Building A Deep Learning Classifier for Enhancing a Biomedical Big Data Service

Junhua Ding
Dept. of Computer Science
East Carolina University
Greenville, NC, USA
dingj@ecu.edu

Xiaojun Kang
School of Computer Science
China University of Geosciences
Wuhan, Hubei, China
xj_kang@126.com

Xin-Hua Hu
Dept. of Physics
East Carolina University
Greenville, NC, USA
hux@ecu.edu

Venkat Gudivada
Dept. of Computer Science
East Carolina University
Greenville, NC, USA
gudivadav15@ecu.edu

***Abstract*—Providing an easily accessible data service with high quality data is important for building big data applications. In this paper, we introduce a big data service for managing and accessing massive-scale biomedical image data. The service includes three major components: a NoSQL database for storing images and data analytics results, a client consisting of a group of query scripts for data access and management, and a data quality enhancement component for improving the performance of data analytics. Low-quality data can result in incorrect analytics results and may lead to no value even harmful conclusions. Therefore, it is important to provide an effective mechanism for ensuring data quality improvement in a big data service. We describe the implement ion of a deep learning classifier to automatically filter low quality data in datasets. To improve the effectiveness of data separation, the classifier is rigorously validated with synthetic data generated by a collection of scientific tools. Design of big data services with data quality improvement as an integral component, along with the best practices collected from this experimental study, will help researchers and practitioners to develop strategies for improving the quality of big data services, building big data applications, and designing machine learning classifiers.**

***Keywords*-machine learning, deep learning, big data, data quality, cross-validation.**

I. Introduction

Providing an easily accessible big data service with high quality data is important for building a big data application. We define a big data service as an online service for managing and accessing massive-scale data. Due to the characteristics of large volume, fast growing, variety of data types, and big value of big data [1], it is fairly challenging to build a big data service that can offer high quality data on demand. For example, the volume of a typical big dataset could be several hundred gigabytes and sharing the data is often needed. An easy solution is to store the data in a database or a file system on a remotely accessible server. However, downloading the data remotely could be a very slow process. For example, the volume of the training data of AlexNet [2] is over 140GB [3], which is a relatively small dataset compared to other training data for deep learning. Downloading the data from the AlexNet project website is a very slow process and takes several days. Even with a peer-to-peer download tool, the download may still take several hours. However, a user probably does not need the whole dataset in many cases. A big data service should offer the flexibility for accessing the data according to the user need. To provide an easy to use big data service, we built a NoSQL database for managing massive-scale biomedical images in a flexible way. We defined a set of scripts that can be easily executed in the client application for accessing the almost any data in the database. A user can define his or her own scripts through customizing the script templates or existing scripts offered in the service.

The data in a big data service repository may grow quite rapidly. New data may increase data heterogeneity as well as introduce low quality data such as invalid or incorrectly labeled data. Low quality data can result in incorrect analytics which would provide no value or even result in harmful conclusions. In this case, providing a function to support users to automatically separate low quality data from valid data is a desirable feature. The quality attributes of big data include availability, usability, reliability, and relevance. Each attribute includes detail quality attributes such as credibility, integrity, and completeness [4]. The focus of this research is on improving data reliability such as filtering out unwanted noisy data or correcting incorrectly assigned labels. There are many approaches for filtering unwanted data such as setting a simple threshold like a mean or max value or defining a sophisticated filtering rule such as those defined for completeness or currency of data attributes.

In this paper, we introduce a deep learning approach for building the filtering rule for automated separation of noisy data from massive-scale biomedical image data. The noisy data include invalid data items like blur or overexposed images and valid data items that were incorrectly labeled, known as *class label noise*. For example, a horse in an image is incorrectly labeled as a donkey. The approach is developed based on a deep learning [5] classifier for automatically classifying image data into several categories, where noisy data and valid data are separated into different categories. To validate the deep learning classification, the classifier verifies with perfect images in different categories. Perfect images are produced using a group of modeling tools. If the images in each category are classified with high accuracy, the effectiveness of the classifier is high and vise versa. Finally, we compare the classification accuracy of the deep learning classification to a Support Vector Machine (SVM) [6] classification on the same

2474-2473/17 $31.00 © 2017 IEEE
DOI 10.1109/SCC.2017.25

dataset.

The big data service we built is used for studying automated cell classification based on the diffraction images of cells. Diffraction images of cells are acquired using a polarization diffraction imaging flow cytometer (p-DIFC), which was invented and developed by co-author Hu [7]. The 3D morphological features of a cell captured in the diffraction image can be used for accurately classifying cell types, which is central to many branches of biology and life sciences research. Co-authors Ding and Hu have been studying cell morphology assay and classification for over a decade. p-DIFC can take the diffraction images of near 100 cells each second. We have collected a large number of diffraction images including many types of cells at different apoptosis phases, and many images are being added to the database daily. However, cell samples for p-DIFC imaging include non-cell particles such as ghost cell bodies or aggregated spherical particles and cell debris. The diffraction images taken from non-cell particles are also collected. When a machine learning classifier is trained for cell classification, the noisy images should be filtered out. It is possible to separate many noisy images from cell images based on their different fringe patterns in the images. However, manually separating the noise images from a large amount of diffraction image data is inefficient since the difference of the fringe patterns in some images is difficult to be observed by eyes. We developed a deep learning classifier in the big data service for automatically classifying the diffraction images into three categories: diffraction images of viable cells with intact structures (or simply called as *cells*), diffraction images of ghost cell bodies or aggregated spherical particles (or simply called *fractured cells*), and diffraction images of cell debris or small particles (or simply called *debris*). The latter two categories are treated as noisy data in cell classification.

