CSE569

Project Topics

Fall 2017

Important Notes:

1. You will need to write a midterm report on your progress, for which you may earn up to 5% credit. Due Nov. 10.

2. There will be a poster presentation, during the lecture time on the last day of class (Dec. 1), for the project, accounting for 5%. The final report is also due by that day.

3. If you are choosing any projects among option 1-6, Vijetha will be your assigned TA. If youare choosing any projects among option 7-13, Kevin will be your assigned TA. Only contact your assigned TA when discussing these projects.

Deliverables

The deliverables for the final project are the following:

1. **Report**: A final report must be produced that must contain a brief introduction to the problem, description of the methodology used, including the students' own interpretation of the methodology, and potential areas of use for the study along with thorough literature survey or related works. The report must also contain description of the implementation, including potential problems encountered during the implementations, assumptions made and reasons for such assumptions and such. The report shall also contain figures and tables containing results with appropriate discussions. There must also be discussion, conclusion and future scope sections.

   The final report is due on **December 1,2017** by midnight. The report must be typeset and adhere to the guidelines and formatting rules pertaining to the camera ready version of IEEE Transactions on Pattern Analysis and Machine Intelligence. All submissions of the report will be handled electronically via Blackboard. Length of the report should be no more than 6 pages excluding references.

2. **Code**: You must submit all the codes used and developed during, and for this project. The codes are expected to run in both sequential and modular fashion.

   (a) Sequential in the sense that one trigger must make the whole code run in sequence without any disruption until results (visual and metric) are produced.

(b) Modular in the sense that the grader must be able to start the code at different *meaningful* sections of the problem such as beginning of testing stage or evaluation stage.

(c) You must also provide an estimated time for completion and profiling for each section of the code. If possible , do display real-time progress on the code.

The code must be handed in with MIT License only. This implies that any toolbox or code that you borrow and adopt must also have appropriate licenses and policies for you to allow this submission. If there are third party licenses involved in the project, or the libraries used or borrowed have licenses on their own ensure that you adhere to it's terms and conditions. Improper use of licensed material will constitute a violation of the academic integrity. You may use any programming languages including proprietary languages, tools and software including, and not limited to Matlab and Mathematica as long as licenses for those software are available for the graders at myASU-myApps. You may not use software that involve a license purchase.

In any cases of borrowed code, be it from forums or external sources, appropriate citation must be provided. Apart from this citation you must include an independent file containing filename and line numbers of borrowed code along with the source of the borrowed code including license. Borrowing code without appropriate citation will be considered a violation of academic integrity. It is safe to include every piece of borrowed code in this citation file.

3. **Synthetic Datasets**: If your project involves the creation of synthetic datasets, try creating datasets that are not easy but also stick to lower dimensional spaces as these synthetic datasets can and should help you visualize the labeling, classification and decision boundaries etc. You will need to submit a code for creating these datasets as part of supplementary material.

4. **Supplementary material**: The code, synthetic datasets, results and other material that are not part of but additional to the report are collectively referred to as supplementary material. A submission link will be created separate from the report submission for the submission of supplementary material. You may upload all the supplementary material at once as a compressed file. If it exceeds blackboard capability, please schedule an appointment (plan accordingly) with the instructor ahead of the deadline to make a submission using hard transfer. Add all supplementary material in the following manner:

   (a) Separate directories for data and code: The code must contain a detailed readme file detailing dependencies, software including compiler / OS requirements and libraries used. If you require external libraries detailed instruction on acquisition and installation must be provided in a manner that is OS independent. If OS dependent, you must provide details.

   (b) License of code (must be MIT). Submission will be void if you don't have a license along with the submission.

   (c) It is your responsibility that your code works on the grader's machine. If you expect discrepencies or trouble, make sure the submission is done with appropriate timing. A missed deadline is not the grader's responsibility.

   (d) Images and other results in addition to the datasets in separate directories.

   (e) Anonymized submissions to other conferences, journals, and appendices or technical reports and project reports for other classes related to and emerging from this project that may contain extended proofs and mathematical derivations that are not essential to the understanding of the submitted report.

   Do not submit datasets, but instead provide a download link and a script for setting up. Supplementary material are limited to 50MB. Submission will be considered void if it exceeds this limit.

Academic Integrity: You are expected to maintain the utmost level of academic integrity in the project. Any violation of the code of academic integrity will be reported to the dean for official actions. It is an academic violation to copy, to include text from other sources, including online sources for both material and code, without proper citation and licensing. To get a better idea of what constitutes plagiarism, consult the ASU policy on student obligations. This is a serious violation and evidence of plagiarism or academic dishonesty, will likely result in failing the course and at worse can lead to disqualification from your degree program. Please contact the TA before borrowing material when unsure. Refer to the section on licensing in the code submission requirements.

1 The global k-means clustering algorithm Aristidis Likas, Nikos Vlassis, Jakob J. Verbeek Pattern Recognition, 2003 [20]

Aim:
In this project, you will implement a global k-means clustering algorithm presented in [20]. The solution of the original k-means clustering algorithm relies heavily on the choice of the initial cluster centers. Thus the solution achieved is not optimal. The paper presents an incremental approach in obtaining the optimal clusters for a given data-set. The abstract of the paper is given below.

