 min_nodes_dnn

- min_nodes_dnn: Integer. Lower bounds of nodes in each layer of DNN used in RMDL, it will default to 128.

6.1.15 max_nodes_dnn

- max_nodes_dnn: Integer. Upper bounds of nodes in each layer of DNN used in RMDL, it will default to 1024.

6.1.16 min_hidden_layer_rnn

- min_hidden_layer_rnn: Integer. Lower Bounds of hidden layers of RNN used in RMDL, it will default to 1.

6.1.17 max_hidden_layer_rnn

- man_hidden_layer_rnn: Integer. Upper Bounds of hidden layers of RNN used in RMDL, it will default to 5.

6.1.18 min_nodes_rnn

- min_nodes_rnn: Integer. Lower bounds of nodes (LSTM or GRU) in each layer of RNN used in RMDL, it will default to 32.

6.1.19 max_nodes_rnn

- max_nodes_rnn: Integer. Upper bounds of nodes (LSTM or GRU) in each layer of RNN used in RMDL, it will default to 128.

6.1.20 min_hidden_layer_cnn

- min_hidden_layer_cnn: Integer. Lower Bounds of hidden layers of CNN used in RMDL, it will default to 3.

6.1.21 max_hidden_layer_cnn

- max_hidden_layer_cnn: Integer. Upper Bounds of hidden layers of CNN used in RMDL, it will default to 10.

6.1.22 min_nodes_cnn

- min_nodes_cnn: Integer. Lower bounds of nodes (2D convolution layer) in each layer of CNN used in RMDL, it will default to 128.

6.1.23 max_nodes_cnn

- min_nodes_cnn: Integer. Upper bounds of nodes (2D convolution layer) in each layer of CNN used in RMDL, it will default to 512.

6.1.24 random_state

- random_state : Integer, RandomState instance or None, optional (default=None)
  - If Integer, random_state is the seed used by the random number generator;

6.1.25 random_optimizor

- random_optimizor : bool, If False, all models use adam optimizer. If True, all models use random optimizers. it will default to True

6.1.26 dropout

- dropout: Float between 0 and 1. Fraction of the units to drop for the linear transformation of the inputs.

6.2 Image_Classification

```
from RMDL import RMDL_Image
```

```
Image_Classification(x_train, y_train, x_test, y_test, shape, batch_size=128,
                         sparse_categorical=True, random_deep=[3, 3, 3], epochs=[500,␣
↪500, 500], plot=True,
                         min_hidden_layer_dnn=1, max_hidden_layer_dnn=8, min_nodes_
↪dnn=128, max_nodes_dnn=1024,
                         min_hidden_layer_rnn=1, max_hidden_layer_rnn=5, min_nodes_
↪rnn=32, max_nodes_rnn=128,
                         min_hidden_layer_cnn=3, max_hidden_layer_cnn=10, min_nodes_
↪cnn=128, max_nodes_cnn=512,
                         random_state=42, random_optimizor=True, dropout=0.05)
```

6.2.1 Input

- x_train
- y_train
- x_test
- y_test

6.2.2 shape

- shape: np.shape . shape of image. The most common situation would be a 2D input with shape (batch_size, input_dim).

6.2.3 batch_size

- batch_size: Integer. Number of samples per gradient update. If unspecified, it will default to 128.

6.2.4 sparse_categorical

- sparse_categorical: bool. When target's dataset is (n,1) should be True, it will default to True.

6.2.5 random_deep

- random_deep: Integer [3]. Number of ensembled model used in RMDL random_deep[0] is number of DNN, random_deep[1] is number of RNN, random_deep[0] is number of CNN, it will default to [3, 3, 3].

6.2.6 epochs

- epochs: Integer [3]. Number of epochs in each ensembled model used in RMDL epochs[0] is number of epochs used in DNN, epochs[1] is number of epochs used in RNN, epochs[0] is number of epochs used in CNN, it will default to [500, 500, 500].

6.2.7 plot

- plot: bool. True: shows confusion matrix and accuracy and loss

6.2.8 min_hidden_layer_dnn

- min_hidden_layer_dnn: Integer. Lower Bounds of hidden layers of DNN used in RMDL, it will default to 1.

6.2.9 max_hidden_layer_dnn

- max_hidden_layer_dnn: Integer. Upper bounds of hidden layers of DNN used in RMDL, it will default to 8.

6.2.10 min_nodes_dnn

- min_nodes_dnn: Integer. Lower bounds of nodes in each layer of DNN used in RMDL, it will default to 128.

6.2.11 max_nodes_dnn

- max_nodes_dnn: Integer. Upper bounds of nodes in each layer of DNN used in RMDL, it will default to 1024.