The rest of this paper is organized as follows: Section 2 introduces the measurement and calculation of cell imaging, and the design of the big data service. Section 3 describes the experimental study of a deep learning classifier for improving the image data quality. Section 4 discusses the related work and Section 5 concludes the paper.

II. Morphology Based Cell Imaging

In this section, we first describe morphology based cell imaging and classification, and then discuss the study of the relation between cell morphology and its diffraction images. Lastly, we describe the design of the big data service.

A. Morphology Based Cell Classification

Cells are fundamental elements of life and possess highly varied and convoluted 3D structures by intracellular organelles to sustain their phenotypic variations and functions. Morphology based cell classification at the single-cell level attracts intense research efforts recently for their direct relations to cellular functions. p-DIFC is used to acquire cross-polarized Diffraction Image (p-DI) pairs from single cells [7]. Fig. 1 shows a schematic diagram of p-DIFC and examples of diffraction image pairs acquired from individual flowing cells.

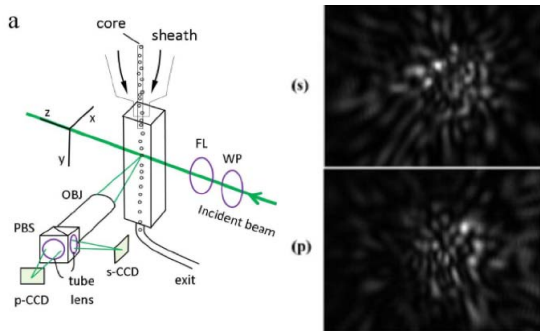


Fig. 1. (a). Schematic of p-DIFC, (s). a sample s-polarization DI, and (p). a sample p-polarization DI.

The s-polarization image and p-polarization image are images acquired by only the s-polarization or the p-polarization of the scattered light, respectively. Different from images acquired by non-coherent light, the p-DI pairs present characteristic patterns due to the coherent light scatter emitted by the intracellular molecular dipoles induced by an incident laser beam. The p-DI data thus provide a big data source to probe the 3D morphology of the illuminated cells. Co-authors Hu and Ding *et al.* have built a Big Data to Knowledge (BD2K) infrastructure [8] for studying the fast and precise cell typing based on cell morphology. The BD2K includes a big data services for managing the massive-scale image data and data analytics results, scientific software for understanding the theoretic foundation of the morphology based cell typing, and machine learning approaches for rapid and accurate cell morphology analysis based on diffraction images of cells.

B. SVM based Image Classification

SVMs are supervised learning models associated with learning algorithms that build a set of hyperplanes in a high-dimensional space through analyzing data for classification or regression analysis [6]. An SVM performs binary classification in general. Given a training dataset, each data item in the training dataset is labeled by the category it belongs to, and then an SVM training algorithm constructs a model to classify test items to a category. However, several SVM classifiers can be combined to implement a multiclass classifier by comparing *'one against the rest'* or *'one against one'*. LIBSVM [9], an open source toolkit for SVM, is used for conducting SVM classification in our projects.

SVM has been used for classifying the cell types based on diffraction images [10] [11] [12] [13]. The SVM feature vector of a diffraction image consists of its GLCM feature values and its label. GLCM [14] defines the textual pattern of an image with the statistics of the spatial relationship of pixels. It defines how often different combinations of gray level pixels occur in an image for a given displacement/distance d in a particular angle θ. The definitions of GLCM features for diffraction images include 14 original GLCM features and 6 extended features for diffraction images, which can be found in Ding *et al.* previous publications [13]. The GLCM features

quantitatively characterize the fringe pattern in a diffraction image. The procedure of building an SVM classifier for diffraction images can be summarized as follows: 1. Calculate the GLCM features for each diffraction image in the training dataset and the test dataset. 2. Label each diffraction image with its category, and build a feature vector consisting of its GLCM feature values and its label. The feature vectors of all diffraction images in the training dataset form a feature matrix. 3. Train the SVM classifier using the feature matrix. 4. Test the classifier with diffraction images in the test dataset, and validate the classification accuracy using N-fold Cross Validation (NFCV) and confusion matrix.

