Online processing of microscopic images

Toma Rončević
University of Split, University Department of Professional Studies, Split, Croatia
roncevic@oss.unist.hr

Jelena Vidović
University of Vienna, Department of Palaeontology, Vienna, Austria
vidovic.jelena@gmail.com

Marina Rodić
University of Split, University Department of Professional Studies, Split, Croatia
mrodić@oss.unist.hr

Abstract. In this paper we present the concept and prototype for online automatic processing of microscopic images. The idea behind the system is based on the needs of different scientific and professional areas for automation of image processing. There are many tasks in these areas that can take advantage of computer vision (CV) and machine learning (ML) techniques for image segmentation and classification. Advances in ML area of deep neural networks enable us to use the same models for many different recognition tasks. The problem is that image processing usually requires particular skills in CV and ML for successfully building these models. With this system, all models would be supervised by independent ML experts while other users could set up systems through simplistic wizards. Another problem is obtaining and preparing large quantities of data for training and testing CV and ML models. Manual sampling and labelling of large quantities of images is necessary for building good automatic models. This problem can be mitigated by online collaboration between interested parties through sharing image databases.

Key words: online, microscopic, image processing

1. Introduction

Image processing is one of most difficult problems in artificial intelligence field. Specialized computer vision (CV in further text) subfield has developed with idea of enabling software to mimic the different phenomena of human vision. From early on, scientist understood that most of mechanisms can’t be clearly described and programmed and tools from field of machine learning (ML in further text) are becoming most successful approach for many computer vision tasks. Even simple tasks as discriminating between images of cats and dogs can be rather involving but can be solved with high accuracy using machine learning models [1]. Similar models can be used for processing of scientific images that are usually simpler since we can control some aspects of image acquiring (e.g. perspective and lighting). Yet, these models usually require large quantity of manually labelled images that are expensive to produce. In this paper we present the concept and prototype for online automatic processing of microscopic images.

During last 50 years, image processing was moving from manual feature construction and rules-based evaluation towards automatic feature discovery and evaluation using approaches from ML field. Early CV was focused on using filters and rules for specific tasks. This approach had little success since digital images are very complex source of information where single pixels have no real meaning while meaningful image segments are hard to extract and
describe. Use of complex ML models (like neural networks) was also limited both by computing power, limitations of current ML models and quantity of available data. Over the last several years, use of graphical processing units for building ML models has significantly increased in conjunction with some theoretical breakthroughs in deep learning. Currently deep neural networks (DNN if further text) are dominating approach in domain of image processing as well as in some other domains like natural language processing. This approach allows researchers to train “end-to-end” DNN architecture for image processing tasks with very little human intervention. These advances resulted in different online services that allow us train and use DNN models but they usually require expert knowledge in both programming and ML field since they usually offer only raw computing power. Proposed online system is an attempt to minimize required expert knowledge for automatic processing of microscopic image. In addition, online system could mitigate labelled data bottleneck through sharing of datasets and/or models for standardized tasks. We have developed a prototype for specific problem of recognizing Foraminifera species. Foraminiferal species are morphospecies, defined according to their external morphological characteristics, primarily by wall structure, chamber and test shape, and the position of the aperture. Classification was developed in collaboration with micropaleontologists, following generic classification of Loeblich and Tappan [2] and Cimerman and Langer [3].

In this paper we describe typical DNN architecture for image classification task (section 2) and architecture of possible online service for building, using and sharing of such models between scientists in other fields (section 3). As use-case we present a prototype for microscopic images segmentation and classification of marine microorganisms (Foraminifera).

2. Image processing with deep neural networks

In this section we describe typical deep neural network (DNN in further text) approach for image classification task. Only recently some important research breakthroughs were made that enabled to train them effectively for many different image processing problems. We begin with digital image representation and typical low-level operations, then we describe feed-forward neural network and in the end we describe deep neural networks with convolutional and dense layers for image classification task.

Digital images are represented by a grid of pixels, each pixel describing intensity of image in that point. Pixels form three matrices (Figure 1) that separately describe red, green and blue brightness (sometimes called channels). Other representations of image are possible, but this is most commonly used with convolutional neural networks since they can learn any other representation directly from RGB channels.

![Figure 1](image.jpg)

**Figure 1** Representation of an image as separate RGB channels. Intensities of single pixels are limited to range 0-255. Each pixel is represented with 3 intensities.

Since individual pixels are usually meaningless it is hard to use them as predictors for simpler ML models. For this reason researchers have constructed different methods for extracting meaningful features from images. Besides global image statistics, different feature descriptors...
have been proposed, like SIFT [4] or SURF [5] descriptors. These descriptors are applied at points of interest in image and can be used both for recognising particular images, objects or classes of objects. Another solution is to convolve image with features (filters) of interest to obtain statistics of feature presence or to obtain dense maps of features. Both solutions imply manual creation and selection of features appropriate for particular task. Also, these operations are expensive and usually only the beginning of some image processing task and some model still has to be built to operate on results of this operation.

