UTILIZING CONVOLUTIONAL NEURAL NETWORKS TO IMPROVE CERVICAL CANCER SCREENING

JONAS KEMP[1], SOPHIA SANCHEZ[1], TIMOTHY WU[2]
[1]*Department of Computer Science, Stanford School of Engineering, 353 Serra Mall*
[2]*Department of Biomedical Informatics, Stanford School of Medicine, 1265 Welch Road*
Stanford, CA 94305, USA

Cervical cancer is the third most common cancer among women worldwide, and the second most frequent cause of cancer-related death, accounting for nearly 300,000 global deaths annually. Especially in rural areas, where screening resources are limited, many women at high risk for cervical cancer receive inadequate treatment due to unique physiological features of their cervix. Specifically, cancerous lesions typically develop in the transformation zone of the cervix, but the precise location of this region is not always easily identifiable, and in some cases it may be partially concealed. As a consequence, treatments which work effectively for some women may obscure future cancerous growth in others, increasing health risks. An automated decision support tool that can accurately locate a patient's transformation zone from cervical images offers an opportunity to improve clinical decision-making regarding patient eligibility for further screening and therapy. Recent advances in deep learning have led to the creation of high-performance image classification systems in many application domains, including medical imaging. We developed an algorithm for cervical image classification, including (1) preprocessing and segmentation of labeled cervical images, where each image is in one of three classes according to the position of the transformation zone; (2) training a deep classifier using different convolutional neural architectures, with hyperparameter selection according to standard model validation procedures; and (3) evaluating the accuracy of the final system via a multi-class logarithmic loss. With an accurate and widely available classifier for cervical images, we envision personalized and cost-effective treatment regimens becoming increasingly accessible to underserved patients, ultimately improving outcomes and reducing cervical cancer mortality.

Keywords: Cervical cancer; Convolutional neural networks; Transformation zone; Image classification

1. Introduction

Cervical cancer is the third most common cancer among women worldwide, and the second most frequent cause of cancer-related death, accounting for nearly 300,000 global deaths annually[1]. More than 85% of cervical cancer incidence and 80% of mortality occur in developing countries, where women still lack access to screening and treatment. With early diagnosis and appropriate treatment, cervical cancer survival is greatly improved[2].

Cervical cancer prevention efforts have thus far focused on screening women at risk for the disease using cytology-based Papanicolaou (Pap) smears. However, most developing countries

have been unable to effectively implement comprehensive Pap smear screening programs due to the high costs associated with equipment and personnel[2]. This lack of effective screening programs in rural or low-resource settings is a key reason for higher cervical cancer mortality. Instead, alternative approaches based on colposcopy, or visual inspection of the cervix, are most frequently implemented in low-resource settings.

One of the greatest challenges in large-scale screening programs in low-resource settings is determining an appropriate method of treatment following identification of a patient with high risk for developing invasive cervical cancer. Especially in rural areas, where screening resources are limited, many women at high risk for cervical cancer receive inadequate treatment due to unique physiological features of their cervix. The transformation zone is an anatomical feature of the cervix where cell types change from columnar cells in the lower uterus (endocervix) into squamous cells that line the vaginal junction (ectocervix)[3]. During colposcopy, identifying the transformation zone is of great importance, as almost all cancerous lesions occur in this zone[3]. A patient's cervix may be classified in one of three broad categories according to the position of the transformation zone (**Fig. 1**). Those that fall under type 2 or 3, with a potentially large endocervical component, might include hidden lesions and therefore require different treatment than those in type 1.

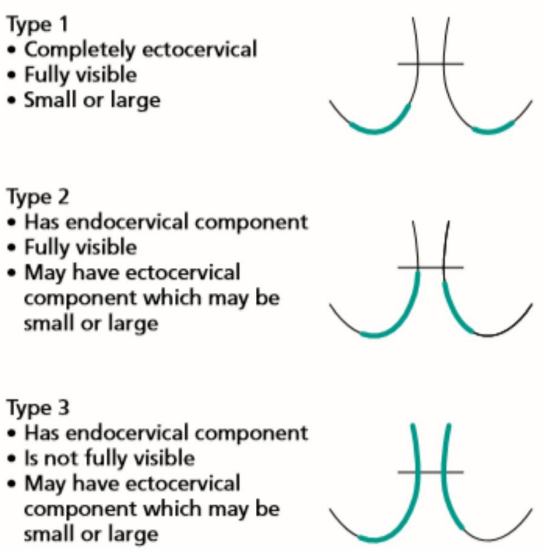

Figure 1. Description of three cervix classes, according to the position of the transformation zone[4].