C. Modeling Light Scattering of Cells

To investigate the correlation between the fringe pattern of a cell diffraction image and the 3D morphology of the cell, we model the light scattering properties of a cell based on its 3D morphology parameters using a scientific software tool called ADDA [15]. The Muller matrix from ADDA simulation is used to produce diffraction images using a ray-tracing technique [16]. The correlation between the 3D morphology parameters and the fringe pattern of the diffraction image is established through an experimental study, which systematically changes the values of the 3D parameters to see the corresponding changes of the fringe pattern in the diffraction images. To generate the 3D morphology parameters of a cell, a stack of confocal image sections are taken from the cell using a confocal microscope. Next, the confocal image sections are reconstructed for the 3D structure of the cell, and each cell organelle in the reconstructed 3D structure is assigned with a refractive index value. The 3D morphology parameters comprise a 3D structure with assigned refractive index value for every cell organelle. We call a diffraction image that is calculated from ADDA as "calculated diffraction image," and a diffraction image that is taken using p-DIFC as "measured diffraction image." In this research, we used the calculated diffraction image to validate the accuracy of the deep learning for classification of diffraction images.

D. The Design of a Massive-Scale Image Data Service

The big data service includes a NoSQL database MongoDB, and a client application that is implemented on MongoChef (it was renamed as Studio3T recently) [17]. The database stores image data, image processing and data analytics results. The client application includes a group of predefined scripts that wrap JSON queries for providing data services such as importing or exporting images, querying or updating data in the database. Users invoke services by running the service scripts. If the script for a requested service is not available, a user can create a script through customizing a pre-defined script template or modifying an existing script. Large data are split into several chunks to support range queries using the MongoDB grid file system. It provides the flexibility to query any portion of a big dataset in the database.

We built a deep learning classifier in the big data service for separating noisy data from regular data. The noisy data and regular data are classified into different categories. The basic process of building the classifier is summarized as follows: 1. Import image datasets into the database. The images that belong to the same category are stored in the same collection. 2. Create a copy for each collection and then insert the label of the category into each document in the collection. Divide each collection into multiple groups and add group information into each document. 3. Export the labeled collections from the database and build a folder structure for storing exported images based on their labels and groups information. 4. Train and test the deep learning classifier using exported data and conduct NFCV and other validations. If the classification accuracy is acceptable, the trained classifier and validation results are imported into the database as separate collections. 5. When an image dataset is imported into the database first time, the classifier can be called to classify each image in the dataset to separate noisy data into separate categories. Multiple classifiers can be trained for filtering different domain specific noisy data.

III. A Deep Learning Classifier for Data Quality Improvement

In this section, we describe the design of a deep learning classifier for classifying diffraction images into three categories: *cells*, *fractured cells*, and *debris*. We also discuss the validation of the classifier and compare the classification accuracy to that of an SVM based classifier.

A. Deep Learning

Over the past few years, deep learning has become one of the fastest-growing and most exciting methods in machine learning. It offers powerful capability for solving a range of problems. Deep learning breakthroughs range from almost halving the error rate for image based object recognition [2] to defeating a low-level professional Go game players for the first time in late 2015 [18]. Go game capability was improved further to defeat a top professional player in March of 2016 [19]. A deep learning neural network is a network with multiple hidden layers. An observation such as an image in deep learning can be represented in many ways such as a vector of intensity values per pixel, or in a more abstract way as a set of edges, regions of particular shape, etc. One of the promises of deep learning is replacing handcrafted features with efficient algorithms for unsupervised or semi-supervised feature learning and hierarchical feature extraction [20].

Various deep learning architectures such as convolutional deep neural networks, deep belief networks and recurrent neural networks have been applied to fields like computer vision, automatic speech recognition, natural language processing, audio recognition and bioinformatics where they have been shown to produce state-of-the-art results on various tasks. Convolutional Neural Network (CNN) is the most widely used deep learning network for image classification. AlexNet is a CNN that has been widely cited for its success in 2012 Large Scale Visual Recognition Challenge (ILSVR2012). Since then, many more sophisticated and deeper CNNs have

been proposed for image classification in ILSVR such as VGG [21] and GoogLeNet [22].

After experimenting with several different deep learning architectures, we selected AlexNet model which is implemented in Caffe to build the deep learning classifier for the classification of diffraction images. AlexNet includes 5 convolutional layers and multiple max-polling layers in addition to 3 fully connected layers. The output of the last layer is fed to a 1000-way softmax to produce a distribution over 1000 classes [2]. Each convolutional layer filters every channel (i.e. red, green, and blue) of the input image with multiple kernels. For example, the first layer has 96 kernels and the fifth one has 256 kernels, and each fully-connected layer has 4096 neurons [2]. Due to the large number of features used in a deep learning, the volume of the training dataset required for a deep learning is also very large. We trained AlexNet with 1.2 million images.

B. Dataset

The fringe patterns of the three categories of diffraction images are different. The diffraction image of a cell contains lots of normal speckle patterns, the diffraction image of a fractured cell consists of significant strip patterns, and the diffraction image of the debris generally includes small number of large diffuse speckle patterns. We label the three categories of diffraction images as *cells* for cells, *strips* for fractured cells, and *debris* for cell debris. Fig. 2 shows a sample diffraction image for each category.

