The Effect of Training Published Computer Vision Models on Unreal Engine Synthetic Images

An Introduction to the Synergy between Graphics Rendering Software and Machine Learning

John W. Belanger Smutny, Software Engineer, Northrop Grumman Corporation
Bradley Department of Electrical and Computer Engineering, Virginia Tech

***Abstract*— Computer vision (CV) technology can be used in an infinite number of applications. However, publicly available images to train these types of models are often not. For niche applications of CV technology, public datasets of real-world images can be difficult to find. This project explores how developers can use synthetic images generated in graphics rendering software (like Unreal Engine) to supplement their limited amounts of real images. Thus, allowing the user to use those real images for validation and testing sets.**

In this report, published CV models are trained on different ratios of real & synthetic images for two binary classification tasks: classifying an image as 1) a cat or dog, and 2) for weld defect detection. The traditional convolutional neural network (CNN) model VGG16 can maintain test accuracy with nearly all training samples being synthetic, even improving when small concentrations of synthetic samples supplemented the real samples. The squeeze-excite smaller CNN MobileNetV3-small showed a steady decline in accuracy as the concentration of real samples in the training set decreased (possibly due to not having enough trainable parameters). For both models, padding synthetic images to the minority class in an imbalanced dataset showed slight improvements in classification performance over other rebalancing methods.

***Index Terms*— Computer Vision, Unreal Engine, Real Synthetic Ratio, Synthetic Utilization, Welding.**

1. Introduction & Problem Statement

All machine learning (ML) models require large amounts of training data to learn. Gathering a dataset large enough and with enough variation of real images can be difficult depending on the application. If a dataset is not publicly available, gathering a dataset can require significant time (manually collecting each image), financial expenditure (licensing), and/or legal constraints (copyright of images from the internet). One alternative is to generate photo-realistic images using free graphics rendering software already used in TV, film video games, and various business applications.

This paper explores how generated images from Unreal Engine (UE) affect binary classification performance of two published CV models: VGG16 and MobileNetV3-small. First, how training on different amounts of real and synthetic images affect key model performance metrics (Accuracy, Loss, and AUCROC). Second, how does the previously described effect change depending on the amount of synthetic data available to be sampled. Lastly, how adding synthetic images to a minority class in an imbalanced dataset compares with other rebalancing techniques. Additionally, this project provides context on how to use the using Unreal Engine (UE) to develop one's own synthetic images, the potential business cases for it, and possible restrictions to consider.

2. Related Research

Between 2018-2022, a technical survey by Dr Keith Man and several papers explored the use of synthetic images on CV models [Man22]. In aggregate, they described that mixing real and synthetic images can improve classifier performance if the synthetic images had enough latent variation to expand the mimicked class's feature space. In practice, the synthetic images allowed developers to expose classification models to objects in new contexts, thus improving the model's understanding of the object's key features [ECCV]. However, if there are not enough real images or not enough differences between synthetic images, using synthetic images can have a negligible or detrimental effect on the CV model [Thornström19][Gastelum20].

3. Methodology

Having a variety of images is the key to introducing synthetic images to a CV model. This report uses datasets and classical image augmentations to introduce more image variation. Various model training techniques and model architecture changes are applied to better adapt the published CV models to these binary classification tasks. As a note, during all experiments Synthetic images are only used during model training, model validation and testing sets only use real images.

3.I. Datasets

Each classification task uses unique sets of real and synthetic images. Real images are gathered from publicly available sources, while synthetic data is created (covered in the next section). The Cat_Dog real dataset comes from the ImageNet classification standard set (a balanced set of 25,000 images) [Elson]. Two synthetic datasets are used for the Cat_Dog task: each with 9425 and 47864 rendered images respectively.

The Welding real image dataset contains 2221 images from several public repositories. Only 38.2% of the images from these repositories are used due to image duplication, improper framing, and lack of relevance to metal welding. Five synthetic image variation groups make up the generated 8160 images: *No change*, *minor imperfections*, *porosity defect*, *crack defect*, and *incomplete weld defect*. Both real and synthetic welding datasets are imbalanced (there are more images of the *Defect* class than *Good* weld class).

3.II. Creating Synthetic Images

All synthetic images are collected from executing a python script in a UE Editor instance. During execution, the python script sends various commands to the UE Editor. Any python script executed from the UE Editor runs on the same thread as the Editor through a software callback. This project's script sends commands to collect screenshots of the target of interest in various lightings, backgrounds, and camera angles at a rate of one command every 30 frames (resulting in 80 unique ~1980x1240 screenshots a minute). This rate is chosen to give each callback enough time to execute all necessary UE commands prior to the prompting the next callback.

