CS 231N Final Project Report: Cervical Cancer Screening

Huyen Nguyen
Stanford University
huyenn@stanford.edu

Tucker Leavitt
Stanford University
tuckerl@stanford.edu

Yianni Laloudakis
Stanford University
jlalouda@stanford.edu

1. Abstract

The type of a patient's cervix determines the type of pre-cancer treatments the patient can undergo, and the medical community would benefit from an efficient method to classify a patient by their cervix type. Kaggle and Mobile ODT have published a collection of several thousand specular photographs of cervixes, each labeled as one of three types. We present our work in developing a convolutional neural network (CNN) to classify the cervix images in this dataset. We constructed and trained two models from scratch, CervixNet-1 and CervixNet-2. We also adapted two existing pretrained models to our dataset, ResNet v1 and Inception v2. We discuss the performance of all four of these models. Our most successful model, CervixNet-2, achieved a classification accuracy of 63%. We hope that this project inspires future work on the cervix classification problem; we suspect that better image segmentation could help improve model performance.

2. Introduction

The earlier the signs of cervical cancer are detected, the easier the treatment path will be for the patient. This treatment path varies for women based on the physiological differences in their cervix. Rural or understaffed clinics would benefit from a way of quickly and accurately classifying patients based on their cervix types. Cervical cancer tends to begin in cells within the transformation zone, which could be completely ectocervical and visible (Type 1), partially endocervical but visible (Type 2), or partially endocervical and not fully visible (Type 3) (see Figure 1). Cervix types 2 and 3 may require different screening or treatment due to the placement and hidden view of precancerous lesions.

The input to our classifiers is a photograph of the cervix taken through a vaginal speculum. The output is the probability distribution over the three classes, from which we extract the most likely class. Quantitatively, the goal is to minimize the cross-entropy loss J for the classification:

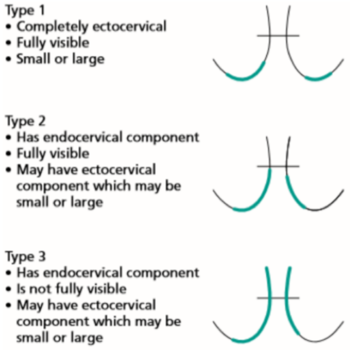
Type 1
• Completely ectocervical
• Fully visible
• Small or large
Type 2
• Has endocervical component
• Fully visible
• May have ectocervical component which may be small or large
Type 3
• Has endocervical component
• Is not fully visible
• May have ectocervical component which may be small or large

Figure 1. Characteristics of the three cervix types (taken from [3])

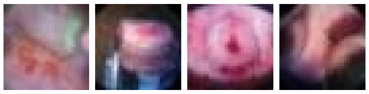

Figure 2. sample data from the Kaggle Dataset

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

where N is the number of testing data points, C is the number of classes (3, in this case), y is the one-hot vector for the correct class, and p_{ij} is the predicted probability that data point i has class j. Our project is to use deep learning and computer vision to automate and improve this important classification process.

This project was inspired by a public Kaggle competition, and the dataset is provided on Kaggle's website.

3. Related Works

Deep learning and computer vision have proven effective in the healthcare domain for classification or segmentation of medical images. Recent efforts using deep learning generally either use transfer learning with models pre-trained on ImageNet or copy the architecture of these models and train them from scratch. One recent attempt at cervical cancer classification combined image features from the last fully connected layer of pre-trained AlexNet with biological features extracted from a Pap smear to make the prediction [4]. Another group used features computed from images of cells from a cervix biopsy as input into a feed-forward neural network to predict the presence of cancer [5]. Other manual features from a Pap smear such as Grey level, wavelet, and Grey level co-occurrence matrix have been used for cancer detection [6]. Deep learning has also been used for other types of cancer detection. A convolutional neural network (CNN) following OxfordNet's structure was used to detect mammographic lesions [7]. A CNN with parameters pre-trained on a similar dataset was also used to differentiate between mammographic cysts and lesions [8]. A recent paper from a group of Stanford researchers has excited the medical community and uses a pre-trained Inception-v3 model and hierarchical algorithm to classify different skin malignancies with results comparable to expert dermatologists [9]. Another study analyzed colonoscopy video footage and used a CNN to compute image features which were then later used to predict the bounding boxes for different polyps [10]. No pre-segmentation was used in one study of lung nodule classification, which used a CNN feature extractor [11].

