% All Rights Reserved.

% ----------------------------------------------------------------------

% Permission to use, copy, or modify this software and its documentation for educational and research purposes only and without fee is here granted, provided that this copyright notice and the original authors' names appear on all copies and supporting documentation. This program shall not be used, rewritten, or adapted as the basis of a commercial software or hardware product without first obtaining permission of the authors. The authors make no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

%----------------------------------------------------------------------

List of Folders:

JSTSP\_publish – Main folder. Make it Matlab's home directory.

databases – This folder contains all data needed to run the simulations.

Fastfood – Main folder of Fastfood method [see 4] described in our paper. No need to run the code from this folder.

LCKSVD – Main folder of LC-KSVD method [see 2] described in our paper. To run LC-KSVD simulations refer to this folder instead of the main folder. Run the files 'DEMO\_LCKSVD\_LKDL\_YaleB' and 'DEMO\_LCKSVD\_LKDL\_ARface'.

packages – Folder containing software packages of omp and ksvd. For installation instructions of these packages, please refer to: <http://www.cs.technion.ac.il/~ronrubin/software.html>. You must install these packages before running our wrapper code.

Simplified FDDL – Main folder of FDDL method [see 3] described in our paper. In order to execute the FDDL simulation from our paper, run the file called: 'DEMO\_FDDL\_LKDL\_USPS'.

%%%%%%%%%%%%%%%%%%%%

List of files in main folder, JSTSP\_publish:

Scripts to create figures appearing in our paper (Follow the instructions in each file):

create\_FIGURE\_1a – create figure showing quality of approximation of the Nystrom method versus the number of selected samples from train set.

create\_FIGURE\_1b – create figure showing classification accuracy of LKDL versus the number of selected samples from the train set.

create\_FIGURE\_2 – create figure showing classification accuracy versus increasing levels of corruption in the form of Gaussian noise and missing pixels.

create\_FIGURE\_3 – create figure showing classification accuracy, training time and test time versus size of input training set.

create\_FIGURE\_4a – create figure showing classification accuracy versus size of input feature vector. This figure compares LKDL feature extraction on the USPS dataset, versus RFF [see 5] and Fastfood [see 4].

create\_FIGURE\_4b – create figure showing classification accuracy versus size of input feature vector. This figure compares LKDL feature extraction on the MNIST dataset, versus RFF [see 5] and Fastfood [see 4].

DEMO\_batch\_LKDL – script running the RLS-DLA algorithm [see 6] of mini-batch dictionary learning, with and without LDKL pre-processing.

---------------------------------------

List of files with pre-calculated results:

RESULTS\_FIGURE\_1a

RESULTS\_FIGURE\_1b

RESULTS\_FIGURE\_2

RESULTS\_FIGURE\_3

RESULTS\_FIGURE\_4a

RESULTS\_FIGURE\_4b

-----------------------------------------

List of main wrapper functions of entire classification pipeline:

classify\_main – external wrapper function for training dictionaries and performing classification on MNIST and USPS datasets. Follow instructions inside.

classify\_aux – internal wrapper function within classify\_main.m

classify\_aux\_batch – mini-batch version of internal wrapper function within classify\_main.m

----------------------------------------

List of functions for training and classifying with KSVD, KKSVD and RLS-DLA algorithms:

init\_dictionary – internal function to initialize dictionaries before performing dictionary learning. See function for different options of initialization.

KSVD\_train – train a dictionary using KSVD.

KSVD\_classify – perform classification using dictionary created with KSVD.

KKSVD\_train – train a dictionary using KKSVD.

KKSVD\_classify – perform classification using dictionary created with KKSVD.

RLS\_train – train a dictionary using RLS-DLA (mini-batch version of MOD).

RLS\_classify – perform classification using dictionary created with RLS-DLA (mini-batch MOD).

--------------------------------------

List of functions connected with LKDL feature extraction:

calc\_virtual\_map – create virtual train and test sets using LKDL.

calc\_support – perform sub-sampling on the train data for Nystrom approximation.

calc\_kernel – compute kernel on input Gram matrix.

randpdf – utility function for sampling from a custom distribution function.

fkmeans – utility function for performing fast kmeans (for Nystrom approximation).

KSVDCoresetAlg – utility function to calculate coreset sampling.

------------------------------------

List of functions for KKSVD (provided by Hien Van Nguyen, see 1):

KKSVD – perform KKSVD dictionary learning. See further

KOMP – perform KOMP sparse coding.

Knorms – utility function of KKSVD. See further instructions inside.

gram – utility function of KKSVD. See further instructions inside.

normcols – utility function of KKSVD. See further instructions inside.

--------------------------------------

List of functions for RLS-DLA algorithm:

dict\_learn\_mb\_simple – main function for training dictionary using mini-batch RLS-DLA.

lambdafun – auxiliary function to update the forgetting factor in RLS-DLA.

------------------------------------------

Additional functions:

create\_enlarged\_MNIST – load MNIST dataset and create enlarged version with 1-pixel shifted images in each direction (including diagonal).

mycolormap – load colormap for displaying figures.

% Please refer to the following papers

% Alona Golts and Michael Elad, "Linearized Kernel Dictionary Learning", in Journal of Selected Topics in Signal Processing, June 16'.

%-------------------------------------------------------------------------

Contact: zadneprovski@gmail.com

%---------------------note------------------------------------

We use the code provided by the following authors and presented in these papers:

[1] Nguyen, H., Patel, V. M., Nasrabadi, N. M., & Chellappa, R. (2012, March). Kernel dictionary learning. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on* (pp. 2021-2024). IEEE.

[2] Jiang, Z., Lin, Z., & Davis, L. S. (2011, June). Learning a discriminative dictionary for sparse coding via label consistent K-SVD. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on* (pp. 1697-1704). IEEE.

[3] Yang, M., Zhang, L., Feng, X., & Zhang, D. (2011, November). Fisher discrimination dictionary learning for sparse representation. In *Computer Vision (ICCV), 2011 IEEE International Conference on* (pp. 543-550). IEEE.

[4] Le, Q., Sarlós, T., & Smola, A. (2013). Fastfood-computing hilbert space expansions in loglinear time. In *Proceedings of the 30th International Conference on Machine Learning* (pp. 244-252).

[5] Rahimi, A., & Recht, B. (2007). Random features for large-scale kernel machines. In *Advances in neural information processing systems* (pp. 1177-1184).

[6] Skretting, K., & Engan, K. (2010). Recursive least squares dictionary learning algorithm. *Signal Processing, IEEE Transactions on*, *58*(4), 2121-2130.

%---------------------note------------------------------------