Abstract:
We present the global k-means algorithm which is an incremental approach to clustering that dynamically adds one cluster center at a time through a deterministic global search procedure consisting of N (with N being the size of the data set) executions of the k-means algorithm from suitable initial positions. We also propose modifications of the method to reduce the computational load without significantly affecting solution quality. The proposed clustering methods are tested on well-known data sets and they compare favorably to the k-means algorithm with random restarts.

Tasks:

The specific tasks that you need to complete as a part of this project are as follows.

- Understand the key idea of the paper. Analyze the run times for various versions of the algorithm presented in the paper and document.
- You have studied Expectation Maximization (EM) algorithm in the course related to the context of Maximum Likelihood Estimate. k-means utilizes the concept of EM to learn the cluster centers. Write a note on the relation between EM and k-means clustering. What are the E and M steps here?
- Implement the algorithms, (i) Original k-means (section 10.4.3 of the textbook) (ii) Global k-means (section 2 of the paper) (iii) Fast global k-means initialized with k-d trees (section 3). You can use any library functions for PCA. However, implement everything else on your own.
- Perform clustering on two data-sets, (1) Artificially generated Mixture of Gaussians (2) MNIST digits [14]. Analyze if the results on the artificial and real dataset are in alignment with what the paper is claiming. Report your observations.
- For both the data-sets compute the clustering error given by the equation (1) of the paper for all the three approaches mentioned above. Compute the errors for various values of ks and plot graphs like the ones given in section 4 of the paper.

Notes:

- TA can support if the implementation is in Python.
- Perform as many experiments for various settings of the parameters based on the availability of computational resources to you.

2 SMOTE: Synthetic Minority Over-sampling Technique
Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, W. Philip Kegelmeyer
Journal of Artificial Intelligence Research, 2002 [15]

Aim:
Dealing with imbalanced datasets is a significant problem in Machine Learning as samples from one class are significantly high in number than the samples from another class, thus leading to a heavily biased model towards the dominant class. Real world examples of such scenarios include, physiological data of cancer tumor vs non-cancer tumor, weather data of storm vs non-storm etc. In this problem setting, metrics like accuracy would be significantly high even for a bad model and do not provide an accurate estimate of the model performance. In this project, you will deal with overcoming such scenario by artificially generating samples of the less frequent class. You will also learn metrics that would analyze the model performance to a better extent than the accuracy metric. The below is the abstract of the paper [15] you will be implementing.

Abstract: An approach to the construction of classifiers from imbalanced data-sets is described. A data-set is imbalanced if the classification categories are not approximately equally represented. Often real-world data sets are predominately composed of "normal" examples with only a small percentage of "abnormal" or "interesting" examples. It is also the case that the cost of mis-classifying an abnormal (interesting) example as a normal example is often much higher than the cost of the reverse error. Under-sampling of the majority (normal) class has been proposed as a good means of increasing the sensitivity of a classifier to the minority class. This paper shows that a combination of our method of over-sampling the minority (abnormal) class and under-sampling the majority (normal) class can achieve better classifier performance (in ROC space) than only under-sampling the majority class. This paper also shows that a combination of our method of over-sampling the minority class and under-sampling the majority class can achieve better classifier performance (in ROC space) than varying the loss ratios in Ripper or class priors in Naive Bayes. Our method of over-sampling the minority class involves creating synthetic minority class examples. Experiments are performed using C4.5, Ripper and a Naive Bayes classifier. The method is evaluated using the area under the Receiver Operating Characteristic curve (AUC) and the ROC convex hull strategy.

Tasks: The specific tasks that need to be performed as a part of this project are as follows.

- Understand from a higher level the existing methods used to tackle the problem of imbalanced data-sets (section 3 of [15]). Understand the performance metrics used to evaluate the models for this kind of problem (section 2 of [15]). Write a brief notes on the existing methods and performance measures.
- Implement the algorithm SMOTE presented in section 4.2 of the paper. The implementation is straight forward and you should not need any sophisticated libraries. You also need to implement the approaches given in sections 4.1 and 4.3 of the paper in order to compare the results. However, for the task of classification, you are free to use any tool box or advanced libraries.
- Pick any 3 data-sets out of the 9 data-sets given in section 5.1 of the paper. Replicate the results from sections 5.2 and 5.3. Compare your results with the ones given in the graphs of sections 5.2 and 5.3.

Notes: The TA can support if the implementation is in Python.

3 Implement a Neural Network from Scratch

Aim:
In this project, you will implement a Neural Network for the task of digit classification [14]. Neural Networks are complex non-linear models used extensively in several machine learning application. Stochastic Gradient Descent (SGD) is the standard optimization algorithm used to train neural networks. As a part of this project, you will implement a 2 hidden layer neural network, that should be trained using SGD. You will then implement a recent regularization scheme called Dropout [25] and analyze the test set performance with and without dropout.