Thus, efficacy of colposcopy for cervical cancer screening is highly dependent on accuracy of interpretation by the practitioner. Depending on a patient's cervix type, treatments which work effectively for some women may obscure future cancerous growth in others, increasing health risks. Here, an automated decision support tool that can accurately locate a patient's

transformation zone from cervical images offers an opportunity to improve clinical decision-making regarding patient eligibility for further screening and therapy.

Recognizing this unmet need, Intel and digital health company MobileODT recently partnered with Kaggle to sponsor a challenge centered around creating a digital tool for cervix classification. MobileODT uses mobile technology to drive portable internet-connected diagnostic systems, including a colposcope; a real-time cervical classifier would fit neatly into their system's technological niche. Our work was designed to address this gap in the existing toolset by developing a deep neural classifier that can be integrated into the MobileODT platform.

2. Related Work

Over the past five years, advances in deep learning have revolutionized computer vision, particularly in the task of image classification. Deep convolutional neural networks (CNNs) are capable of automatically extracting useful intermediate feature representations from raw image data, which have proven to be very powerful in a variety of classification tasks. In 2012, Krizhevsky et al. introduced AlexNet, the first deep CNN to achieve (by a wide margin) state-of-the-art performance on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), a classification task on a thousand classes and over a million images[5]. Since then, novel architectures developed by researchers at NYU, Oxford, Google, Microsoft, and elsewhere have consistently achieved nontrivial performance gains in ILSVRC every year. Pretrained versions of many of these models are freely available to practitioners through common deep learning frameworks such as Caffe and Keras.

These state-of-the-art technologies have already been applied successfully to a number of application areas in medical image analysis, including lesion detection, segmentation, and labeling, with promising results[6]. In particular, deep learning in medical domains is challenged by lack of large-scale annotated datasets for medical images comparable to ImageNet, without which training deep networks from scratch becomes increasingly difficult. However, recent research has demonstrated the strength of transfer learning, in which a CNN pretrained on a natural image dataset such as ImageNet is subsequently fine-tuned on a (potentially much smaller) medical image dataset, and a separate classifier is implemented in the final layers. Recent experiments with transfer learning for various medical tasks have shown strong results[7]. These results are generally superior to learning a deep network from scratch, particularly for datasets of limited size[8,9].

Although deep networks have previously been applied to the task of cervical cancer cell segmentation[10], no deep classifiers currently exist for the cervical cancer screening task proposed here. Our tool brings the cutting edge in computer vision technology to bear on this critical issue in global women's health.

3. Methods

3.1. *Image preprocessing and segmentation*

3.1.1. *Data cleaning and duplicate removal*

The first task in the development of the classifier was preprocessing and segmentation of the labeled cervical images. The initial dataset consisted of 1481 images, but a supplemental set of 6734 additional images was added during the course of the competition. Often, these additional images were of lower quality or were repeats from the same patient, and so the data required extensive filtering before use. **Table 1** summarizes the initial cleaning steps by class label.

Table 1. Summary of data set, according to image label.

	Initial Training Images	Additional Training Images	Duplicates Removed	Corrupt or Spurious Images Removed	Total Images Remaining
Type 1	250	1191	453	60	**928 (15.7%)**
Type 2	781	3567	1021	234	**3096 (52.5%)**
Type 3	450	1976	442	109	**1877 (31.8%)**
Total	1481	6734	1916	403	**5896**

During cleaning, two concerns emerged. First, the dataset included many duplicate or near-duplicate images (i.e. different images from the same patient, **Fig. 2**). Though many agreed on class label, some did not; moreover, some images were duplicated from the test set, creating a leakage issue. Second, the dataset contained several corrupt or spurious images. These images typically consisted either of empty files or non-cervical images (such as an image of a smiling woman).