Automated cervix and cervical cell segmentation is another important area of study. One method takes care to remove glare from the photo and uses K-nearest neighbors (KNN) with images pre-segmented by a distance metric based off of the histogram of oriented gradients to locate the most similar bounding boxes and averages them [12]. A model by researchers at Medical College of Georgia also used glare removal, K-means clustering, and texture features to segment the different cell types around the cervix [13]. A similar method fed color and cell area features into K-means to segment the cervix [14]. Another group performed cervix segmentation by first transforming the image from RGB to luminosity, red-green chromaticism, and blue-yellow chromaticism, and then ran K-means and selected the largest region [15]. One group found that using a CNN to segment cervical cell cytoplasm and nuclei outperformed traditional filters and classification methods, especially when multiple cells were in the picture [16]. LeNet5 was used as inspiration for another group's epithelial cell segmentation task [17]. They coped with dataset scarcity by extensively augmenting the dataset with flips and rotations. Similarly, a LeNet-like architecture was also used for segmentation of bones in x-rays using pixel-wise classification [18].

4. Dataset

Kaggle provides a dataset of approximately 1500 labeled cervix images. The images are graphic and may offend some viewers. Included in Figure 2 are several thumbnail sized versions of the training data. Our training set contained a total of 1481 images (see Table 1 for a breakdown by type) while the test set contains 512 images with the labels not publicly available. Notice that Type 2 makes up over half of the available training data, while Type 1 only makes up 17%. Each image has a variable number of pixels but all are colored images.

Table 1. Class distribution of the dataset

Type 1	Type 2	Type 3
251	782	451
16.9%	52.7%	30.4%

Kaggle provides additional data for training, but the additional data is of low quality. Manual inspection of the data reveals that many images are duplicated, and some images are not even of cervixes (e.g. we found a picture of a woman's face, a picture of a finger, and a picture of some newsprint). We found that training on the additional data did not improve model performance; this is likely because the additional dataset is not drawn from the same distribution as the training dataset. We excluded the additional data from our analysis for this reason.

In an attempt to visualize out data set, we performed PCA on the raw images values to look for clustering and grouping by type and also performed t-SNE analysis on the first 3 principal components us sci-kit learn [23]. Unsurprisingly, as can be seen in Figures 3 and 4, the cervix types do not fall into clusters based on this analysis, indicating that our input data points resemble each other.

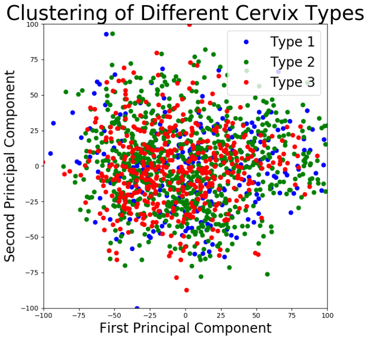

Figure 3. Training data distribution over the first two principle components. No obvious clusters have emerged.

We used either only the original dataset or with the additional dataset, combined with different data augmentation methods (see Preprocessing section). We then randomly chose 10% of the labeled data for validation, and the rest for training.

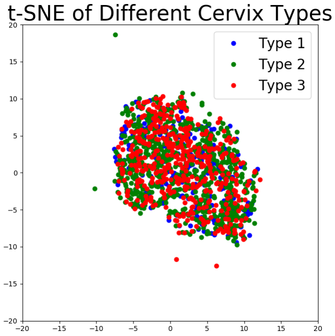

Figure 4. t-SNE clustering of first four principal components

5. Methods

5.1. Preprocessing

Since the initial images provided were much to large (more than 2000 pixels a side) as well as irregularly shaped, the first step was to crop the initial images into a square with the length of the shortest initial side. Then, a 160x160 or 224x224 segment of the image was cut from the center of the larger image. The assumption, which turns out to be true most of the time, is that the cervix will be in the center of the image since it is the most important.

We attempted a variety of data set augmentation methods to cope with the small dataset. We performed random horizontal and vertical flipping, 90°and 270°rotations, as well as random rotation, random cropping, and random scaling of the inputs.

5.2. Segmentation

Although not the main focus of this project, given the attention paid to segmentation in the literature, we thought it best to make an effort to segment the cervix, which would help with removing extraneous objects and tissues from the input. We took inspiration from [12, 13, 14, 15], who used K-means and KNN to aid in their segmentation process. Our segmentation pipeline is as follows: first, the image is run through scikit-learn's image segmentation algorithm, which uses K-means to create roughly K image patches based on proximity and color similarity [21, 24]. Then KNN is used to determine which of these patches is cervical tissue. While [12] used the relatively sophisticated histogram of oriented gradients approach to find the patches closest to pre-segmented cervices, we did not have the luxury of many pre-segmented cervices. Instead, we manually segmented 10 random cervices and computed the average red, green, and blue values, giving us a 3 element feature vector. Then, to decide which of the K patches contained cervical tissue, we performed KNN using the average color vector for the patch as the feature. We took the M patches with the lowest distances as well as any patches that were contained within these patches and used them to create a binary mask. This final modification was necessary because the center of the cervix frequently had a redder color than could be represented by the average color vector, causing it to be mistakenly excluded. In practice, we used 10 for K and 5 for M. The higher K and M are, the higher the chance of including cervical tissue but also extraneous objects. Some successful and unsuccessful segmentations are shown in the following figures.