We acquired many diffraction images for the three categories of particles using p-DIFC. We selected a total 7519 diffraction images. Each diffraction image was manually inspected and labeled for its category: *cells*, *strips*, and *debris*. The initial image data set includes 2232 images of cells, 3642 debris, and 1645 fractured cells. Since each image was manually labeled, some of the diffraction images could be incorrectly labeled.

We experimented with several different deep learning neural architectures including VGG [21] and GoogLeNet [22], but the classification accuracy these architecture was not high. However, the preliminary experiment on AlexNet implemented in Caffe had fairly high classification accuracy. Therefore, we built the classifier based on AlexNet and Caffe. Based on the targeted application of diffraction images, we initially made some minor changes of the architecture of AlexNet such as changing the output of the last fully connected layer, adding and removing some convolutional layers. However, neither the training performance nor the classification accuracy improved. To avoid introducing any bugs via our changes, we finally decided to keep the net architecture as is.

The original AlexNet was trained with 1.2 millions images for 1000 categories. The raw diffraction images we collected are only 7519, which are not large enough for training AlexNet for classifying the three categories of diffraction images. Therefore, we need an approach for producing large amount of training data.

C. Preparing the Training Data

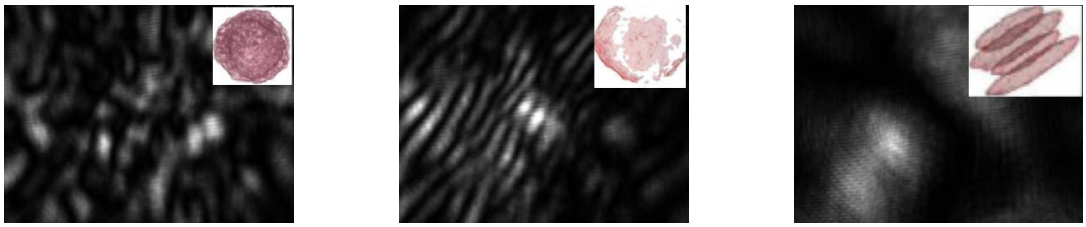
We developed a large number of diffraction images through pooling of raw images. A raw diffraction image, which is of size 640×480 pixels, is downsampled into a small image of size 227×227 using pooling. Multiple smaller size images are produced from a raw image with different pooling configurations such as max pooling and average pooling [23]. We apply different pooling window sizes and different stride of the sliding to the same image. Since the size of the small image is 227×227 pixels, and the size of an original image is 640×480 pixels, we need to resize the original image into a square. In this experiment, we cut three different sizes of squares from an image, which are 455×455 pixels, 456×456 pixels, and 457×457 pixels. The pooling is applied to the three squares. Thee pooling windows, which are 3×3 pixels, 4×4 pixels and 5×5 pixels, are applied for pooling square 455×455 pixels, 456×456 pixels, and 457×457 pixels, respectively. The stride distance of moving the pooling window is set at 2 pixels. The size of the image output from the pooling is $s \times s$ pixels, and $s = (x - m)/c + 1$, where $x \times x$ pixels are the size of the input image of the pooling, $m \times m$ pixels are the size of the pooling window, and c is stride distance. For example, if the input image is 455×455 pixels, pooling window is 3×3 pixels, and stride distance is 2, then the size of the output image is 227×227 pixels. The pooling steps are summarized below:

1) For each diffraction image, select position (10,10) of the image as position (0,0) for the new cropped images: 455×455 pixels, 456×456 pixels, and 457×457 pixels.
2) Move the cropping position from (10,10) to (10+h, 10) to crop another three square images of size 455×455 pixels, 456×456 pixels, and 457×457 pixels; where h is 10 for normal cell images, 20 for debris, and 5 for strips. Continue this step 16 times for normal cell images, 8 times for debris, and 32 times for strips. Therefore, each original cell image produces 48 different square images, each debris image produces 24 square images, an each strips image produces 96 square images. Around 100,000 images are produced for each category from the original diffraction images.
3) Pooling window 3×3 pixels is applied to 455×455 pixels images, 4×4 pixels window is applied to 456×456 pixels images, and 5×5 pixels window is applied to 457×457 pixels images. Each square image is downsampled into a 227×227 image after pooling.
4) Each small image's label is same as its parent.

Fig. 3 shows a comparison between a raw diffraction image and the corresponding downsampled image. It is easy to see that the fringe patterns in a raw image are well preserved in its pooled image. Three different pooling functions including average-pooling, max-pooling and min-pooling were used, but each dataset used only one pooling function.