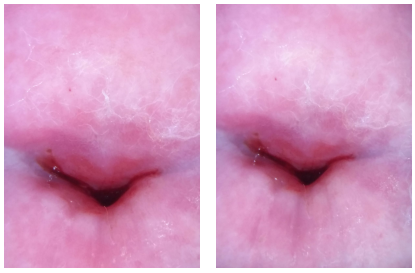

Figure 2. Near-duplicate images of the same cervix contained in the dataset.

In order to identify duplicates or near duplicates, our preprocessor employed the difference hash, or dhash, algorithm. While the md5 algorithm is the most commonly used duplicate image

detector, it compares image hashes for equivalence and therefore does not flag near-duplicate images, as even a one-pixel difference between images results in unique hashes. The dhash algorithm, in contrast, is ideally suited for both duplicate and near-duplicate image detection. For a given image, dhash reduces its size, converts to grayscale, computes the differences between adjacent pixels, and uses the result of the computation to generate a hash value. The hash value for the given image is then compared to all other images hash values in the set. If the Hamming distance[11] between two image hashes is below a certain threshold, the images are identified as equivalent. For our image hashes, we converted the hash values to longs and set our dhash threshold to 22. This threshold was selected both on the basis of a review of the literature[12], as well as through testing on a subset of our data for which duplicates and near-duplicates were manually identified.

After running the dhash algorithm, we identified and removed 1916 duplicates. For sets of duplicates where all images had the same label, we removed all but one of the set from the data. The selected image was chosen at random. (It is worth noting that in this case, duplicate removal would not be strictly required for robust CNN training, and indeed data augmentation with artificially generated near-duplicates is a common training strategy[13]. However, given that a) duplication was not uniform across patients and b) we sought to guard against accidental leakage when creating a validation split, we chose to exclude these images.) For sets of duplicates with more than one label, we removed all images in the set from the data, as the true type was uncertain. If the set of duplicates contained at least one image in the test set, we deleted all instances of the image in the training set, so as to avoid data leakage. Finally, we identified corrupt or spurious images manually for removal.

3.1.2. *Image segmentation*

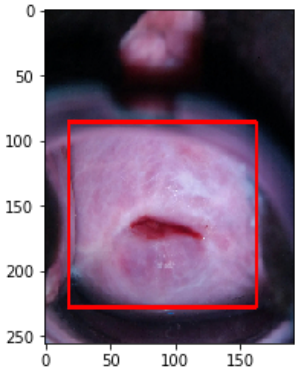

Figure 3. Initial segmentation of a cervical image to remove camera artifacts. The preprocessor crops the image, as indicated by the red square, using automatic detection of cervical landmarks.

Using the Python libraries numpy[14], cv2[15], sklearn[16], and skimage[17], we segmented the cervical images to standardize and remove unwanted camera artifacts (**Fig. 3**). Our preprocessing tools were derived from an open-source script available via Kaggle[18]. The algorithm used for cervical

image processing builds on previous work on automatic segmentation and labeling of cervical regions of anatomic interest[19], specifically cropping the image to remove circular speculum or colposcope frames, as follows:

1. The algorithm defines a circular frame, identified via cv2 as the largest contour of the image, and finds the largest inscribed rectangle within this frame.
2. For delineation of the cervix, each pixel within the rectangle is assigned two feature values: *a*, the color channel of the source image in Lab color space, and *R*, the distance of a pixel from the image center.
3. The image is separated into two clusters in the 2-D *a*-*R* feature space, and the cluster with highest *a*-mean and lowest *R*-mean is selected. The ROI is chosen as the largest connected component within the pixels associated with this cluster, and the rectangle coordinates are adjusted for final cropping.