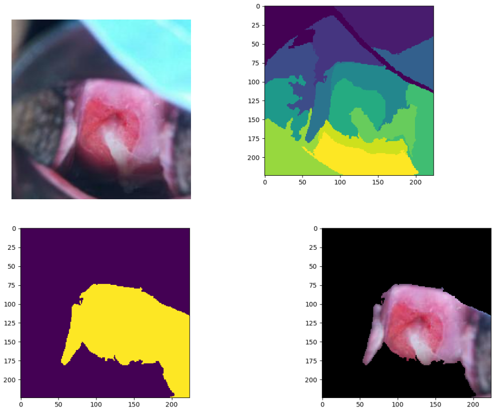

Figure 5. A successful segmentation. The initial image, the K-means patches, the KNN binary mask, and the final image. Note the glove and speculum are removed but the cervix remains.

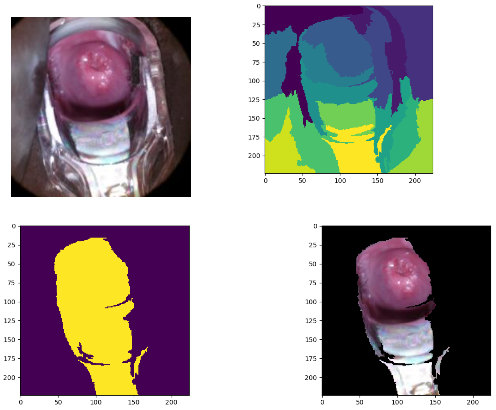

Figure 6. A semi-successful segmentation. The initial image, the K-means patches, the KNN binary mask, and the final image. Perhaps the average color of the plastic was close enough to cervical tissue to be included.

5.3. Model Architectures

We built two models from scratch for this project: CervixNet-1, a shallow net with two convolutional layers, and CervixNet-2, a deeper net with five convolutional layers.

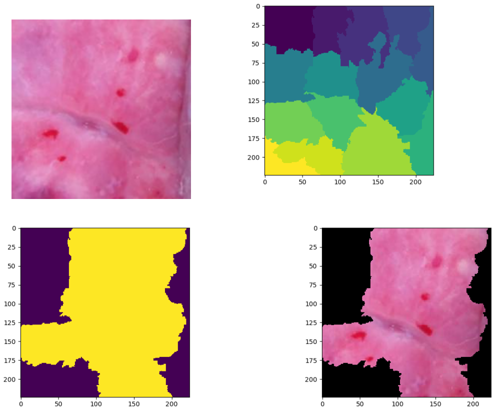

Figure 7. A failed segmentation. The initial image, the K-means patches, the KNN binary mask, and the final image. Because the initial image was so zoomed in, the final segmentation actually lost tissue.

5.3.1 CervixNet-1

Our first attempt was a relatively shallow convolutional network that used a batchnorm layer after every convolutional layer. The network's architecture is described in detail in Figure 8.

This model was inspired by Question 5 in Assignment 2, in which we built a multilayer convolutional net to classify images in the CIFAR-10 dataset. The assignment description recommended a network architecture with a batch-normalization layer after every convolutional layer. This confers a number of advantages:

- it reduces the dependence of the model on weight initializations
- it improves gradient flow through the network, increasing training speed
- it acts as a form of regularization

CervixNet-1 trained noticeably faster than the model we submitted for our project milestone, which used fewer batch normalization layers.

CervixNet-1 contains 12,684,876 parameters, and 33.1% of the total parameters are in the two fully connected-layers at the output. Batch normalization significantly improved model performance. We did not use pooling in this model, preferring instead to use strided convolutions to decrease the output size.

Because of the gap between our train and validation losses, we incorporated dropout in every convolutional layer to increase regularization. Additionally, we applied L2 regularization to the last two fully-connected layers to discourage overfitting to the features learned by the previous convolutional layers.