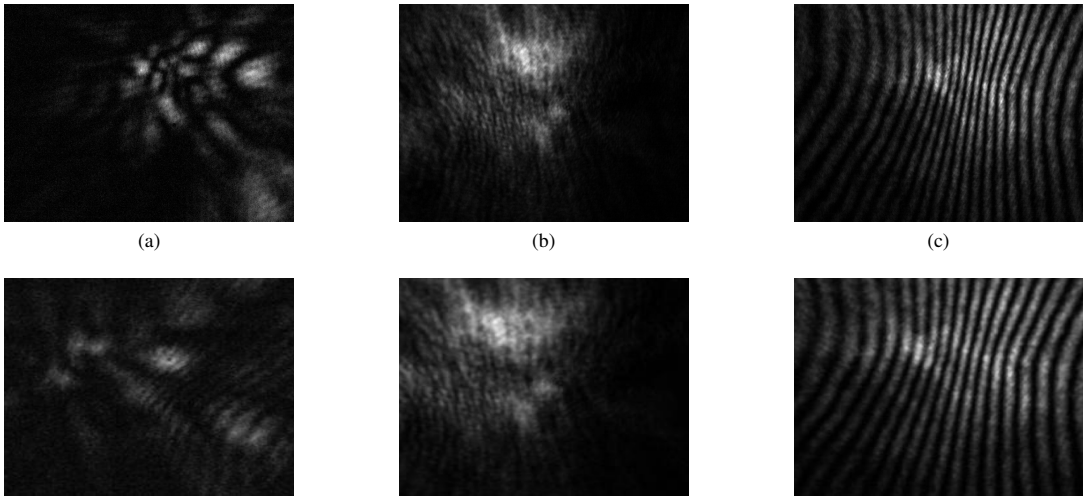
D. Experiment Results

All deep learning experiments were conducted on the pooled images. The three categories of images are stored in three



(a) (b) (c)

Fig. 2. Sample p-DIFC acquired diffraction image of (a) a viable cell with intact structures, (b) a ghost cell body or aggregated spherical particles, and (c) the cell debris or small particles. The top right corner of each image shows the corresponding 3D structure.



(a) (b) (c)

(d) (e) (f)

Fig. 3. The raw diffraction image and its pooled image: (a) a cell and (d) its pooled image, (b) the debris and (e) its pooled image, (c) a fractured cell and (f) its pooled image.

different folders, and then pooling is applied to each image to produce and label around 100,000 small images for each category. The small images were stored in three folders according to their labels/categories. 8FCV and confusion matrix were used to validate classification results. In each folder, the small images were divided into 8 almost equal groups. Among the 8 groups of data, 6 groups were used for training, another 1 group was used for validation, and the remaining 1 group was used for testing. The classifier is trained, validated, and tested with the training, validation, and test data, respectively. The process is repeated 8 times with each group taking turn to serve exactly once as validation data and test data.

Fig. 4 shows the 8FCV result of the classification based on the dataset generated using average-pooling, where **Img.** represents the number of total images, **Class** means the percentage of the classification for the category of the row, every three rows in the table are a group for the 8FCV. The average classification accuracy for cells is 85.7%. We checked the confusion matrix as shown in Table I, we found near 10% cells were incorrectly classified as debris, and 4.5% were incorrectly classified as strips. As we know, the difference between the textual patterns in diffraction images of cells and debris is the size of speckles. The debris normally includes large diffuse speckles. However, the average-pooling would decrease the difference between the normal cells and debris, which could be a reason that more cells were classified as debris. The 8FCV result and confusion matrix confirm the deep learning classifier is effective for classifying the three categories of diffraction images. However, the dataset created using average-pooling could be improved. Therefore, we also experimented with max-pooling and min-pooling for pooling the diffraction images to produce training data.

TABLE I
A CONFUSION MATRIX OF AN AVERAGE-POOLING DATA SET

	Cells	**Debris**	**Strips**
Cells	0.857	0.098	0.045
Debris	0.006	0.987	0.006
Strips	0.005	0.052	0.943

The 8FCV result of the classification based on the dataset created using max-pooling is almost identical to the result shown in Fig. 4. The average classification accuracy of cells is 87.9%, of debris is 98.5%, and of strips is 94.6%. However, the 8FCV result of the classification based on the dataset created

	Cell		Debris		Strip	
	Img.	Class	Img.	Class	Img.	Class
Cells	11226	86.34	1242	9.55	534	4.10
Debris	17	0.14	11688	99.64	25	0.21
Strips	0	0	574	5.64	9602	94.36
Cells	11156	85.80	1149	8.83	697	5.3605
Debris	52	0.443	11623	99.088	55	0.4689
Strips	0	0	388	3.813	9788	96.187
Cells	10979	84.441	1394	10.721	629	4.838
Debris	59	0.503	11641	99.241	30	0.256
Strips	20	0.197	515	5.061	9641	94.743
Cells	11020	84.756	1360	10.460	622	4.784
Debris	74	0.631	11595	98.849	61	0.52
Strips	49	0.482	452	4.442	9675	95.077
Cells	11367	87.425	1014	7.799	621	4.776
Debris	94	0.801	11577	98.696	59	0.503
Strips	32	0.314	496	4.874	9648	94.811
Cells	11747	90.348	952	7.322	303	2.330
Debris	129	1.1	11486	97.92	115	0.980
Strips	76	0.747	443	4.353	9657	94.9
Cells	10977	84.425	1495	11.498	530	4.076
Debris	63	0.537	11550	98.465	117	0.997
Strips	109	1.071	605	5.945	9462	92.983
Cells	10678	82.126	1598	12.29	726	5.584
Debris	100	0.853	11501	98.048	129	1.1
Strips	148	1.454	758	7.449	9270	91.097

Fig. 4. 8FCV result of a classification experiment.