Network Specifications:
The architecture of the network you should implement will be as follows. The input layer will have 784 neurons(same as the dimensionality of the input data). Both the hidden layers will have 256 neurons each, and the output layer will have 10 neurons. Apply *sigmoid* activation [3] on the hidden neurons and *softmax* activation [4] on the output neurons. Apply *categorical cross entropy loss* (slide 6 of [13], also called as negative log probability) on this network and back propagate the loss. Apply dropout on the two hidden layers. Try various dropout rates and analyze the train and test set performances.

Tasks:
The specific tasks you need to perform as a part of this project are as follows.

- Understand neural network training using stochastic gradient descent thoroughly. Try to answer the following questions,
  Is the neural network(with 1 or more hidden layers and non-linear activations) optimization concave or convex or neither (with respect to the parameters of the network)? Explain.
  What happens when you impose a linear activation function on all the neurons?
  Write briefly on why dynamic programming should be used in SGD implementation?
- Implement a neural network with the architecture described above. You should implement it using analytic gradients.
- Train the network for MNIST [14] dataset. Based on the capacity of your computer, choose any number of samples between $10,000$ to $50,000$ for training, 5000 for validation and $5,000$ for testing. However, make sure to have equal number of samples in all the classes in each of the three splits. Plot how the loss behaves for training and validation sets as the training progresses.
- Verify if the gradients obtained from analytic derivatives match the gradients obtained numerically. Use the theory presented in [10] to compute numerical gradients.
- Read sections 1 to 5 of the paper [25] and implement dropout. Note that the dropout layer is not applied during testing phase. Build your code to incorporate this fact.

Notes:

- The TA can support if the implementation is in Python.
- All the implementation should be done with out using any advanced libraries. Only usage of basic libraries like numpy and it's equivalent in Matlab are allowed.
- Scale the data between [0, 1](i.e divide the data by 255) before feeding in to the network.
- As the loss is categorical cross entropy, you need to encode the labels as one-hot vectors (slides 1 and 2 of [13])

4 An Introduction to Locally Linear Embedding Lawrence K. Saul, Sam T. Roweis Science, 2000 [23]

Aim:
The goal of this project is to understand and implement a non-linear dimensionality reduction technique called Local Linear Embedding. In this technique, the problem of dimensionality reduction is tackled by exploiting the local symmetries of linear reconstructions. To attempt this project successfully, you need to first understand the algorithm from [23] and the follow the experiments and evaluation part from [27].

Abstract Many problems in information processing involve some form of dimensionality reduction. Here we describe locally linear embedding (LLE), an unsupervised learning algorithm that computes low dimensional, neighborhood preserving embeddings of high dimensional data. LLE attempts to discover nonlinear structure in high dimensional data by exploiting the local symmetries of linear reconstructions. Notably, LLE maps its inputs into a single global coordinate system of lower dimensionality, and its optimizations—though capable of generating highly nonlinear embeddings—do not involve local minima. We illustrate the method on images of lips used in audiovisual speech synthesis.

Tasks: More detailedly, the set of tasks you need to perform for this project are as follows:

- Read and understand the LLE algorithm from section 2 of the paper [23]. Explain the technique in your own words. Understand the evaluation metrics, *trustworthiness* and *continuity* of the low dimensional features presented in the the section 6.1 of [27].
- Implement the algorithm from scratch(without using any libraries)
- Create five artificial datasets namely, the helix, the twin peaks, the swiss roll, the broken swiss roll and the high dimensional dataset. Create them using the formulae given in the appendix of [27]. Also, download the natural dataset called MNIST from [14]. You will be evaluating the technique on the five artificial and one natural datasets.
- For evaluation, you should follow the approach in [27]. i.e evaluate the approach for (1) 1-NN Classification trained on the obtained low dimensional features, (2) Measure the trustworthiness and the continuity of the low-dimensional features. Compare the results obtained with PCA.

Notes:

- The TA can support if the implementation is in Python.

5 Supervised Discrete Hashing
Fumin Shen, Chunhua Shen, Wei Liu, Heng Tao Shen Computer Vision and Pattern Recognition, 2015 [24]

Aims:
Through this project, you will be introduced to a new problem called Image Hashing, which is of significant importance in the area of Image Retrieval. Image Hashing is the task of mapping images to binary vectors such that similar images are mapped to similar binary codes and dissimilar ones are mapped to dissimilar codes. The notion of similarity is application specific. For the sake of this project, we will call images belonging to the same class as similar and different classes as dissimilar. In this project, you will implement a specific paper [24] which tries to formulate the problem of image hashing as an optimization problem and uses the procedure of alternate optimization to learn the unknowns. The abstract of the paper is given below.

Abstract:
Recently, learning based hashing techniques have attracted broad research interests because they can support efficient storage and retrieval for high-dimensional data such as images, videos, documents, etc. However, a major difficulty of learning to hash lies in handling the discrete constraints imposed on the pursued hash codes, which typically makes hash optimizations very challenging (NPhard in general). In this work, we propose a new supervised hashing framework, where the learning objective is to generate the optimal binary hash codes for linear classification. By introducing an auxiliary variable, we reformulate the objective such that it can be solved substantially efficiently by employing a regularization algorithm. One of the key steps in this algorithm is to solve a regularization sub-problem associated with the NP-hard binary optimization. We show that the sub-problem admits an analytical solution via cyclic coordinate descent. As such, a high-quality discrete solution can eventually be obtained in an efficient computing manner, therefore enabling to tackle massive datasets. We evaluate the proposed approach, dubbed Supervised Discrete Hashing (SDH), on four large image datasets and demonstrate its superiority to the state-of-the-art hashing methods in large-scale image retrieval.