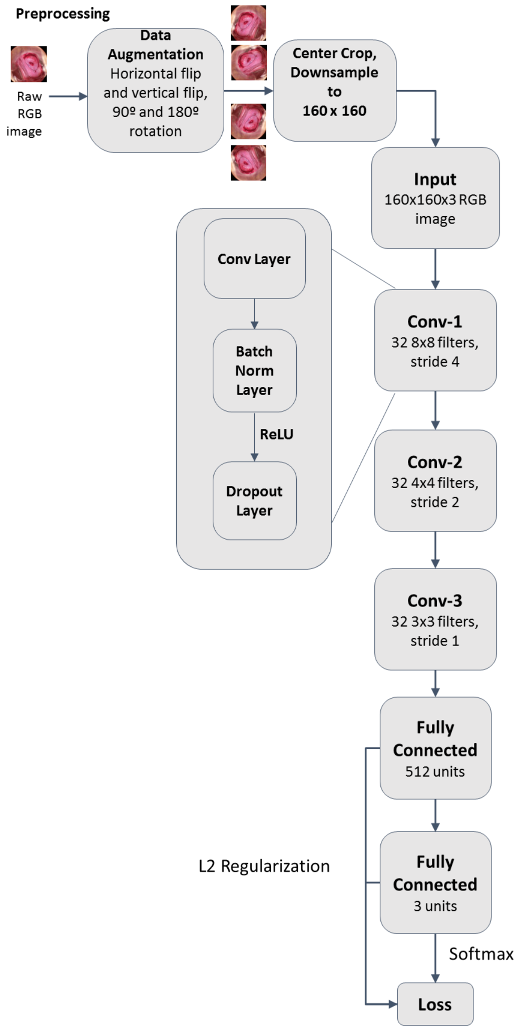

Figure 8. Model architecture for CervixNet-1

5.3.2 CervixNet-2

CervixNet-2 was intended to be a compromise between CervixNet-1 and larger pretrained models and is described in Figure 9. It features more convolutional layers and uses max pooling instead of strided convolutions to reduce the dimensions. It followed the general design principle of first building up information with convolutional layers before losing information with pooling to avoid representational

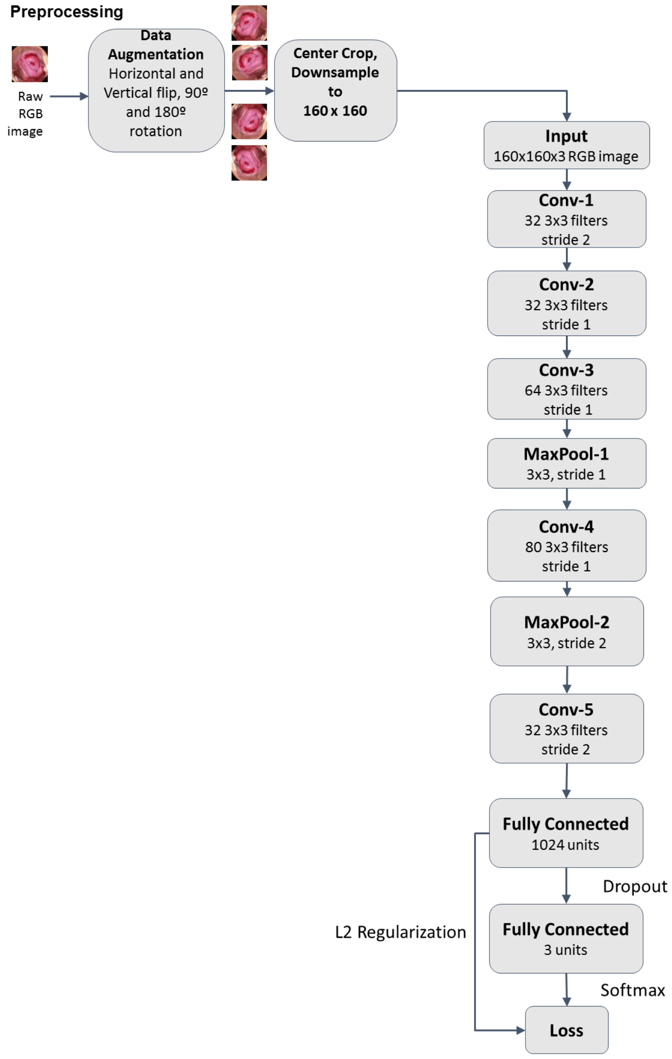

Figure 9. Model architecture for CervixNet-2

bottlenecks. Also, it made use of a 3x3 filter with a stride of two followed by another 3x3 filter with a stride of one, which is a computationally cheaper way of boosting receptive field size instead of using a larger filter size. Both of these design principles were recommended in [20], a paper which analyzed the inception architecture.

5.3.3 Pretrained Models

Using pretrained models as a basis for training can help jump-start the training process and take advantage of known successful model architectures. We applied the following pretrained models to our problem, both of which have been highly successful on the ImageNet dataset:

- ResNet v1 [19]
- Inception v2 [20]

In many problems, retraining only the final fully-connected layers of the model is necessary. This is because the pretrained models are already tuned to extract meaningful features from their inputs, and the job of the final fully-connected layers is to decide which of these meaningful features is relevant for the current classification problem.

However, in our case, we found that the pretrained models performed poorly unless the entire network was retrained. This may be because our dataset has a much lower dimensionality than ImageNet, on which the pretrained models had been trained. This means the models will produce similar features when run on our images, and so all our input images look “the same” in feature space and classification accuracy is poor. We must retrain the entire model to learn a new set of features that better represents the differences between images in our dataset.