TABLE II
A CONFUSION MATRIX OF A MIN-POOLING DATA SET

	Cells	Debris	Strips
Cells	0.935	0.036	0.030
Debris	0.024	0.961	0.015
Strips	0.023	0.044	0.933

using min-pooling is much better. The min-pooling function chooses the minimal value of the slide window to represent the whole window in the new image. The average classification accuracy of cells improved to 93.5%, of debris to 96.1% and of fractured cells to 93.3%. The confusion matrix is shown in Table II.

E. Validation of the Classifier

The category of each raw diffraction image is manually labeled, and a small image pooled from a raw image is labeled same as the raw image. It is not a problem when the fringe pattern in the image significantly shows its difference from other categories of images. However, the difference of the fringe patterns between some diffraction images is not easily detected by human eye. For example, the difference between the fringe patterns of the diffraction image of a viable cell with intact structure and the cell whose nucleus has been slightly broken could be very small. In this case, the labeling of the image is totally based on experience. Therefore, it is necessary to rigorously validate the classifier that was trained on the dataset that is manually labeled. The validation will answer two questions: (1) How well would the classifier classify diffraction images that are only slightly different in 3D morphology parameters? There is no way to know whether the sample used for taking measured diffraction images for the training and testing include particles that are only slightly different in 3D morphology parameters. Even it includes these particles, they could be incorrectly labeled since they are difficult to be differentiated. (2) Are the small images downsmapled from raw images good enough to represent the raw images? We will check how well the small images are correctly classified. The idea of the validation is to produce a set of calculated diffraction images using ADDA and then test the classifier through classifying the small images downsampled from the raw images using a pooling technique.

The validation procedure is summarized below:

1) Select 10 different types of viable cells with intact structures and take the confocal image sections for each cell.
2) Build the 3D morphology parameters of each cell based on its 3D structure that is reconstructed from the processed confocal image sections.
3) Conduct an ADDA calculation based on the 3D morphology parameters of a cell and produce diffraction image on different polarizations using a ray tracing tool. The diffraction images are labeled as *cells*.
4) Modify the 3D structure of a cell such as removing portion of its nucleus using Matlab to create new morphology parameters for producing new set of diffraction images. The diffraction images are labeled as *strips*. We make a series of changes of the 3D structures, which include some slight change and some significant changes. The process is repeated for all viable cells.
5) Calculate a group of diffraction images with some small particles using ADDA and ray tracing, and then label the images calculated from the particles as *debris*.
6) A group of small calculated diffraction images are produced Using the pooling technique and each small image is labeled as its raw image. The small images is used for testing the deep learning classifier.

We found many of the calculated images that were labeled for different categories were difficult to be differentiated by human eye. We produced 20 viable cell images and 20 debris images in different polarizations, and produced 30 diffraction images from modified 3D structures. A total of 560 small images were produced using the pooling technique with max-pooling function. The classification accuracy of the trained deep learning classifier for each category of the 560 images is 100% for cells, 100% for debris, and around 92% for strips. About 6% of strips were incorrectly classified as cells, and 2% were incorrectly classified as debris. The validation result show that the deep learning classifier performs well.

F. An SVM Classifier

Before building the deep learning classifie, we had built an SVM classifier for the same set of diffraction images. Each image is converted into a set of GLCM feature values for training and testing. The classification accuracy for the three categories of the images on the same dataset used for the deep learning classifier is as low as 65% [24], which is significantly lower than that of the deep learning classifier. However, if the dataset is preselected using image processing algorithms and other machine learning algorithms, the classification accuracy could be as high as 90% [12]. However, the preselection of images is very complex, and it is only useful for a specific application. The classification accuracy of the deep learning classifier is much higher than the SVM classifier. The incompatible size of the raw diffraction images for the classifier, the limited number of diffraction images, and computational demands of deep learning could cause difficulties for building an effective deep learning classifier. SVM is still an useful supplementary tool for developing the automated classification.

G. Discussion

Building a classifier for classifying data to achieve high accuracy is not trivial. Selecting an appropriate deep learning neural net architecture is a tedious experimental process. We have investigated several different net architectures including VGG, GoogLeNet, and self developed nets in different deep learning framework including Caffe, TensorFlow, and MxNet. The classification accuracy for the three categories of diffraction images in different architectures was ranged from 60% in GoogLeNet to over 90% in AlexNet using the same dataset. This might be due to limited volume of training data. Another difficulty is how to get a large enough training data. It is a common problem and also might be the most significant problem that many deep learning projects face.