Tasks:
The main tasks for this project are as follows.

- Understand the problem of Image Hashing and the solution presented in the paper from a higher level. Understand the metrics precision and recall in the context of image retrieval.
- Implement the optimization algorithm presented in the paper. Only implement the algorithm for l_2 loss (section 2.2). Also, implement functions to compute precision and recall.
- Download the digit dataset called MNIST from [14]. Follow the procedure given in the experiments (section 3.4) and try to replicate the results shown in Figure 2 of the paper.

Notes:

- The TA can support if the implementation is in Python.
- You need to do the above experiment on MNIST for all the hash codes length described in the paper.
- You need to scale MNIST raw features between [0,1] as preprocessing (just divide by 255).

6 A Global Geometric Framework for Nonlinear Dimensionality Reduction
Joshua B. Tenenbaum, Vin de Silva, John C. Langford Science 2000[26]

Aim:
The goal of this project is to understand and implement a non-linear dimensionality reduction technique called Isomaps. In this technique, the problem of dimensionality reduction is tackled by preserving the pariwise geodesic distance of the data points. To attempt this project successfully, you need to first understand the algorithm from [26] and the follow the experiments and evaluation part from [27].

Abstract:
Scientists working with large volumes of high-dimensional data, such as global climate patterns, stellar spectra, or human gene distributions, regularly confront the problem of dimensionality reduction: finding meaningful low-dimensional structures hidden in their high-dimensional observations. The human brain confronts the same problem in everyday perception, extracting from its high-dimensional sensory inputs—$30,000$ auditory nerve fibers or 10^6 optic nerve fibers—a manageably small number of perceptually relevant features. Here we describe an approach to solving dimensionality reduction problems that uses easily measured local metric information to learn the underlying global geometry of a data set. Unlike classical techniques such as principal component analysis (PCA) and multidimensional scaling (MDS), our approach is capable of discovering the nonlinear degrees of freedom that underlie complex natural observations, such as human handwriting or images of a face under different viewing conditions. In contrast to previous algorithms for nonlinear dimensionality reduction, ours efficiently computes a globally optimal solution, and, for an important class of data manifolds, is guaranteed to converge asymptotically to the true structure.

Tasks:
More detailedly, the set of tasks you need to perform for this project are as follows:

- Read and understand the Isomaps technique. You can refer to other tutorials on the internet to understand the technique. Some of them are given here for your reference. [26], [1], [2]. Explain the technique in your own words. Understand the evaluation metrics, *trustworthiness* and *continuity* of the low dimensional features presented in the the section 6.1 of [27].
- Implement the algorithm from scratch(without using any libraries even for the MDS part)
- Create five artificial datasets namely, the helix, the twin peaks, the swiss roll, the broken swiss roll and the high dimensional dataset. Create them using the formulae given in the appendix of [27]. Also, download the natural dataset called MNIST from [14]. You will be evaluating the technique on the five artificial and one natural datasets.
- For evaluation, you should follow the approach in [27]. i.e evaluate the approach for (1). 1-NN Classification trained on the obtained low dimensional features, (2). Measure the trustworthiness and the continuity of the low-dimensional features. Compare the results obtained with PCA.

Notes:
The TA can support if the implementation is in Python.

7 Discriminative K-SVD for dictionary learning in face recognition
Qiang Zhang and Baoxin Li, CVPR2010 [22]

Abstract:
In a sparse-representation-based face recognition scheme, the desired dictionary should have good representational power (i.e., being able to span the subspace of all faces) while supporting optimal discrimination of the classes (i.e., different human subjects). We propose a method to learn an over-complete dictionary that attempts to simultaneously achieve the above two goals. The proposed method, discriminative K-SVD (D-KSVD), is based on extending the K-SVD algorithm by incorporating the classification error into the objective function, thus allowing the performance of a linear classifier and the representational power of the dictionary being considered at the same time by the same optimization procedure. The D-KSVD algorithm finds the dictionary and solves for the classifier using a procedure derived from the K-SVD algorithm, which has proven efficiency and performance. This is in contrast to most existing work that relies on iteratively solving sub-problems with the hope of achieving the global optimal through iterative approximation. We evaluate the proposed method using two commonly-used face databases, the Extended YaleB database and the AR database, with detailed comparison to 3 alternative approaches, including the leading state-of-the-art in the literature. The experiments show that the proposed method outperforms these competing methods in most of the cases. Further, using Fisher criterion and dictionary incoherence, we also show that the learned dictionary and the corresponding classifier are indeed better-posed to support sparse-representation-based recognition.

Tasks:
The goal of this project is to implement the D-KSVD described in [22].