3.2. *CNN training*

3.2.1. *Classifier architecture*

We developed our models using Keras, a high-level deep learning library for Python, running on top of TensorFlow[20,21]. Our model implements a simple multilayer perceptron (MLP) on top of an existing convolutional model pretrained on ImageNet. We experimented with two pretrained architectures during development:

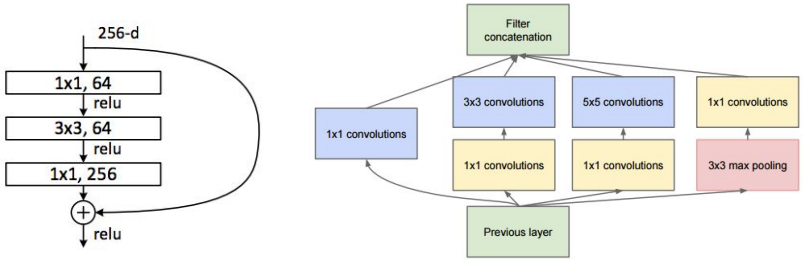

Figure 4. Diagrams showing a residual block[22] (left) and an Inception module[24] (right). These small modules are stacked many times to build the full ResNet and Inception architectures.

- *ResNet50*: Introduced by He et al. in 2015[22]. Implemented using "residual blocks" (Fig. 4), stacks of three convolutional layers with an additional shortcut connection bypassing these layers. The ResNet architecture was shown to substantially increase model depth and accuracy while reducing model complexity and difficulty of training. The version available in Keras uses 50 total weight layers, though deeper networks are possible.
- *Inception v3*: Introduced by Szegedy et al. in 2015[23], as an update to 2014's GoogLeNet. Built on an extended version of the Inception module (Fig. 4), which computes multiple convolutional filters of different sizes in parallel at each level of the model.

Both of these architectures have shown state-of-the-art performance on the ImageNet challenge, and Inception v3 was recently used as the basis of a highly accurate classifier for skin cancer[7]. For each pretrained model, we retained the convolutional layers for feature extraction, and replaced the top-level fully connected layers (trained for ImageNet classification) with a new set of fully connected layers to be trained specifically for our task. We used ReLU activations at each fully connected layer, added dropout[25] between layers for regularization, and computed a final softmax layer to output predicted probabilities for each class.

3.2.2. *Training and hyperparameters*

We trained our model to optimize cross-entropy (or multi-class logarithmic) loss, defined as follows:

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(p_{ij})$$

where we have N images and $M = 3$ classes, and each $y_{ij} = \mathbf{1}\{$observation i belongs to class $j\}$, p_{ij} = predicted probability that observation i belongs to class j. We used Adam optimization[26], a variant of stochastic gradient descent, to fit the model. Hyperparameters tuned during training included learning rate, dropout rate, and the number of fully connected layers in the top-level MLP. We also experimented with the following approaches and evaluated their effect on model performance:

- *Preprocessing:* the procedures described in section 3.1 (compared to training on raw images).
- *Data augmentation:* add a small amount of noise to the training data, in our case through random (horizontal or vertical) image flipping during training.
- *Class weighting:* increase the weight of type 1 and type 3 examples in the loss function during training, to correct for the class imbalance in the dataset (see Table 1).
- *Fine-tuning of pretrained layers:* rather than freezing the pretrained model and treating it as a fixed feature extractor, continue tuning the convolutional layers of the pretrained network at a low learning rate.
- *Using lower-level activations:* train the MLP using output from an intermediate layer of the pretrained network, rather than the final output after all convolutional layers.
- *Ensembling:* train multiple models and average their predictions.

3.3. *Model selection and evaluation*

Before training, we withheld a 10% stratified sample of the labeled data for validation. For each model, we computed loss and accuracy on the validation set at each epoch of training, and

stopped training upon observing a local minimum in the validation loss. We also analyzed confusion matrices for each model's final predictions on the validation set. From among our experiments, we manually selected a model demonstrating strong generalization (i.e. low validation loss, high validation accuracy) and little evidence of overfitting (i.e. minimal difference between train and validation statistics).

Additionally, we submitted our predictions on the blinded test set to Kaggle for evaluation. Kaggle computed the log-loss for these predictions and ranked our score on a leaderboard of over 800 teams participating in competition.

4. Results

4.1. *Model selection and hyperparameter tuning*

Between the two pretrained architectures tested, we found that ResNet50 outperformed Inception v3 on our data. **Table 2** summarizes the key hyperparameter values selected during validation:

Table 2. Final selections for key model hyperparameters.