6.2.12 min_nodes_rnn

- min_nodes_rnn: Integer. Lower bounds of nodes (LSTM or GRU) in each layer of RNN used in RMDL, it will default to 32.

6.2.13 max_nodes_rnn

- maz_nodes_rnn: Integer. Upper bounds of nodes (LSTM or GRU) in each layer of RNN used in RMDL, it will default to 128.

6.2.14 min_hidden_layer_cnn

- min_hidden_layer_cnn: Integer. Lower Bounds of hidden layers of CNN used in RMDL, it will default to 3.

6.2.15 max_hidden_layer_cnn

- max_hidden_layer_cnn: Integer. Upper Bounds of hidden layers of CNN used in RMDL, it will default to 10.

6.2.16 min_nodes_cnn

- min_nodes_cnn: Integer. Lower bounds of nodes (2D convolution layer) in each layer of CNN used in RMDL, it will default to 128.

6.2.17 max_nodes_cnn

- min_nodes_cnn: Integer. Upper bounds of nodes (2D convolution layer) in each layer of CNN used in RMDL, it will default to 512.

6.2.18 random_state

- random_state : Integer, RandomState instance or None, optional (default=None)
  - If Integer, random_state is the seed used by the random number generator;

6.2.19 random_optimizor

- random_optimizor : bool, If False, all models use adam optimizer. If True, all models use random optimizers. it will default to True

6.2.20 dropout

- dropout: Float between 0 and 1. Fraction of the units to drop for the linear transformation of the inputs.

CHAPTER 7

Example

7.1 MNIST

- The MNIST database contains 60,000 training images and 10,000 testing images.

7.1.1 Import Packages

```
from keras.datasets import mnist
import numpy as np
from RMDL import RMDL_Image as RMDL
```

7.1.2 Load Data

```
(X_train, y_train), (X_test, y_test) = mnist.load_data()
X_train_D = X_train.reshape(X_train.shape[0], 28, 28, 1).astype('float32')
X_test_D = X_test.reshape(X_test.shape[0], 28, 28, 1).astype('float32')
X_train = X_train_D / 255.0
X_test = X_test_D / 255.0
number_of_classes = np.unique(y_train).shape[0]
shape = (28, 28, 1)
```

7.1.3 Using RMDL

```
batch_size = 128
sparse_categorical = 0
n_epochs = [100, 100, 100]  ## DNN-RNN-CNN
Random_Deep = [3, 3, 3]  ## DNN-RNN-CNN

```

(continues on next page)

(continued from previous page)

```
RMDL.Image_Classification(X_train, y_train, X_test, y_test,shape,
                      batch_size=batch_size,
                      sparse_categorical=True,
                      random_deep=Random_Deep,
                      epochs=n_epochs)
```

7.2 IMDB

- This dataset contains 50,000 documents with 2 categories.

7.2.1 Import Packages

```
import sys
import os
from RMDL import text_feature_extraction as txt
from keras.datasets import imdb
import numpy as np
from RMDL import RMDL_Text as RMDL
```

7.2.2 Load Data

```
print("Load IMDB dataset....")
(X_train, y_train), (X_test, y_test) = imdb.load_data(num_words=MAX_NB_WORDS)
print(len(X_train))
print(y_test)
word_index = imdb.get_word_index()
index_word = {v: k for k, v in word_index.items()}
X_train = [txt.text_cleaner(' '.join(index_word.get(w) for w in x)) for x in X_train]
X_test = [txt.text_cleaner(' '.join(index_word.get(w) for w in x)) for x in X_test]
X_train = np.array(X_train)
X_train = np.array(X_train).ravel()
print(X_train.shape)
X_test = np.array(X_test)
X_test = np.array(X_test).ravel()
```

7.2.3 Using RMDL

```
batch_size = 100
sparse_categorical = 0
n_epochs = [100, 100, 100]  ## DNN--RNN-CNN
Random_Deep = [3, 3, 3]  ## DNN--RNN-CNN

RMDL.Text_Classification(X_train, y_train, X_test, y_test,
                      batch_size=batch_size,
                      sparse_categorical=sparse_categorical,
                      random_deep=Random_Deep,
                      epochs=n_epochs)
```

7.3 Web Of Science

- Linke of dataset:
  - Web of Science Dataset WOS-11967
    - This dataset contains 11,967 documents with 35 categories which include 7 parents categories.
  - Web of Science Dataset WOS-46985
    - This dataset contains 46,985 documents with 134 categories which include 7 parents categories.
  - Web of Science Dataset WOS-5736
    - This dataset contains 5,736 documents with 11 categories which include 3 parents categories.

