Biomedical Imaging

Biomedical Image Augmentation Using Augmentor

Marcus D. Bloice [1,*], Peter Roth [2] and Andreas Holzinger [1,3]

[1] Institute for Medical Informatics, Statistics and Documentation, Medical University Graz, Austria
[2] Institute of Computer Graphics and Vision, Graz University of Technology, Austria.
[3] Institute of Interactive Systems and Data Science, Graz University of Technology, Austria.

[*] To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: Image augmentation is a frequently used technique in computer vision and has been seeing increased interest since the popularity of deep learning. Its usefulness is becoming more and more recognised due to deep neural networks requiring larger amounts of data to train, and because in certain fields, such as biomedical imaging, large amounts of labelled data are difficult to come by or expensive to produce. In biomedical imaging, features specific to this domain need to be addressed.
Results: Here we present the Augmentor software package for image augmentation. It provides a stochastic, pipeline-based approach to image augmentation with a number of features that are relevant to biomedical imaging, such as z-stack augmentation and randomised elastic distortions. The software has been designed to be highly extensible, meaning an operation that might be specific to a highly specialised task can easily be added to the library, even at runtime. Although it has been designed as a general software library, it has features that are particularly relevant to biomedical imaging and the techniques required for this domain.
Availability: Augmentor is a Python package made available under the terms of the MIT licence. Source code can be found on GitHub under https://github.com/mdbloice/Augmentor and installation is via the **pip** package manager.[*]
Contact: marcus.bloice@medunigraz.at
Supplementary information: The GitHub repository contains supplementary information, code examples, and Jupyter notebooks. Extensive documentation is hosted on *Read the Docs* under https://augmentor.readthedocs.io. For continuous integration tests see https://travis-ci.org/mdbloice/Augmentor.

1 Introduction

Image augmentation is the procedure by which an existing dataset is expanded by transforming the original dataset in order to create new data, and in such a way that the new data is label preserving. The goal is to increase the variance of the dataset, while ensuring that new data is meaningful and does not merely add unnecessary volume to the dataset. When used in a machine learning context, it can improve model generalisation, make trained models more robust to unseen data, and increase model accuracy. Image augmentation is becoming more relevant of late due to the popularity of deep neural networks that often require much data to train due to their large learning spaces. In biomedical imaging where good, labelled data can be scarce, image augmentation can be a highly useful technique when building prediction models.

Biomedical image augmentation has its own particular characteristics that make it unlike standard augmentation. Data are often in obscure formats, and image data are often

[*]A Julia version of the package, developed in parallel by Christof Stocker, is also available under https://github.com/Evizero/Augmentor.jl

time series based or z-stacked/layered. These properties make it difficult to use software that has not been written with biomedical uses cases in mind. While a number of augmentation libraries exist, and indeed several machine learning frameworks such as TensorFlow or Keras have built in rudimentary augmentation functionality, we felt no library existed that fulfilled the requirements for biomedical imaging projects satisfactorily. The Augmentor software package was written to address such issues, and these features are discussed in more detail in the Section 3.

Installation is via the pip Python package manager. Running `pip install Augmentor` from the command line will install the latest stable version from the Python Package Index (PyPI).

2 Approach

The Augmentor package uses a stochastic, pipeline-based approach where *operations* are pieced together, each with a user-defined probability of being executed on any image that passes through it. Operations in this case are the functions that are applied to the images, such as zoom, rotate, distort, and so on. Every operation has at least a probability parameter that controls the likelihood of it being applied to the current image passing through it, and most operations are parametric and allow for its degrees of freedom to be controlled. For example, the rotate operation has parameters to control the maximum left rotation and maximum right rotation in degrees. When an image passes through the pipeline and encounters this operation, the rotation angle is randomly chosen between these two values. As mentioned, every operation has an associated probability argument, which defines the probability of that operation being executed. The stochastic nature of the pipeline's output is therefore provided each operation's probability parameter as well as by the range arguments, all of which are supplied by the user. Applying a probability of 1 means that this operation is applied to every image that passes it. Once a pipeline has been constructed and fed with an image source, augmented images can effectively be sampled from it. This stochastic approach means that each time a single image passed is through a pipeline of sufficient length, it will result in a new, randomly augmented image as an output. While recent efforts have focussed on ways to learn augmentation strategies and policies (Ratner *et al.*, 2017; Cubuk *et al.*, 2018), Augmentor does not attempt to do so, as we wished to avoid requiring any lengthy learning phases for the generation of data, or the learning of augmentation parameters.