Pretrained architecture	Number of fully connected layers	Dropout rate	Learning rate
ResNet50	4	0.25	0.001

We summarize our experimental evaluation of each of the approaches outlined in section 3.2.2 as follows:

- *Preprocessing:* We found that our preprocessing procedures had a negligible effect on model performance. However, image cropping accelerated computation speed, allowing for a more rapid training and development cycle. For this reason, and because preprocessing eliminated test leakage, guarded against accidental leakage in validation split, and improved data integrity through the elimination of images with uncertain label, we chose to work with the processed dataset.
- *Data augmentation:* We trained our final model using image flipping, as we found that it improved generalization and reduced overfitting (effectively serving as a form of regularization), and thereby increasing the robustness of the classifier.
- *Class weighting:* We did not train our final model using a weighted loss. Even modest weighting to compensate for class imbalance led to significant underfitting (**Fig. 5**).
- *Fine-tuning of pretrained layers:* Our final model froze the pretrained layers rather than fine-tuning them. In contrast with class weighting, fine-tuning quickly led to significant overfitting, even with a very low learning rate and increased regularization (**Fig. 5**).

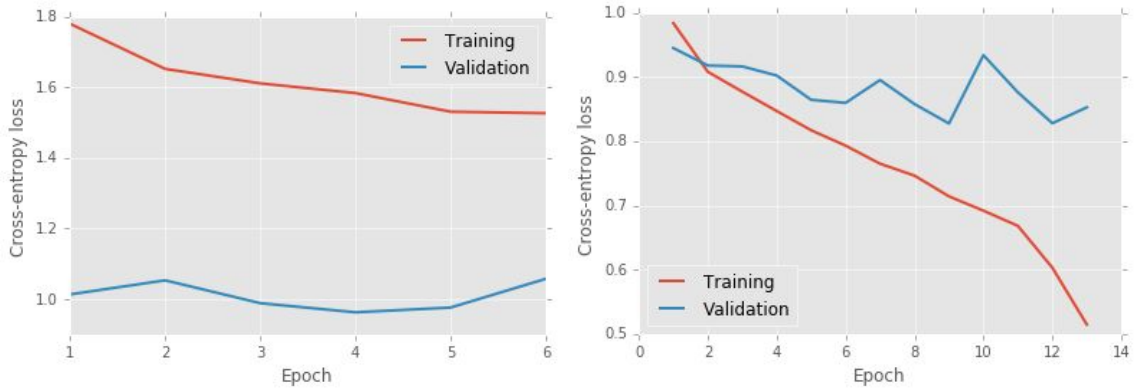

Figure 5. Sample training and validation loss plots showing underfitting when using class weighting (left) and overfitting when fine-tuning pretrained layers (right).

- *Using lower-level activations:* Our final model was trained using all pretrained layers. We attempted to train using activations from the middle of ResNet (the 6th residual block, or 18 convolutional layers), but observed slight underfitting relative to using the top-level activations.
- *Ensembling:* We created a simple ensemble in which we averaged predictions from a model trained with class weights and one trained without. We observed improved performance relative to training with class weights alone, but still underfit relative to training without class weights.

4.2. *Evaluation of final model*

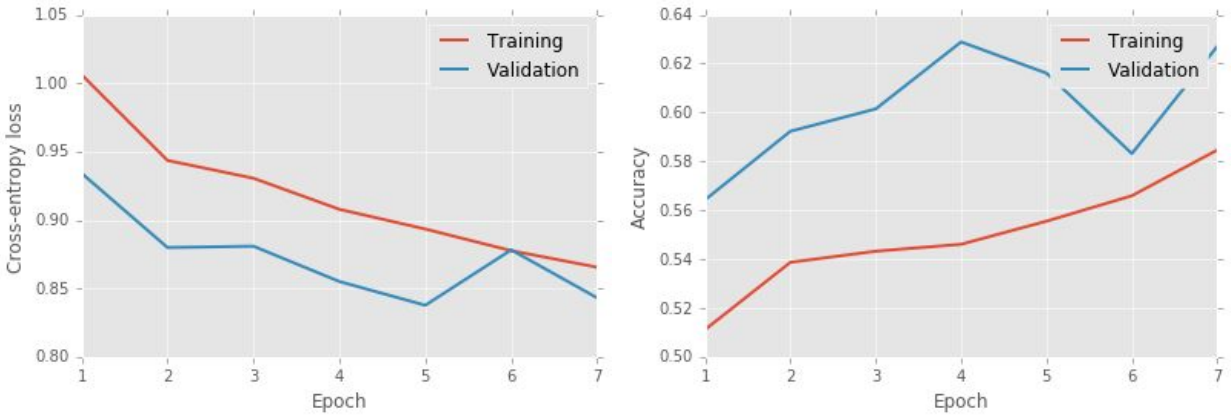

Figure 6. Loss and accuracy plots for final model.