Over-fitting becomes a significant risk when retraining the entire pretrained models since our dataset is several orders of magnitude smaller than the dataset used to train the pretrained models (our dataset contains $\sim 10^3$ images, whereas ImageNet contains $\sim 10^7$).

ResNet v1 ResNet is an unusually deep neural network (containing hundreds of layers) that aims to learn ”residual functions” with respect to the layer inputs, instead of learning unreferenced functions like most other models. It does this by ”shortcutting” the layer inputs to the layer outputs, so that the output of the layer is the input plus some ”residual function” learned by the model. The ”shortcuts” provide an avenue for uninterrupted gradient flow and allow for the training of much deeper models than with conventional architectures.

Inception-v2 We choose to use the inception architecture [20] because of its success in [9] of transfer learning with skin cancer. The main advantage of the inception architecture is that it examines the input at multiple granularities by using different filters and spans and concatenates these results together as show in the figure below. By concatenating these different granularities, less information is lost as the depth increases. As part of our experimentation with this model, we trained many Inception-v2 nets with different dropout and L2 regularizations, which is described in the experiments section. Additionally, we varied the number of iterations of training the full net versus training the last fully connected layer. We used the weights provided in TensorFlow-Slim [25].

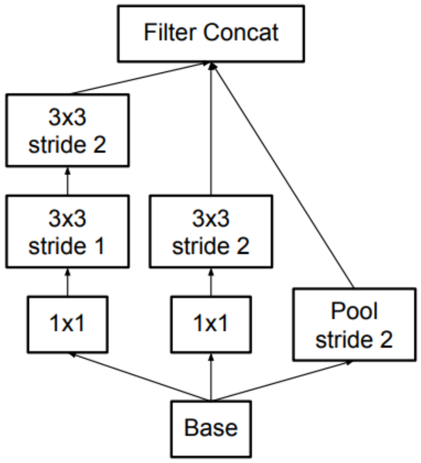

Figure 10. One kind of inception module. [20]

6. Experiments

6.1. CervixNet-1

With CervixNet-1, we experimented with tuning the:

- learning rate,
- L2 regularization strength,
- dropout probability, and
- number of filters in each layer

We also experimented with training the net on the entire provided dataset (including the duplicate and incorrect images) instead of only the original, higher-quality dataset. Below is the learning curve for some experiments with different datasets.

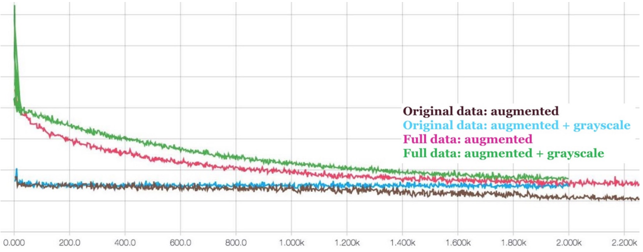

Figure 11. Training oss curves during for CervixNet-1. Original means original data. Full means original + additional data

We can see that the val loss when trained on original data goes down much faster than when trained on full dataset. This makes sense since the original dataset is much cleaner, and therefore it's easier to interfere the validation data from the train data.

Table 2. Results of some best experiments on CervixNet-1

Dataset	Hyperparameters	Val loss	Test loss
orig augmented, 160 x 160	keep=0.8, l2=0.0, lr=0.0001	0.8242	0.86433
full augmented, 160 x 160	keep=0.85, l2=0.01, lr=0.0001	1.0821	0.87287
full augmented, 160 x 160	keep=1.0, l2=0.0, lr=0.0001	0.8412	0.89765
full augmented, 160 x 160	keep=0.8, l2=0.0, lr=0.0001	0.8926	0.898

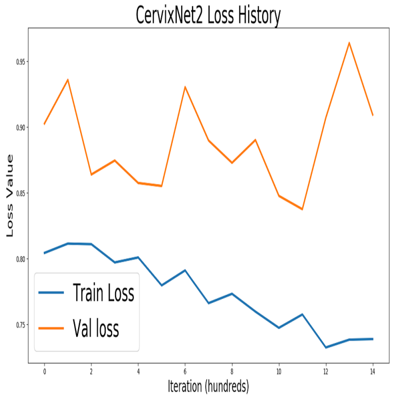

Figure 12. Loss curves for the best performing model.

6.2. CervixNet-2

We pursued this model after seeing the results from CervixNet-1 and Inception v2. The best model benefited from a heavy dropout of .5 and a modest L2 regularization of 0.1. We found that batch-norm actually harmed this model and that it attained its lowest loss without it. We used a learning rate of .0012, which we found via grid search. We used the RMSProp optimizer and annealed the learning rate every 400 iterations by .95 [22].

This model turned out to be our best performing model, giving us a test loss of 0.81768.