- Read and understand the paper. Explain the main ideas of the paper in your own words.
- Implement the D-KSVD algorithm by using the KSVD-BOX.
- Train and test on synthetic data. (*i.e.* generate some clusters for X, the elements inside the same cluster are in the same class)
- Train and test on a face dataset (*i.e.* each column of X represents a face). Remember to use the RandomFace matrix to extract the feature.
- Analyze the results (*i.e.* testing error, etc).

Notes:

- The KSVD-Box can be downloaded in the following link [12].
- You can use the Extended Yale Face Database B (cropped version) [11]

8 Two-dimensional PCA: a new approach to appearance-based face representation and recognition
Jian Yang, David Zhang, Alejandro F. Frangi and Jing-yu Yang, TPAMI2004[28]

Abstract:
In this paper, a new technique coined two-dimensional principal component analysis (2DPCA) is developed for image representation. As opposed to PCA, 2DPCA is based on 2D image matrices rather than 1D vectors so the image matrix does not need to be transformed into a vector prior to feature extraction. Instead, an image covariance matrix is constructed directly using the original image matrices, and its eigenvectors are derived for image feature extraction. To test 2DPCA and evaluate its performance, a series of experiments were performed on three face image databases: ORL, AR, and Yale face databases. The recognition rate across all trials was higher using 2DPCA than PCA. The experimental results also indicated that the extraction of image features is computationally more efficient using 2DPCA than PCA.

Tasks:
The goal of this project is to implement the 2DPCA described in [28].

- Read and understand the paper. Explain the main ideas of the paper in your own words.
- Implement the 2DPCA algorithm described in section 2 and 3.
- Apply the algorithm on a face dataset.
- Analyze the result (you should implement at least one of the experiments did in the paper).

Notes:

- You can use the Extended Yale Face Database B (cropped version) [11] or other face dataset.

9 Algorithms for Non-negative Matrix Factorization Daniel D. Lee and H. Sebastian Seung, NIPS2000[19]

Abstract:
Non-negative matrix factorization (NMF) has previously been shown to be a useful decomposition for multivariate data. Two different multiplicative algorithms for NMF are analyzed. They differ only slightly in the multiplicative factor used in the update rules. One algorithm can be shown to minimize the conventional least squares error while the other minimizes the generalized Kullback-Leibler divergence. The monotonic convergence of both algorithms can be proven using an auxiliary function analogous to that used for proving convergence of the Expectation- Maximization algorithm. The algorithms can also be interpreted as diagonally rescaled gradient descent, where the rescaling factor is optimally chosen to ensure convergence.

Tasks: The goal of this project is to implement the NMF algorithm described in [19].

- Read and understand the paper. Explain the main ideas of the paper in your own words.
- Write a program to factorize V into W and H using the update rule in Theorem 1.
- Apply NMF on face dataset (*i.e.* each column of V represents a face). Visualize W and H, analyze the results.

Notes:

- If you are interested in this topic, you can read the paper [18].

10 Convex and Semi-Nonnegative Matrix Factorizations Chris Ding, Tao Li and Michael I. Jordan, TPAMI2010[17]

Abstract:
We present several new variations on the theme of nonnegative matrix factorization (NMF). Considering factorizations of the form $X = FG^T$, we focus on algorithms in which G is restricted to containing nonnegative entries, but allowing the data matrix X to have mixed signs, thus extending the applicable range of NMF methods. We also consider algorithms in which the basis vectors of F are constrained to be convex combinations of the data points. This is used for a kernel extension of NMF. We provide algorithms for computing these new factorizations and we provide supporting theoretical analysis. We also analyze the relationships between our algorithms and clustering algorithms, and consider the implications for sparseness of solutions. Finally, we present experimental results that explore the properties of these new methods.

Tasks:

- Read and understand the paper. Explain the main ideas of the paper in your own words.
- Implement the algorithms for semi NMF and convex NMF, using the update rule mentioned in the paper.
- Generate at least one synthetic dataset, and apply the semi NMF and convex NMF on the dataset
- Analyze the results (*i.e.* by using figures, plots, etc).

11 Learning with Noisy Labels
Nagarajan Natarajan, Inderjit S. Dhillon, and Pradeep Ravikumar, NIPS2013[21]

Abstract:
In this paper, we theoretically study the problem of binary c lassification in the presence of random classification noise — the learner, instead of seeing the true labels, sees labels that have independently been flipped with some small probability. Moreover, random label noise is class-conditional — the flip probability depends on the class. We provide two approaches to suitably modify any given surrogate loss function. First, we provide a simple unbiased estimator of any loss, and obtain performance bounds for empirical risk minimization in the presence of iid data with noisy labels. If the loss function satisfies a simple symmetry condition, we show that the method leads to an efficient algorithm for empirical minimization. Second, by leveraging a reduction of risk minimization under noisy labels to classification with weighted 0-1 loss, we suggest the use of a simple weighted surrogate loss, for which we are able to obtain strong empirical risk bounds. This approach has a very remarkable consequence — methods used in practice such as biased SVM and weighted logistic regression are provably noise-tolerant. On a synthetic non-separable dataset, our methods achieve over 88% accuracy even when 40% of the labels are corrupted, and are competitive with respect to recently proposed methods for dealing with label noise in several benchmark datasets.