The 47864 image Cat_Dog synthetic dataset is made up of 13 Cat and 10 Dog publicly available low-polygon (~10,000) .fbx model files [3D Ace]. The screenshots capture 100 different camera angles, 10 backgrounds, and 8 lighting settings. For welding, the 8160 synthetic weld images are collected from 6 color variations (Aluminum, Steel, copper, gold, etc), 68 unique locations along a random weld seam (provided by a purchased UE material. At each point, 4 screenshots are taken at random camera angles and point light offsets of each of the 5 seam modifications described in the previous section.

Figure 3.1: Example screenshots taken at different angles, models, and backdrops prior to 2D image augmentation.

Figure 3.2: (left) Image of the plate in which screenshots are collected (specifically shown is the purchased 'Metal_Weld' material with the UE starter content 'steel' basecolor). (right) Side view of the metal plate used to collect images. Note: all texture map edits are to a 2D plane. Any perceived height is an optical illusion from the Normal and Ambient Occlusion maps.

The three weld defects (*porosity*, *crack*, & *incomplete weld*) are simulated by modifying UE Texture Maps that combine to form the defect captured in each screenshot. These defects are applied to a ~34 pixels wide weld seam with different dimensions at each of the 68 locations. The defect dimensions are randomly chosen from that dimension's set range of values. Porosity defects are formed by editing the Normal and Ambient Occlusion (AO) maps in an ellipsis shape. Cracks are formed from offsetting a line segment. Incomplete welds remove the Normal & AO map from parts of the weld seam and inserts a stand in 'seam' representing the space in between the desired metal plates. Smaller scaled edits of these defects are a separate variation referred to as *minor imperfections*. These *minor* imperfections represent small changes to the weld seam but not enough to be a *Defect*.

For this project, {*no change*, *minor imperfections*} are classified as *Good* welds (positives - 1), while {*porosity*, *crack*, and *incomplete welds*} are classified as *Defects* (negatives – 0).

Figure 3.3 : Examples of (left) porosity, (middle) crack, and (right) incomple weld defects created from Normal, Metallic, Roughness, and Ambient Occlusion texture maps.

3.III. Used Models & Architecture Changes

VGG16 and MobileNetV3-small are the two models examined in this research. Both represent different ways CV models can be applied. The VGG16 model is a model with many parameters that is best ran where there are minor limitations on power consumption and computer memory, such as hosted on an on-site or cloud server. The model has a common convolutional neural network (CNN) architecture involving 16 sets of *Convolution-Pooling-BatchNorm* layers [Simonyan14].

MobileNetV3-small is a variation of the MobileNetV3 model that has the fewest number of trainable parameters. The model is specifically designed for running on mobile devices & on-chip hardware, prioritizing lower power usage, a low memory footprint, and reduced latency from model input to output. MobileNetV3's architecture utilizes Squeeze-Excite layers that advance upon residual neural networks introduced by Dr Kaiming He's ResNet model [He]. After each convolutional layer, MobileNetV3 uses Squeeze-Excite residual network blocks to pool (squeeze the output) the Conv2D layer's output, passes them through fully connected (FC) layers (excite), before scaling the output for the next Conv2D layer. These FC layers apply weights to each of the previous Conv2D layer's filters, thus prioritizing which filters contain the most important features [Howard19].

	VGG16	**MobileNetV3**
General Use Case	Minor resource constraints	Constraints on Memory, Power & Latency
General Architecture	Conv-AvgPool-BatchNorm	Squeeze & Excite
Total Parameters in the published model for 1000 class classification	138,357,544	2,554,968
Total Parameters in the transfer learning models used in this research for binary classification	21,203,521 (15.3%)	1,365,617 (53.4%)
File size of weights (.h5)	80.9 MB	5.4 MB

Figure 3.4: Summary of the VGG16 and MobileNetV3-small models and magnitude of the size reduction in trainable parameters from the original paper to the models used in this report [Simonyan14][Howard19]

3.IV. Model Training

Various techniques are used during model training to limit model overfitting. These techniques include initial learning (five epochs), learning rate decay (0.95 decay per epoch), early stopping (0.001 tolerance to Val_Acc or Val_ROC), and 2D image augmentations such as flips, shears, zooms, rotation, color contrast, and width+height shifts. Because of these augmentations, every epoch every training image has a uniformly random chance for none, one, or many of the augmentations to be applied. Resulting in the model calculating error gradients from a slightly different training set every epoch. Unlike the training set, the validation and testing sets for all models and all tests are only comprised of real images, no augmentations are applied, and the sets are always balanced for each binary class.