6.3. ResNet v1

We used the 101-layer architecture presented in [19] and used the pretrained model weights presented in the enclosed github repo. We retrained the entire network on the cervix dataset for:

- 500 gradient update steps,
- an initial learning rate of 0.003,
- an annealing factor of .4 every 100 steps, and
- an l2 regularization strength of 0.0001

In the original paper, they use a higher initial learning rate, a stricter annealing schedule, and the same l2 regularization penalty, but they train the model for $\sim 10^4$ iterations instead of $\sim 10^3$. To run the optimization step, we followed the paper and used SGD with Momentum, with a momentum weight of 0.9.

To choose the learning rate and regularization strength, we trained the model for 50 iterations at 5 different learning rates distributed logarithmically from 10^{-4} to 10^{-2} and regularizations strengths of $10^{-3}, 10^{-4}$, and 10^{-6}. These cross-validation values were chosen based on the reported hyperparameter values in the paper. To preprocess the data, images are randomly dilated (i.e. resized), randomly cropped, and randomly flipped [19]. This is both to augment the dataset and prevent over-fitting to irrelevant image features related to spatial location or image size.

Figure 13 shows a representative training loss curve for the ResNet training. The learning rate was annealed by a factor of 0.4 at iteration 100 and 200, and the logging frequency was reduced by a factor of 5 at iteration 230.

Though the training loss decreased appreciably over the first hundred iterations, the loss begins to plateau after iteration 100. Due to time and resource constraints, the ResNet model could only be trained for a limited number of gradient steps. It is likely that model performance could have been improved by training for more iterations. The decrease in loss over the first 100 iterations is likely due to the last fully connected layer training to fit the data; it yields significant progress relatively quickly. For the rest of the time, the entire model is training, and will likely take on the order of 10^4 iterations to fully converge.

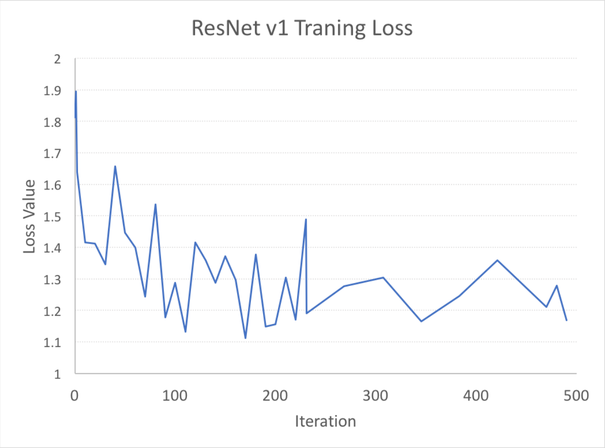

Figure 13. Training loss curve for the ResNet v1 model.

By examining the predictions ResNet makes, its clear that the model has not yet converged. Table 3 shows the average softmax probability assigned to images in the validation data set, and compares it with the actual prevalance of each class in the entire data set. For an accurate model, these two statistics should be the same in expectation. Clearly, for this model, they are not. The Resnet model predicts class 1 much more often than it should, and hardly ever predicts class 3, which represents almost a third of the dataset.

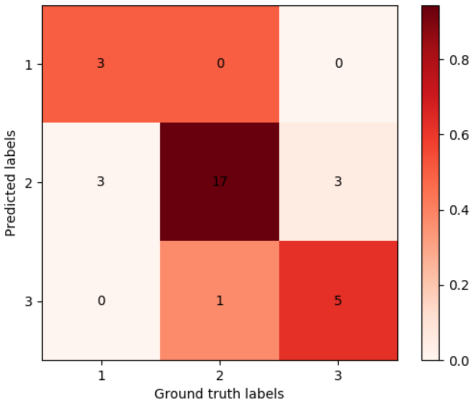

Figure 14. Confusion matrix for 32 validation data points with the ResNet v1 model

Table 3. ResNet predicted class distributions vs. actual Class distributions

	Type 1	Type 2	Type 3
Average predicted Class Prevalence	0.499	0.454	0.046
Actual Class Prevalance	0.169	0.527	0.304

6.4. Inception

We had great difficulty attaining good results using the pre-trained Inception v2 network. To try and get different results, we changed the dropout keep probability, the L2 regularization, the iteration where we switched from training the full net to training the last layer, and which dataset we used (segmented or not segmented cervices). The results are shown in Table 2. To address the large discrepancy between train and validation losses, which can be seen in Figure 11, we tested heavy dropout and L2 regularization. The same learning rate of .001 was used for all experiments because the train loss dropped quickly enough. TensorFlow's Momentum Optimizer with a learning rate of .001 and a momentum of .9 was used. The learning rate was annealed by a factor of .95 every 400 steps and gradients were clipped so that the maximum global norm is 2. The poor performance of transfer learning with Inception v2 can be ascribed to the ImageNet pictures being too different from cervices and the fact that the net is too powerful for this dataset. Despite heavy regularization there was still a massive gap between the training and validation sets, indicating that the net was learning noise.