Tasks:
In this project, you are expected to study the problem of binary classification in the presence of random classification noise—the learner, instead of seeing the true labels, sees labels that have independently been flipped with some small probability. Moreover, random label noise is class-conditional— the flip probability depends on the class. Your task in this project is to implement the first method in paper [21], evaluate your implementation in the synthetic dataset you created and report the performance ad observations. Goals of the project:

- Read and understand the paper. Explain the main ideas of the paper in your own words.
- Create at least one set of synthetic dataset, split it into training and testing parts, and add some label noise to the training set.
- Pick a classification method with lost function that satisfied the conditions mentioned in paper [21] and build the unbiased estimator of the original loss function (see Lemma 1).
- Train your classifier on the synthetic noisy dataset (dataset (1)), and generate prediction results on dataset (2).
- Analyze your results (*i.e.* by using figures, error rates, etc).

12 Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs) Djork-Arné Clevert, Thomas Unterthiner and Sepp Hochreiter, ICLR2016[16]

Abstract:
We introduce the "exponential linear unit" (ELU) which speeds up learning in deep neural networks and leads to higher classification accuracies. Like rectified linear units (ReLUs), leaky ReLUs (LReLUs) and parametrized ReLUs (PReLUs), ELUs alleviate the vanishing gradient problem via the identity for positive values. However, ELUs have improved learning characteristics compared to the units with other activation functions. In contrast to ReLUs, ELUs have negative values which allows them to push mean unit activations closer to zero like batch normalization but with lower computational complexity. Mean shifts toward zero speed up learning by bringing the normal gradient closer to the unit natural gradient because of a reduced bias shift effect. While LReLUs and PReLUs have negative values, too, they do not ensure a noise-robust deactivation state. ELUs saturate to a negative value with smaller inputs and thereby decrease the forward propagated variation and information. Therefore, ELUs code the degree of presence of particular phenomena in the input, while they do not quantitatively model the degree of their absence. In experiments, ELUs lead not only to faster learning, but also to significantly better generalization performance than ReLUs and LReLUs on networks with more than 5 layers. On CIFAR-100 ELUs networks significantly outperform ReLU networks with batch normalization while batch normalization does not improve ELU networks. ELU networks are among the top 10 reported CIFAR-10 results and yield the best published result on CIFAR-100, without resorting to multi-view evaluation or model averaging. On ImageNet, ELU networks considerably speed up learning compared to a ReLU network with the same architecture, obtaining less than 10% classification error for a single crop, single model network.

Tasks:

- Read and understand the paper. Explain the main ideas of the paper in your own words.
- Implement at least one neural network on MNIST dataset[14], by using Sigmoid, ReLU, ELU as the activation function.
- Analyze the result (*i.e.* running time, testing error rate, etc.)

Notes:

- You are not expected to use GPU for training, so you can use a small neural network (*i.e.* a neural network with 6 layers).
- Do not use different types of activation functions in the same network.

13 Recommender System

Introduction

This project will be an exploration into modern recommender systems. Implemented models will range from a basic global expectation to Latent Factor Models utilizing SVD and gradient descent (the most popular recommender in the 2009 Netflix challenge). The suggested programming languages are Python and Matlab—other languages may be used though. Every step has been given detailed thought, however, since this project has not been implemented before, it's possible unforeseen errors could arise, necessitating modifications in the requirements or implementation methodology. In addition, this proposal is largely based on the lectures from Stanford University: Mining Massive Datasets [9]—which we *highly* refer to for additional information.

DataSet

Multiple datasets are listed to facilitate various interests in addition to testing the generalizability of the models. In order for the recommender models to work, each dataset must have a training and test set split. The split between the training and test set will be 80%-20%, respectively. To create the test set, withhold 20% of the data and use that to validate each model. The test data should NOT be used to train the models. In order to use a dataset other than the primary ones, it must have user ratings for each object (movie, TV show, etc.), otherwise the algorithms will not function as intended.

The first two will be the primary datasets:

1. Movie Dataset [7] —20M Dataset
2. Anime Dataset [8]
3. Alternative Datasets [6]

Step 0: Pre-processing

Due to the scale of the datasets, they are not suited to being held in sparse matrix form in memory. For example, a matrix of size 10k $\times$ 10k will have a hard time fitting in memory, let alone something of much more practical size—100k $\times$ 100k. Therefore, a package like Scipy's Sparse Matrix will be needed to store the data in compressed form (e.g. compressed sparse row—CSR). More information on CSR and additional matrix storage reduction techniques can be found on Wikipedia[5].

The second part of the pre-processing step is to split the training and test data from the original dataset (assuming separate test data is not already given for your chosen dataset). An example of how to do this can be seen in Figure 1 below. The figure does not represent an 80%-20% split of training to test data respectively, however, the actual data should.

Figure 1: Training/Test Dataset Split

Output

1. Training dataset as compressed sparse row (CSR) matrix using Scipy Sparse Matrix package. This can optionally be left in COO format.
2. Test dataset as Coordinate list (COO) matrix using Scipy Sparse Matrix package.