Our final model achieved a validation loss of 0.843, our best result observed without significant overfitting. Though validation accuracy was rather unstable relative to loss (**Fig. 6**), the model's corresponding accuracy was 62.7%.

Notably, perhaps as an unfortunate consequence of ignoring class weights, most of the accuracy gains come from a strong bias towards type 2 images, the majority class. In fact, we find on the validation set that type 1 images are ignored entirely (**Fig. 7**).

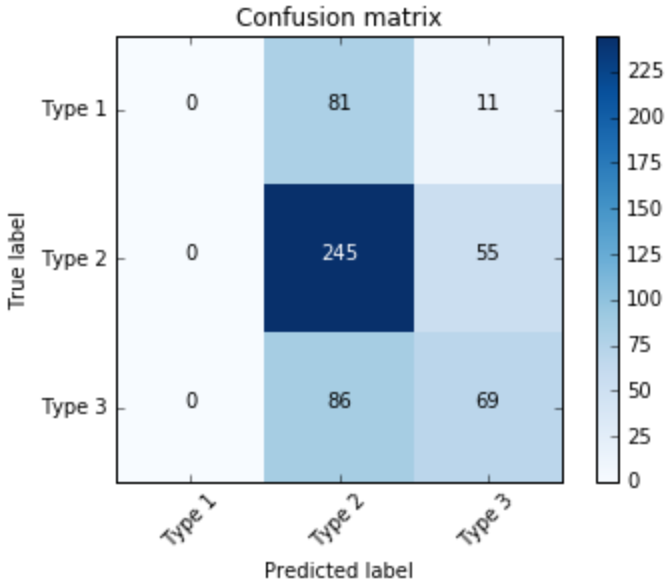

Figure 7. Confusion matrix for model predictions on validation set.

Upon prediction on the test set, the model achieved a test loss of 0.882. As of this report's submission, this score corresponded to a position of 282 / 802 on the public leaderboard, just outside the top third of contestants.

5. Discussion

Although our best model did not perform well enough for clinical applications, some important trends emerge from our analysis that will be useful in guiding future research. Our relatively high position on the leaderboard, in spite of our model's poor performance, suggests that this problem in particular poses a serious challenge for conventional deep learning approaches, as we found repeatedly in our experimentation and tuning.

In particular, our model's inability to identify type 1 images presents an extremely serious issue. However, in all of our experiments correcting for this phenomenon through class weighting, forcing our model to learn type 1 predictions reduced overall accuracy by as much as 10%. Given that this brute-force correction for class imbalance was not an effective solution, it is likely that the class imbalance alone is not responsible for this behavior. Future research in this direction should attempt to examine saliency maps for each of the three image types, in order to assess whether particular class-specific features could lead to this pathology. If so, using ImageNet-pretrained models for feature extraction may not be appropriate. Developing more

task-specific models, or perhaps more sophisticated ensembles to correct for multiple challenging sources of error, might be fruitful next steps. Dataset size was also a limiting factor, and likely the reason that a deep pretrained model like ResNet50 could not be adequately fine-tuned without severely overfitting.

Moreover, the scope of the Kaggle challenge format itself posed several limitations. The data cleaning challenges described, particularly the test set leakage, made it difficult to fully ensure that test set evaluation accurately assesses the classifier's generalization performance. Additionally, cross-entropy loss is a somewhat limited metric and (as we observed empirically) is not fully correlated with accuracy, but the blinding of the test set prevented a more comprehensive error analysis. A second set of test data will be released on June 15, along with the labels for the first set, and will provide the means for more reliable model evaluation. We are confident that further research in this area will produce an effective tool that will one day feature in clinician workflows and improve cervical cancer treatment worldwide.