It is extremely important to share data with the research community and this is the prime motivation for developing the big data service for biomedical image data. We have taken more than one million diffraction images using p-DIFC, and produced calculated diffraction images using scientific modeling tools. Downsampling and other techniques have been used for producing more images. Since the training data were manually labeled and images were downsampled, it is necessary to rigorously validate the classification results. We calculated a set of perfect diffraction images using many different 3D structures to test the classification accuracy. The validation approach would detect problems that cannot be found by NFCV and confusion matrix. Our experimental results show that the classification accuracy of the deep learning classifier is much higher than the SVM classifier.

IV. Related Work

Quality of big data could greatly impact the value extraction from big data. Poor quality data could cause serious problems such as wrong prediction or low accuracy of the classification. The quality attributes of big data such as availability, usability and reliability have been well defined in some publications [4] [25]. Gao *et al.* have given an overview of the issues, challenges and tools of validation and quality assurance of big data [26], where they defined big data quality assurance as the study and application of quality assurance techniques and tools to ensure the quality attributes of big data. Although general techniques and tools were developed for quality assurance of big data, much more work are on the quality assurance of domain specific big data such as health care management data, social media data and finance data.

There is a large body of work on the evaluation of the veracity of web sources such as the evaluation based on hyperlinks and browsing history or the factual information provided by the source [27], and evaluation based on the relationships between web sources and their information [28]. Finding the duplicate information from different data sources is an important task of quality assurance of big data. Machine learning algorithms such as Gradient Boosted Decision Tree (GBDT) are used for detecting data duplication [29]. Data filtering is an approach for quality assurance of big data through removing bad data from data sources. For example, Apache Samza [30], which is a distributed stream processing framework, has been adopted for finding bad data [31]. In this paper, we proposed a deep learning approach for automated classification of big data to enable selecting desired data from large repositories that may include lots of undesired data. The undesired data could be due to incorrect labeling and is known as class label noise. The impact of the class noise can be iteratively reduced in our proposed approach through multiple rounds of selection.

Deep learning researchers need to make a trade-off between using better deep learning architectures and using more training data when they look for a deep learning based classification solution [32]. However, a better deep learning architecture might not be feasible with limited amount of training data. Using large training data is a more feasible approach. For example, the original AlexNet was trained with 1.2 million images, and the classifier for classifying the three categories of diffraction images requires over 100,000 diffraction images for each category. However, many domain specific applications do not have enough data for the deep learning training.

Producing high quality artificial training data based on original data is a good practice for enhancing the training dataset for deep learning. Each domain specific application can produce artificial data according to the domain models using ADDA for producing diffraction images of cells. Sampling from a large image is also a routine approach for producing image data [33] [34]. In this paper, different approaches for producing large amount of training data are described and systematically validated. Generative models were proposed recently for producing artificial data using deep learning techniques [35]. Although large amount of initial data are required to produce artificial data using generative models, it is a promising technique for producing a large amount high quality artificial data.

V. Summary and Future Work

Providing a big data service for accessing high quality data is an important task in big data research and applications. In this paper, we described the development of a big data service which provides a means for accessing large-scale data in a flexible manner. The most important feature of the big data service is the component that enhances the data quality by separating noisy data from big data using a deep learning classifier. We discussed the approach to classifying three categories of diffraction images, which include two categories of noisy data. The deep learning classifier was built on AlexNet and Caffe, and was trained by using more than 300,000 diffraction images. The latter were generated from raw data using downsammpling techniques. The quality of the data can be iteratively improved though multiple rounds of separation of noisy data. The classifier and its classification accuracy are validated through classifying the calculated diffraction images that precisely model the features for the classification. The experimental results show that the proposed deep learning approach is effective for improving the quality of big data. Our approach can easily be adapted for automated data quality improvement of other domain-specific big data applications.

Acknowledgment

The authors would thank Jiabin Wang and Min Zhang at East Carolina University for assistance of the experiment. This research is supported in part by grants #1262933 and #1560037 from the National Science Foundation. We gratefully acknowledge the support of NVIDIA Corporation for the donation of the Tesla K40 GPU used in this research.

References

[1] V. Gudivada, R. Raeza-Yates, and V. Raghavan, "Big data: Promises and problems," *IEEE Computer*, vol. 48, no. 3, pp. 20–23, 2015.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., 2012, pp. 1097–1105.

[3] O. Russakovsky, J. Deng, H. Su, and etc., "Imagenet lsvrc 2012 training set (object detection)," 2012. [Online]. Available: http://image-net.org/challenges/LSVRC/2012

[4] L. Cai and Y. Zhu, "The challenges of data quality and data quality assessment in the big data era," *Data Science Journal*, vol. 14:2, pp. 1–10, 2015.

[5] Y. Bengio, "Learning deep architectures for ai," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[6] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121–167, Jan. 1998.

[7] K. Jacobs, J. Lu, and X. Hu, "Development of a diffraction imaging flow cytometer," *Opt. Lett.*, vol. 34, no. 19, p. 29852987, 2009.