Feed forward neural networks are one possible ML model that could theoretically provide “end-to-end” learning from raw pixels to image class and in past there has been many attempts to use them. Feed-forward neural networks (FFNN in further text) are composed of many small units (neurons) interconnected in a way to form flow of information in one direction. Basic unit is composed of weighted sum over inputs and non-linear activation function $f$ (Figure 2). Weights are part of network parameters that have to be determined through training.

![Figure 2](image1)

*Figure 2* Single neuron is composed of weights and activation function. One of the weights is usually bias that always has input 1.0. Output of the neuron is the result of activation function applied on weighted sum of inputs.

In FFNNs neurons can be organized in layers and neurons in each layer usually receive input only from neurons in previous layer. First layer is receiving input vector, computes its output and passes it to the next layer. Last layer usually implements some regression or classification model (Figure 3).

![Figure 3](image2)

*Figure 3* Two-layer feed-forward neural network with two inputs and two outputs.

In image processing first layers of DNN are often convolution layers, thus forming convolutional neural network (CNN). These layers basically consist of small FFNNs that are applied (and trained) across all pixels of single image. This operation equals to convolution of image with different filters. Highest responses to convolutions within small pixel neighbourhood are pulled and passed to next layer. This operation reduces amount of data that is processed by further layers while also enabling small spatial invariance. It was determined that these first few convolutional layers usually describe some simple image features (e.g. edges or corners) and vary little between different image domains. This opens possibility of reusing convolutional layers between different image processing tasks (transfer learning [6]). These networks usually have dozens of different hyperparameters (e.g. number of
convolutional and dense layers, convolution window width, learning parameters, etc.) that have to be determined in advance before training the final network. These hyperparameters can be determined by cross-validation but that usually requires extensive search in hyperparameters space and many training iterations. However, for large DNNs, any set of reasonable hyperparameters can usually produce acceptable model in terms of accuracy.

Training of FFNN consists of showing it samples of input-output pairs from some unknown distribution. For each pair we can calculate error $E(\text{input})$ as squared difference between output of the network and desired output or some other convenient function of error. For hidden layers we propagate error backwards according to the estimate of how much each parameter contributed to the error. This process is known as backpropagation. Estimate of parameter contribution to the error is given by gradient with respect to model parameters: $dE(\text{input})/dw$. So the algorithm for learning consists of choosing samples from true distribution that we’re trying to model, showing it to the network and minimizing the error by moving in opposite direction of error gradient. This process is known as gradient descent and there are numerous variations of this basic algorithm (stochastic gradient descent). Most common variations repeatedly iterate trough some set of input-output pairs and adjust network parameters by some fraction of the error (standard gradient descent). This can be done by using small batches (like 100 samples) to calculate more stable mean gradient. Exact training regime depends on domain but in image classification domain it is usually done on fixed training set that is prepared and labelled in advance.

Gradient descent is also used to train deep neural networks but some practical problems arise. Since most neural networks use sigmoid output function, error that is backpropagated can vanish because of limited computer precision and first network layers learn slowly or don’t learn at all. This problem was addressed in [7] and ReLU activation was proposed as solution. ReLU activation has simple form:

$$f(x)=\left\{ \begin{array}{ll} x & \text{for } x > 0 \\ 0 & \text{for } x \leq 0 \end{array} \right.$$ (1)

ReLU is still nonlinear activation function but its gradient is conserved better than with sigmoid or tanh functions. Another problem is that these deep models have high variance and thus can easily overfit the training data and perform poorly on new data points. This problem was addressed in different ways in the past. Usually this required to carefully choose model size and hyperparameters using cross-validation. Today, advice is to use model as large as time and hardware constraints permit and rely on other means to address overfitting. Most common ways are to add additional constraints to network parameters (e.g. regularization). For some domains (like image recognition) generation of additional training samples by distorting original samples can help also (data augmentation). Another method (dropout) to prevent overfitting was proposed in [8]. Core idea is to randomly disable parts of the network during training. This training regime effectively results in an ensemble of smaller networks that share some parameters. During prediction this network has much lower variance and this usually results in much better final model. In the end, aggregating several of these final models can also be aggregated in even larger, more robust model.