Instructions

1. Choose an appropriate dataset to work with for this assignment.
2. Determine the row and column indexes to partition the data into the training set and test set.
3. Load *all* the data into the training set using Coordinate list (COO) format. After the data has been loaded, convert the matrix to Compressed sparse row (CSR) format for faster arithmetic and matrix operations. Entries with no ratings **along with** the test data should be entered as zeros.
4. Load the actual test data values into another matrix using Coordinate list (COO) format.

Step 1: Baseline

This algorithm will be fairly basic. It will consider a user's tendency to rate movies relative to the global mean over all movies. For each new movie being considered, this system will produce a result based on the sum of the global mean, the mean of how other users rated this movie relative to the global mean, and this user's rating tendency relative to the global mean.

$$r_{xi} = b_{xi} = \mu + b_x + b_i \tag{1}$$

- r_{xi} = Rating of user x on item i
- b_{xi} = Global baseline estimate on user x and item i
- μ = Overall mean movie rating
- b_x = Rating deviation of user x = (avg. rating of user x across all movies user x has rated) - μ
- b_i = (avg. rating of movie i) - μ

$$\text{Root-mean-square error (RMSE)} = \sqrt{\frac{\sum_{(x,i)\epsilon T}(r_{xi} - \hat{r}_{xi})^2}{N}} \tag{2}$$

- r_{xi} = The true rating of user x on item i
- $\hat{r}_{xi}$ = The predicted rating of user x on item i
- $N = |T|$
- T = Test set

Output

The output will be the RMSE on the test set.

Instructions

1. Calculate the global baseline statistics on the training data: μ, b_x, b_i.
2. Using the baseline statistics, calculate the predicted value, r_{xi}, for each entry in the test set using equation 1.
3. Find the error between the predicted values and actual values by calculating the Root-Mean-Square Error (RMSE) using equation 2.

Step 2: Item-Item Collabrative Filtering

This model will use the similarity between items to predict a user's rating. The goal will be to predict which movies (or items) a given user likes based on the item-item similarities. Similar movies will be determined by using the Pearson Correlation (centered cosine similarity), with only the N most similar movies considered.

$$r_{xi} = b_{xi} + \frac{\sum_{j\epsilon N(i;x)} s_{ij} \times (r_{xj} - b_{xj})}{\sum_{j\epsilon N(i;x)} s_{ij}} \tag{3}$$

- r_{xi} = Rating of user x on item i
- r_{xj} = Rating of user x on item j
- b_{xi} = Global baseline estimate on user x and item i
- b_{xj} = Global baseline estimate on user x and item j
- $N(i;x)$ = Set of items rated by x similar to i
- s_{ij} = Similarity of objects (items, users) i and j

$$s_{ij} = cos(\theta) = \frac{A \times B}{||A||_2 \times ||B||_2} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}} \tag{4}$$

Output

The output will be the RMSE on the test set. Compare this to the baseline approach in Step 1.

Instructions

1. Re-use the global baseline values calculated in Step 1.
2. For each row of movies, calculate the mean of the known ratings (empty entries are not included in the calculation).
3. Subtract the mean from each value in the respective rows (empty entries are again not included). This is the 'centered' part of the cosine similarity.
4. For each entry that you want to predict in the training set (aka the values in the test), compute the cosine similarity between the row of the chosen movie and every other row, as seen in equation 4.
5. Select an appropriate Neighbor Selection value—$N(i; x)$. For example, a value of 5 means that you are going to look for 5 movies similar to that movie rated by user 'x'. In practice, this means you will select the 5 most similar rows to your chosen movie row.
6. Calculate the baseline parameter b_{xj} for each neighbor following the same method as b_{xi}.
7. Calculate the predicted value, r_{xi}, for your test set entry.
8. Repeat instructions 4-6 for each non-zero entry in the test set.
9. Find the error between the predicted values and actual values by calculating the Root-Mean-Square Error (RMSE) using equation 2.

Step 3: Latent Factor Model

The idea behind the Latent Factor Model is to decompose the matrix, R, using Singular Value Decomposition (SVD) into two separate components Q and P instead of the traditional three components that SVD produces. Next, the decomposed matrices, Q and P are optimized using the Root Square Summed Error (RMSE) and gradient descent. The missing values along with the test data in R should be set to 0.

- u = number of users
- i = number of items
- k = number of factors

- R = $R_{i \times u}$
- Q = $R_{i \times k}$
- P = $R_{k \times u}$
- q_i = row i of Q
- p_x = column x of P^T
- $\hat{r}_{xi}$ = Estimated rating of user x on item i
- λ = user defined regularization parameter, $0 \leq \lambda \leq 1$

Traditional SVD:

$$R = U \Sigma V^T \tag{5}$$

Modified SVD where Q can be thought of as U and P^T can be thought of as ΣV^T.

$$R \approx Q \times P^T \tag{6}$$

Estimation for missing rating on user x and item i, r_{xi}. This is effectively a dot product of item row q_i and user column p_x.