[8] J. Ding, X. Hu, and V. Gudivada, "A machine learning based framework for verification and validation of massive scale image data," *IEEE Transactions on Big Data*, vol. DOI: 10.1109/TBDATA.2017.2680460, March 2017.

[9] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011.

[10] K. Dong, Y. Feng, K. Jacobs, J. Lu, R. Brock *et al.*, "Label-free classification of cultured cells through diffraction imaging," *Biomed. Opt. Express*, vol. 2, no. 6, p. 17171726, 2011.

[11] Y. Feng, N. Zhang, K. Jacobs, W. Jiang, L. Yang *et al.*, "Polarization imaging and classification of jurkat t and ramos b cells using a flow cytometer," *Cytometry A*, vol. 85, no. 11, pp. 817–826, 2014.

[12] J. Zhang, Y. Feng, M. S. Moran, J. Lu, L. Yang *et al.*, "Analysis of cellular objects through diffraction images acquired by flow cytometry," *Opt. Express*, vol. 21, no. 21, pp. 24 819–24 828, 2013.

[13] S. K. Thati, J. Ding, D. Zhang, and X. Hu, "Feature selection and analysis of diffraction images," in *4th IEEE Intl. Workshop on Information Assurance*, Vancouver, Canada, August 2015.

[14] R. Haralick, "On a texture-context feature extraction algorithm for remotely sensed imagery," in *Proceedings of the IEEE Computer Society Conference on Decision and Control*, Gainesville, FL, Dec. 1971, pp. 650–657.

[15] M. Yurkin and A. Hoekstra. (2014) User manual for the discrete dipole approximation code adda 1.3b4. [Online]. Available: https://github.com/adda-team/adda/tree/master/doc

[16] R. Pan, Y. Feng, Y. Sa, J. Lu, K. Jacobs, and X. Hu, "Analysis of diffraction imaging in non-conjugate configurations," *Opt. Express*, vol. 22, no. 25, pp. 31 568–31 574, 2014.

[17] "Studio3t." [Online]. Available: https://studio3t.com

[18] E. Gibney, "Google ai algorithm masters ancient game of go," *Nature*, vol. 529, pp. 445–446, Jan. 2016.

[19] T. Chouard, "The go files: Ai computer clinches victory against go champion," *Nature*, vol. doi:10.1038/nature.2016.19553, March 2016.

[20] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, Aug 2013.

[21] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014. [Online]. Available: http://arxiv.org/abs/1409.1556

[22] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *CoRR*, vol. abs/1512.00567, 2015. [Online]. Available: http://arxiv.org/abs/1512.00567

[23] (2017, Jan.) Deep learning tutorial. [Online]. Available: http://deeplearning.net/tutorial/lenet.html

[24] J. Wang, "Automated classification of massive scale image data," Master Thesis, East Carolina University, Nov. 2016.

[25] D. Rao, V. Gudivada, and V. Raghavan, "Data quality issues in big data," in *IEEE International Conference on Big Data Workshop on Data Quality*. Santa Clara, California: IEEE Computer Society, Oct 2015, pp. 2654–2660.

[26] J. Gao, C. Xie, and C. Tao, "Big data validation and quality assurance – issuses, challenges, and needs," in *2016 IEEE Symposium on Service-Oriented System Engineering (SOSE)*, March 2016, pp. 433–441.

[27] X. Dong, E. Gabrilovich, K. Murphy, V. Dang, W. Horn, C. Lugaresi, S. Sun, and W. Zhang, "Knowledge-based trust: Estimating the trustworthiness of web sources," *Proc. VLDB Endow.*, vol. 8, no. 9, pp. 938–949, May 2015.

[28] X. Yin, J. Han, and P. S. Yu, "Truth discovery with multiple conflicting information providers on the web," *IEEE Trans. on Knowl. and Data Eng.*, vol. 20, no. 6, pp. 796–808, Jun. 2008.

[29] C. H. Wu and Y. Song, "Robust and distributed web-scale near-dup document conflation in microsoft academic service," in *2015 IEEE International Conference on Big Data (Big Data)*, Oct. 2015, pp. 2606–2611.

[30] (2016, Nov.) Apache samza. [Online]. Available: http://samza.apache.org/

[31] S. Kamburugamuve and G. Fox, "Survey of distributed stream processing," 2016.

[32] H.-C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers, "Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning," *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1285–1298, 2016.

[33] D. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Mitosis detection in breast cancer histology images with deep neural networks," in *Intl. Conf. on Medical Image Computing and Computer-assisted Intervention*, 2013, pp. 411–418.

[34] B. Dong, L. Shao, M. D. Costa, O. Bandmann, and A. F. Frangi, "Deep learning for automatic cell detection in wide-field microscopy zebrafish images," in *2015 IEEE 12th Intl. Symposium on Biomedical Imaging (ISBI)*, April 2015, pp. 772–776.

[35] (2017, Jan.) Open ai: Generative models. [Online]. Available: https://openai.com/blog/generative-models/