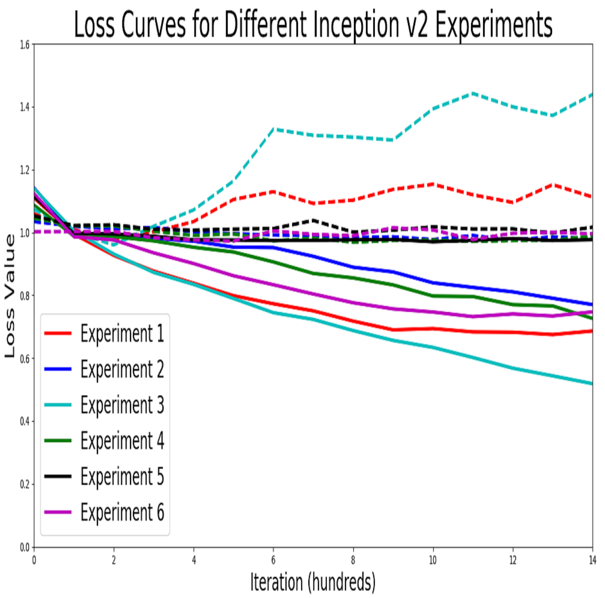

Figure 15. Train (solid) and Validation (dashed) plots for the different experiments listed in Table 2.

Table 4. Performance Statistics for Inception v2 Experiments

Exp. Num	Dropout	L2 Reg.	Switch	Dataset	Test Loss
1	0.5	1.0	1000	Seg.	.95655
2	0.4	10.0	Never	Unseg.	.97413
3	0.5	1	Never	Unseg.	.96866
4	0.5	10.0	Never	Unseg.	.95646
5	0.5	5	500	Unseg.	.99432
6	0.5	5	1000	Unseg.	.97094

7. Results

Table 5 summarizes the best results achieved by each of our four model architectures. Figure 16 shows a confusion matrix produced by our best model on 100 validation data points. Figure 17 shows a saliency map on several validation images. Our model assigns a high probability to the correct class for the left three images and a low probability to the correct class for the right three images. The maps indicate that the model is not successfully identifying the important features of the cervix, since nearly all of pixels in each image impact the gradient and the saliency maps for the correctly classified and incorrectly classified images look nearly indistinguishable.

Table 5. Performance Statistics for best performing models of each Architecture

Model	Validation Loss	Validation Accuracy	Test Loss
CervixNet-1	0.8242	61.7%	0.86433
CervixNet-2	0.83758835	65.1%	0.81768
Resnet v1	0.8971841	62.3%	1.08586
Inception v2	0.96845	52.6%	0.95646

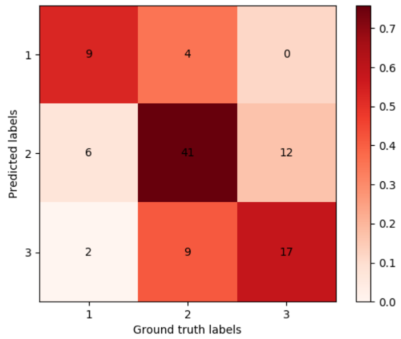

Figure 16. Confusion matrix for 100 validation data points on CervixNet-2 [5.4].

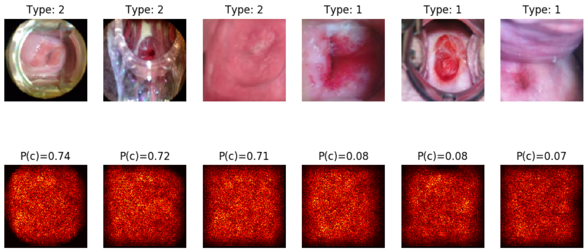

Figure 17. Saliency maps for CervixNet-1 on several validation images.

8. Conclusion and Future Work

The cervix classification problem is a challenging one. Our data is limited and of low quality. There are leaks (images on the test set that is also on the given train set with labels), inconsistency (duplicated images with different labels). Additionally, we didn't have enough background in image processing to really take advantage of the data we had. The results we got after cleansing and augmenting the data are only small improvements from results obtained with the original, un-augmented data. Furthermore, We struggled with understanding the correlation between the loss on our validation set and the loss on the Kaggle's test set. For some models, lower val loss does lead to lower test loss, but for some models, lower val loss leads to higher test loss.

If we had more time to work on this project, we would try more and different data preprocessing techniques, such as training a dilated convolutional network to localize the cervix: with each pixel, the network determines whether that pixel belongs to the cervix or not. We spent some time to get rid of low quality images, but maybe we should have spent more time to rigorously cleanse the additional data. Batch normalization proved to increase performance on some models at the cost of lower speed. We could have tried weight normalization instead.