$$\hat{r}_{xi} = q_i \times q_x^T = \sum_f q_{if} \times p_{xf} \tag{7}$$

SVD isn't defined when there are empty values in matrix R, therefore we need to solve the optimization problem given below over the *known* ratings of R, instead of all the cells in the matrix. This will optimize the matrices P and Q by minimizing the error between the predicted value and the actual value. Once P and Q are optimized, they will allow us to predict the unknown ratings in the test set.

$$J(P,Q) = min_{P,Q} \sum_{(i,x) \in R} (r_{xi} - q_i \times p_x^T)^2 \tag{8}$$

In order to minimize the error between the predicted value and the actual value, we need to minimize J(P,Q) by optimizing P and Q. This can be done using gradient descent which takes steps opposite to the direction of function gradient. In addition, to prevent overfitting of the data we need to introduce a user set regularization parameter, λ. When labmda is set to 0, it will create a model that begins to fit the noise in the data. On the other hand, when the value of lambda is 1, it forces the model to become overly simple—resulting in underfitting of the data. Therefore, it's recommended that values of labmda be chosen closer to the middle of it's range.

$$J(P,Q) = min_{P,Q} \sum_{training} (r_{xi} - q_i \times p_x^T)^2 + \lambda [\sum_x ||p_x||^2 + \sum_i ||q_i||^2] \tag{9}$$

The equations below will be used to implement gradient descent, optimizing matrices P and Q. Matrices P and Q are initialized using SVD as discussed above.

$$P \leftarrow P * \eta * \bigtriangledown P \tag{10}$$

$$Q \leftarrow Q * \eta * \bigtriangledown Q \tag{11}$$

$\bigtriangledown Q$ is the gradient of matrix Q (partial derivative). q_{ik} is entry k of row q_i of matrix Q. The calculation for $\bigtriangledown P$ is repeated similarly below.

$$\bigtriangledown \text{ Q} = [\bigtriangledown q_{ik}] \tag{12}$$

$$\bigtriangledown q_{ik} = \sum_{xi} -2(r_{xi} - q_i p_x^T) p_{xk} + 2\lambda q_{ik} \tag{13}$$

$$\bigtriangledown \text{ P} = [\bigtriangledown p_{ik}] \tag{14}$$

$$\bigtriangledown p_{ik} = \sum_{xi} -2(r_{xi} - q_i p_x^T) q_{xk} + 2\lambda p_{ik} \tag{15}$$

Output

1. The output will be the RMSE on the test set. Compare this to the baseline approach in Step 1 and Item-Item Collabrative Filtering in Step 2.

Instructions

1. Calculate the SVD or matrix R and create matrices P and Q.
2. Optimize matrices P and Q by performing gradient descent using equations 10-15.
3. Calculate the RMSE error on the test set by using the predicted value in equation 7 with equation 2.

References

[1] http://cseweb.ucsd.edu/classes/fa02/cse252c/branson_isomap.pdf.

[2] http://math.stanford.edu/~yuany/course/2017.spring/lecture09.1.pdf.

[3] https://en.wikipedia.org/wiki/Sigmoid_function.

[4] https://en.wikipedia.org/wiki/Softmax_function.

[5] https://en.wikipedia.org/wiki/Sparse_matrix.

[6] https://gab41.lab41.org/the-nine-must-have-datasets-for-investigating-recommender-systems-ce9421

[7] https://grouplens.org/datasets/movielens/.

[8] https://www.kaggle.com/CooperUnion/anime-recommendations-database/data.

[9] https://www.youtube.com/playlist?list=PLLssT5z_DsK9JDLcT8T62VtzwyW9LNepV.

[10] http://ufldl.stanford.edu/wiki/index.php/Gradient_checking_and_advanced_optimization.

[11] http://vision.ucsd.edu/~iskwak/ExtYaleDatabase/ExtYaleB.html.

[12] http://www.cs.technion.ac.il/~ronrubin/software.html.

[13] http://www.cs.toronto.edu/~guerzhoy/321/lec/W04/onehot.pdf.

[14] http://yann.lecun.com/exdb/mnist/.

[15] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[16] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

[17] Chris HQ Ding, Tao Li, and Michael I Jordan. Convex and semi-nonnegative matrix factorizations. *IEEE transactions on pattern analysis and machine intelligence*, 32(1):45–55, 2010.

[18] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788, 1999.

[19] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

[20] Aristidis Likas, Nikos Vlassis, and Jakob J Verbeek. The global k-means clustering algorithm. *Pattern recognition*, 36(2):451–461, 2003.

[21] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *Advances in neural information processing systems*, pages 1196–1204, 2013.

[22] Qiang Zhang and Baoxin Li. Discriminative k-svd for dictionary learning in face recognition. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2691–2698, June 2010.

[23] Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

[24] Fumin Shen, Chunhua Shen, Wei Liu, and Heng Tao Shen. Supervised discrete hashing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 37–45, 2015.

[25] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958, 2014.

[26] Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

[27] Laurens Van Der Maaten, Eric Postma, and Jaap Van den Herik. Dimensionality reduction: a comparative review. *J Mach Learn Res*, 10:66–71, 2009.

[28] Jian Yang, D. Zhang, A. F. Frangi, and Jing yu Yang. Two-dimensional pca: a new approach to appearance-based face representation and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1):131–137, Jan 2004.