3 Features

The package was created as a solution to issues encountered by our group when working on layered laser scanning microscopy image data (Wiltgen and Bloice, 2016; Wiltgen *et al.*, 2011), and also by the requirements of the ISIC challenge, where lesion image data and their corresponding mask data are made available (see Figure 1). The following features focus on operations relevant to biomedical imaging:

- Randomised distortions can create extremely large datasets (Simard *it et al.*, 2000)
- Z-stack augmentation that can handle series, stacked, or masked image data *of arbitrary depth* (see Figure 1)
- Multi-threading for large datasets or large images

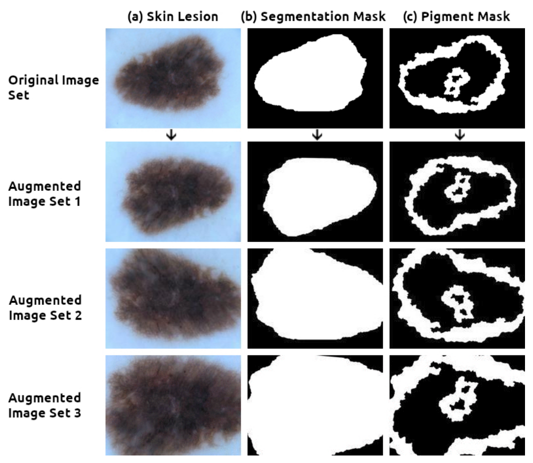

Fig. 1. Augmenting an image with multiple masks. The top row contains, from left to right, (a) the original lesion scan, (b) the segmentation mask, and (c) the pigment mask. Images can be grouped along with their masks and fed simultaneously into the pipeline, creating randomly augmented images yet ensuring the transformations are applied identically to all images within each set (Augmented sets 1, 2, and 3 above). Images from ISIC challenge dataset (Codella *et al.*, 2017; Tschandl *et al.*, 2018).

- Generators for creating datasets on the fly
- Precise control of the image augmentation pipeline

Multi-mask or z-stack image augmentation, an example of its usage can be seen in Figure 1, is particularly relevant for the biomedical domain. An arbitrarily long or list of images can be passed simultaneously through the pipeline, grouped in sets, and be identically augmented within each set. As mask ground-truth data is time consuming to create, the ability to augment this type of data is quite practical. Other use cases include time-series or time-resolved modalities, such as Digital Subtraction Angiography (DSA) image data, and cross-sectional modalities such as MRI, CT, or LSM data.

Randomised elastic distortions are likewise highly relevant for situations where data is scarce. The procedure functions by mapping a grid (which is user defined in size, e.g. 4×4) onto an original image and then applying distortions at the cross sections of this grid in random directions and with random magnitude. This process can generate large volumes of real-world feasible data due to the high number of possible combinations of parameters. See the project's GitHub repository main page for an animated example. These randomised distortions can be applied to images with mask data as well as to layered image data.

The software has been developed in Python, and many libraries exist for reading medical image formats, such as *pyLSM* for confocal laser stacks, *Astropy* for FITS times series images, the more broad *python-bioformats*, and many other packages for handling DICOM/PACS. To make it possible to handle data in these formats, images can be supplied to an augmentation pipeline using a custom array-based data-structure, as well as being able to read images directly from disk.

4 Discussion

In summary, Augmentor is a stochastic, pipeline-based image augmentation library that includes real-world relevant features for the biomedical imaging domain. We are eager for contributors and researchers working in this domain to become involved in the project: those wishing to enhance the software through the addition of new operations are welcome to make a pull request on GitHub, and others who require a specific or niche feature are welcome to make a feature request. This way biomedical-relevant features can be accumulated, which would be beneficial for the community at large.

Requirements: Windows, macOS, or Linux. Python >=2.7 or >=3.3. **Conflict of Interest**: none declared.

References

Codella, N.C.F. *et al.* (2017) Skin Lesion Analysis Toward Melanoma Detection: A Challenge at the 2017 International Symposium on Biomedical Imaging (ISBI), Hosted by the International Skin Imaging Collaboration (ISIC), *arXiv:1710.05006v3.*

Cubuk, E.D. *et al.* 2018: AutoAugment: Learning Augmentation Policies from Data, arXiv:1805.09501.

Ratner, A.J. *et al.* (2017): Learning to Compose Domain-Specific Transformations for Data Augmentation, *NIPS*, 3236–3246

Simard, P.Y. *et al.* (2003) Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis, *ICDAR*, **3**, 958–62.

Tschandl, P. *et al.* (2018) The HAM10000 Dataset: A Large Collection of Multi-Source Dermatoscopic Images of Common Pigmented Skin Lesions, *arXiv:1803.10417v2.*

Wiltgen, M. and Bloice, M.D. (2016) Automatic Interpretation of Melanocytic Images in Confocal Laser Scanning Microscopy, In *Microscopy and Analysis*, InTech Publishing, DOI: 10.5772/63404.

Wiltgen, M. *et al.* Computer-aided Diagnosis of Melanocytic Skin Tumors by use of Confocal Laser Scanning Microscopy Images, *Analytical and Quantitative Cytology and Histology*, **33(2)**, 85–100.