Our best training cross-enntropy loss score of 0.817 puts us within the top 200 submissions on Kaggle. Given more time to experiment and refine, we expect this score can be improved.

We learned a lot from the project, both about image processing and deep neural networks. This is also the first Kaggle competition for all our team members, and we all thought that it was a fun experience. This motivates us to do not only more Kaggle competitions in the future, but to apply what we've learned in class to real world problems.

References

[1] Ioffe, S., & Szegedy, C. "Batch Normalization : Accelerating Deep Network Training by Reducing Internal Covariate Shift." arXiv Preprint arXiv:1502.03167v3. (2015).

[2] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. J. Machine Learning Res. 15, 19291958 (2014).

[3] MobileODT, Intel, & Kaggle Inc. "Intel & MobileODT Cervical Cancer Screening." www.kaggle.com/c/intel-mobileodt-cervical-cancer-screening (2017).

[4] Xu, Tao, et al. "Multimodal Deep Learning for Cervical Dysplasia Diagnosis." International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer International Publishing, 2016.

[5] Sokouti, Babak, Siamak Haghipour, and Ali Dastranj Tabrizi. "A framework for diagnosing cervical cancer disease based on feedforward MLP neural network and ThinPrep histopathological cell image features." Neural Computing and Applications 24.1 (2014): 221-232.

[6] Sukumar, P., and R. K. Gnanamurthy. "Computer aided detection of cervical cancer using PAP smear images based on hybrid classifier." International Journal of Applied Engineering Research 10.8 (2015): 21021-32.

[7] Kooi, Thijs, et al. "Large scale deep learning for computer aided detection of mammographic lesions." Medical image analysis 35 (2017): 303-312.

[8] Kooi, Thijs, et al. "Discriminating solitary cysts from soft tissue lesions in mammography using a pretrained deep convolutional neural network." Medical physics 44.3 (2017): 1017-1027.

[9] Esteva, Andre, et al. "Dermatologist-level classification of skin cancer with deep neural networks." Nature 542.7639 (2017): 115-118.

[10] Park, Sun Young, and Dusty Sargent. "Colonoscopic polyp detection using convolutional neural networks." SPIE Medical Imaging. International Society for Optics and Photonics, 2016.

[11] Shen, Wei, et al. "Multi-scale convolutional neural networks for lung nodule classification." International Conference on Information Processing in Medical Imaging. Springer International Publishing, 2015.

[12] Song, Dezhao, et al. "Multimodal Entity Coreference for Cervical Dysplasia Diagnosis." IEEE transactions on medical imaging 34.1 (2015): 229-245.

[13] Li, Wenjing, et al. "Automated image analysis of uterine cervical images." Medical Imaging. International Society for Optics and Photonics, 2007.

[14] Srinivasan, Yeshwanth, et al. "A probabilistic approach to segmentation and classification of neoplasia in uterine cervix images using color and geometric features." Medical Imaging. International Society for Optics and Photonics, 2005.

[15] Das, Abhishek, Avijit Kar, and Debasis Bhattacharyya. "Elimination of specular reflection and identification of ROI: The first step in automated detection of Cervical Cancer using Digital Colposcopy." Imaging Systems and Techniques (IST), 2011 IEEE International Conference on. IEEE, 2011.

[16] Song, Youyi, et al. "A deep learning based framework for accurate segmentation of cervical cytoplasm and nuclei." Engineering in Medicine and Biology Society (EMBC), 2014 36th annual international conference of the IEEE. IEEE, 2014.

[17] Malon, Christopher, et al. "Identifying histological elements with convolutional neural networks." Proceedings of the 5th international conference on Soft computing as transdisciplinary science and technology. ACM, 2008.

[18] Cernazanu-Glavan, Cosmin, and Stefan Holban. "Segmentation of bone structure in X-ray images using convolutional neural network." Adv. Electr. Comput. Eng 13.1 (2013): 87-94.

[19] He, K. et al. "Deep Residual Learning for Image Recognition" arXiv Prepring arXiv:1512.03385. (2015)

[20] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. "Rethinking the inception architecture for computer vision." arXiv preprint arXiv:1512.00567. (2015)

[21] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Suesstrunk, SLIC Superpixels Compared to State-of-the-art Superpixel Methods, TPAMI, May 2012.

[22] Hinton, Geoffrey, NiRsh Srivastava, and Kevin Swersky. "Neural Networks for Machine Learning Lecture 6a Overview of mini-batch gradient descent." (2012).

[23] Pedregosa, Fabian, et al. "Scikit-learn: Machine learning in Python." Journal of Machine Learning Research 12.Oct (2011): 2825-2830.

[24] Van der Walt, Stefan, et al. "scikit-image: image processing in Python." PeerJ 2 (2014): e453.

[25] Martn Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Rafal Jozefowicz, Yangqing Jia, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Man, Mike Schuster, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Vigas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.