7.3.1 Import Packages

```
from RMDL import text_feature_extraction as txt
from sklearn.model_selection import train_test_split
from RMDL.Download import Download_WOS as WOS
import numpy as np
from RMDL import RMDL_Text as RMDL
```

7.3.2 Load Data

```
path_WOS = WOS.download_and_extract()
fname = os.path.join(path_WOS,"WebOfScience/WOS11967/X.txt")
fnamek = os.path.join(path_WOS,"WebOfScience/WOS11967/Y.txt")
with open(fname, encoding="utf-8") as f:
    content = f.readlines()
    content = [txt.text_cleaner(x) for x in content]
with open(fnamek) as fk:
    contentk = fk.readlines()
contentk = [x.strip() for x in contentk]
Label = np.matrix(contentk, dtype=int)
Label = np.transpose(Label)
np.random.seed(7)
print(Label.shape)
X_train, X_test, y_train, y_test = train_test_split(content, Label, test_size=0.2,␣
↪random_state=4)
```

7.3.3 Using RMDL

```
batch_size = 100
sparse_categorical = 0
n_epochs = [5000, 500, 500]  ## DNN--RNN-CNN
Random_Deep = [3, 3, 3]  ## DNN--RNN-CNN

RMDL.Text_Classification(X_train, y_train, X_test, y_test,
                         batch_size=batch_size,
                         sparse_categorical=True,
                         random_deep=Random_Deep,
                         epochs=n_epochs,no_of_classes=12)
```

7.4 Reuters-21578

- This dataset contains 21,578 documents with 90 categories.

7.4.1 Import Packages

```
import sys
import os
import nltk
nltk.download("reuters")
from nltk.corpus import reuters
from sklearn.preprocessing import MultiLabelBinarizer
import numpy as np
from RMDL import RMDL_Text as RMDL
```

7.4.2 Load Data

```
documents = reuters.fileids()

train_docs_id = list(filter(lambda doc: doc.startswith("train"),
                            documents))
test_docs_id = list(filter(lambda doc: doc.startswith("test"),
                           documents))
X_train = [(reuters.raw(doc_id)) for doc_id in train_docs_id]
X_test = [(reuters.raw(doc_id)) for doc_id in test_docs_id]
mlb = MultiLabelBinarizer()
y_train = mlb.fit_transform([reuters.categories(doc_id)
                             for doc_id in train_docs_id])
y_test = mlb.transform([reuters.categories(doc_id)
                        for doc_id in test_docs_id])
y_train = np.argmax(y_train, axis=1)
y_test = np.argmax(y_test, axis=1)
```

7.4.3 Using RMDL

```
batch_size = 100
sparse_categorical = 0
n_epochs = [20, 500, 50]  ## DNN--RNN-CNN
Random_Deep = [3, 0, 0]  ## DNN--RNN-CNN

RMDL.Text_Classification(X_train, y_train, X_test, y_test,
             batch_size=batch_size,
             sparse_categorical=True,
             random_deep=Random_Deep,
             epochs=n_epochs)
```

7.5 Olivetti Faces

- There are ten different images of each of 40 distinct subjects. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial

details (glasses / no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement).

7.5.1 Import Packages

```
from sklearn.datasets import fetch_olivetti_faces
from sklearn.model_selection import train_test_split
from RMDL import RMDL_Image as RMDL
```

7.5.2 Load Data

```
number_of_classes = 40
shape = (64, 64, 1)
data = fetch_olivetti_faces()
X_train, X_test, y_train, y_test = train_test_split(data.data,
                                                    data.target, stratify=data.target, test_
↪size=40)
X_train = X_train.reshape(X_train.shape[0], 64, 64, 1).astype('float32')
X_test = X_test.reshape(X_test.shape[0], 64, 64, 1).astype('float32')
```

7.5.3 Using RMDL

```
batch_size = 100
sparse_categorical = 0
n_epochs = [500, 500, 50]  ## DNN--RNN-CNN
Random_Deep = [0, 0, 1]  ## DNN--RNN-CNN

RMDL.Image_Classification(X_train, y_train, X_test, y_test,
                          shape,
                          random_optimizor=False,
                          batch_size=batch_size,
                          random_deep=Random_Deep,
                          epochs=n_epochs)
```

More Exanmple link

7.6 Error and Comments:

Send an email to kk7nc@virginia.edu

7.7 Citations

```
@inproceedings{Kowsari2018RMDL,
title={RMDL: Random Multimodel Deep Learning for Classification},
author={Kowsari, Kamran and Heidarysafa, Mojtaba and Brown, Donald E. and Jafari␣
↪Meimandi, Kiana and Barnes, Laura E.},
booktitle={Proceedings of the 2018 International Conference on Information System and␣
↪Data Mining},
```

(continues on next page)

(continued from previous page)

```
year={2018},
DOI={https://doi.org/10.1145/3206098.3206111},
organization={ACM}
}
```