In conclusion, we can say that these recent findings allowed to train models more easily and with less human intervention. Another important factor that makes DNN models viable is widespread use of GPU units for execution of heavy calculations needed for training. Also, transferred learning enables us to reuse parts of the pre-trained DNN and fine-tune them for another similar problem. All of these findings can be incorporated in a system for automatic image processing with some limitations that we describe in next section.
3. Online image processing architecture

Manual processing of microscopic images can be very time consuming and repetitive. In many cases, results of such processing are summarized in statistics that describes some given sample. Good examples are identifying and counting foraminiferal species in samples of marine sediment, blood cell counting or mitosis detection on histology images. Manual processing can have serious drawbacks both in cost and accuracy (one of the problems faced by those studying foraminifera is inconsistent use of species and generic names). This leads to limited size of samples that can be processed. Automatic processing on the other hand can be much faster but it can be expensive to develop a system for smaller specific domains. As it was described in previous section, DNNs offer couple of ways for reusing models in similar domains. While such models could be less accurate, if trained on small number of samples, they can compensate it with size of the test sample if task permits it. Centralizing and sharing models and image datasets could also improve accuracy for more common tasks. Basic architecture would be composed of web API accessible by browser based or desktop client application, image storage, models storage and computing instances for training and applying models (Figure 4).

![Figure 4 Proposed online image processing architecture.](image)

Online nature of this architecture is essential for collaboration between users. Sharing trained models can be done by creating ensembles for specific tasks or by transferred learning. Sharing of datasets for training models could resolve one of the most serious bottle-neck of ML systems: scarcity of data. This collaboration could also lead to establishing standards for image acquisition and creation of new benchmark datasets. This wouldn’t compromise privacy of datasets, since they would be only shared for training new models as extra data. Hardware requirements for such system in production would include GPU computing units to make CNN/DNN training viable. Since exact computing needs can be hard to anticipate in advance, we propose to rely on 3rd party resources like Amazon’s GPU EC2 instances and S3 storage [9]. Since most of computing power is required only during model training, these instances could be allocated only occasionally while most of computing for applying models can be done on dedicated machines. Limitation of this approach is the need to transfer large quantities of data between server and clients. While this still remains an inconvenience, today this is not a large problem and can be mitigated with better client applications.

As a proof of concept, similar architecture was developed consisting of a single server machine relying on ad-hoc segmentation and simple linear model for classification task. Main goal was to build intuitive interface and workflow, while use of DNN models and managing more realistic quantities of data is left for future implementation. To test the usability of such interface we developed a prototype of this architecture specialized for segmentation and classification of foraminiferal species. It was developed in collaboration with micropaleontologists with no experience in ML and CV. Most of requirements were
concentrated on user interface for samples labelling and reporting results. Remained obstacle is still automatic acquisition of images since it would require specialized equipment (like in [10]). Task was to identify, classify and count foraminiferal individuals from microscopic images of marine sediment extracted from some area of interest. The results obtained in this way could be used for different purposes, like in petroleum exploration. Images were obtained by simple 5 megapixel CCD camera pointed at microscope ocular (Figure 5).

Figure 5  Microscopic image of foraminiferal specimens.

Images contained different foraminiferal specimens that were classified according to the wall type into three primary groups: agglutinated, calcareous porcelaneous and calcareous hyaline group (Figure 6). Training and testing samples (single foraminiferal individuals) were extracted automatically by ad-hoc algorithm. Extraction was done by thresholding and segmenting connected areas of appropriate size and shape. Several global shape and texture features were extracted and training and validation sets were created. For wall type classifications, linear model was trained on 100 images of single foraminiferal individuals. Other classifications were implemented but not largely tested. This simple model with relatively small training set was able to achieve above 60% accuracy.

Figure 6  Three classes of foraminiferal wall type: porcelaneous, hyaline and agglutinated.

Figure 7  Inspection/manual correction of foraminiferal classification.
Agreed workflow consisted of uploading images, sending them for automatic segmentation, manually labelling images and training initial model. After training of initial model (on some smaller quantity of data), we can replace manual labelling of new data with automatic labelling and manual inspection of these labels. We implemented a simple interface tailored for that task but concluded that it should be replaced with more flexible design for general tasks. Inspection and correction of labels automatically increases training set and better model can be trained (Figure 7). Finally, simple statistical reports were made for datasets.

4. Conclusion and further work

We presented an online microscopic images processing architecture that relies on current advancements in ML and CV fields. Major breakthroughs in DNN enable us to (re)build different models with minimum adjustments. Datasets and models sharing is central for this architecture. Simplicity of use is one of primary concerns since system is intended for use in research outside ML and CV. Online nature of this architecture also facilitates assistance of ML experts. Simple prototype was developed as proof-of-concept for this architecture while DNN models were presented as a possible solution for more realistic implementation. Currently, we’re implementing more general prototype that would rely on external services for DNN models training. Prototype will implement separate workflows for image segmentation, classification and detection tasks.

REFERENCES