Acknowledgements

We thank Russ Altman, Steven Bagley, and Hunter Boyce for guidance and advice. We thank Intel and MobileODT for providing the dataset. We thank Kaggle for hosting the Cervical Cancer Screening competition.

References

1. National Institutes of Health. *Cervical Cancer* [Fact sheet], 2017. Retrieved from https://report.nih.gov/nihfactsheets/viewfactsheet.aspx?csid=76.
2. Sankaranarayanan, R., et al. HPV screening for cervical cancer in rural India. In *N Engl J Med.* 360(14), pp. 1385-94, 2009.
3. Elson, D.A., et al. Sensitivity of the cervical transformation zone to estrogen-induced squamous carcinogenesis. In *Cancer Res.* 60(5), pp. 1267-75, 2000.
4. Jordan, J., et al. *The cervix*. John Wiley & Sons, 2009.
5. Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet classification with deep convolutional neural networks. In *NIPS*, pp. 1106–14, 2012.
6. Greenspan, H., van Ginneken, B., and Summers, R. M. Deep learning in medical imaging: overview and future promise of an exciting new technique. In *IEEE Transactions on Medical Imaging* 35(5)*,* pp. 1153-59, 2016.
7. Esteva, A., et al. Dermatologist-level classification of skin cancer with deep neural networks. In *Nature* 542, pp. 115-18, 2017.
8. Shin, H., et al. Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. In *IEEE Transactions on Medical Imaging* 35(5)*,* pp. 1285-98, 2016.

9. Tajbakhsh, N., et al. Convolutional neural networks for medical image analysis: full training or fine tuning? In *IEEE Transactions on Medical Imaging* 35(5)*,* pp. 1299-1312, 2016.
10. Song, Y., et al. A deep learning based framework for accurate segmentation of cervical cytoplasm and nuclei. In *EMBC*, 2014.
11. Gionis, A., Indyk, P., and Motwani, R. Similarity search in high dimensions via hashing. In *VLDB* 99(6), 1999.
12. Arefkhani, M., and Soryani, M. Malware clustering using image processing hashes. In *IEEE Machine Vision and Image Processing*, 2015.
13. Wong, S.C., et al. Understanding data augmentation for classification: when to warp? In *IEEE Digital Image Computing: Techniques and Applications*, 2016.
14. van der Walt, S., Colbert, S.C., and Varoquaux, G. The NumPy array: a structure for efficient numerical computation. In *Computing in Science & Engineering* 13(2), pp. 22-30, 2011.
15. Howse, J. OpenCV: Computer Vision with Python. Packt Publishing Ltd., 2013.
16. Pedregosa, F., et al. Scikit-learn: Machine learning in Python. In *J Mach Learn Res.* 12, pp. 2825-30, 2011.
17. van der Walt, S., et al. scikit-image: image processing in Python. In *PeerJ* 2, e453, 2014.
18. Cervix Segmentation (GMM). Intel & MobileODT Cervical Cancer Screening - Kaggle. May 6, 2017. Retrieved from https://www.kaggle.com/chattob/cervix-segmentation-gmm.
19. Greenspan, H., et al. Automatic detection of anatomical landmarks in uterine cervix images. In *IEEE Transactions on Medical Imaging* 28(3), pp. 454-68, 2009.
20. Chollet, F., et al. Keras (2015), GitHub: https://github.com/fchollet/keras.
21. Abadi, M., et al. TensorFlow: large-scale machine learning on heterogeneous systems (2015). Software available from tensorflow.org.
22. He, K., et al. Deep residual learning for image recognition. In *CVPR,* pp. 770-78, 2016.
23. Szegedy, C., et al. Rethinking the Inception architecture for computer vision. In *CVPR,* pp. 2818-26, 2016.
24. Szegedy, C., et al. Going deeper with convolutions. In *CVPR,* pp. 1-9, 2015.
25. Srivastava, N., et al. Dropout: a simple way to prevent neural networks from overfitting. In *J Mach Learn Res.* 15, pp. 1929-58, 2014.
26. Kingma, D.P., and Ba, J. Adam: a method for stochastic optimization. In *ICLR*, 2015.