4. Experimental Results

In total, five experiments are performed in this research. Four (3 Cat_Dog & 1 Welding) are Real:Synthetic Ratio (RSR) experiments where testing metrics (Accuracy & AUCROC) are plotted as real data replaces synthetic data in the training set. The last experiment (1 Welding) compares the performance of various classification dataset rebalancing methods. Results are expressed in terms of confusion matrix performance. Specifically focusing on True-Positives, True-Negatives, and False-Positives. For most experiments, the shown plots will be for the VGG16 model (MobileNet plots not shown are in the Appendix). Results for both models will be commented on.

Additional Notes:

- For Weld RSR tests, the training set is imbalanced with a skew of 7:3 (7 *Defect* images for every 3 *Good* images).
- All numerical results are the average of three runs.

ID	Task	Type	Goal
1	Cat_Dog	RSR	Analyze how training on different amounts of real and synthetic images effect model performance.
2	Cat_Dog	RSR	
3	Cat_Dog	RSR	
4	Welding	RSR	
5	Welding	Rebalancing	Compare *Synthetic Padding* to other dataset rebalancing techniques on model classification performance.

Figure 3.5: Outline of all experiments for both classification tasks.

4.I. RSR: Curves with increasing Real Images (#1-4)

For the Cat_Dog classification task, as the 8000 training images included more real images, the test accuracy and test loss for the VGG16 model improved quickly before converging as the RSR reached 25:75 (2000 Real & 6000 Synthetic training images). The model converged at ~96% test accuracy and a loss of ~0.860. This is expected behavior since synthetic images can never perfectly capture the feature space of real images but can help expand the feature space (reducing Loss). Notably, at RSR75:25, the test accuracy improved ~1% compared to using 100% real images. In addition, even with low-quality models (~10,000 polygons) the VGG16 classifier reaches an accuracy of ~85% when trained on only synthetic images (a ~10% drop compared to 100% real images).

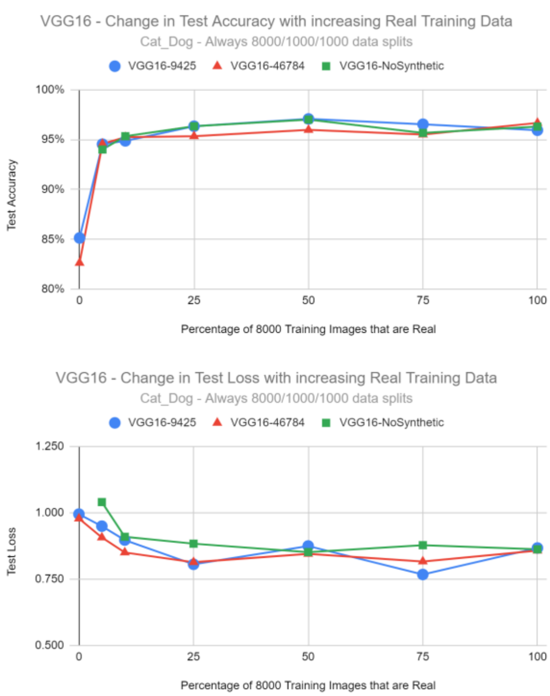


Figure 4.1: VGG16 comparison of three different training sets with their (top) test Accuracy and (bottom) test Loss. 'VGG16-NoSynthetic' uses a training set made of only real images but ranges from 400 images (5%) to 8000 (100%) images. The plots show that, compared to using a smaller training set (green), using additional synthetic images reliably lowers test Loss without a decrease in test Accuracy.

Unlike VGG16, MobileNetV3-small only converged to a single test accuracy or test loss when a higher number of real images are used in the training set. As seen in Figure 4.2, without synthetic images, the model's test accuracy converged to ~90% when training on only 400-2000 real images (5%-25%). With 100% synthetic images (0% real), test accuracy started at 50-60%, then reaches ~81.0% when real images are introduced to the training set (400 real images & 7600 synthetic images). The test accuracy gap between only real and mixed training sets narrowed as more real images are trained on.

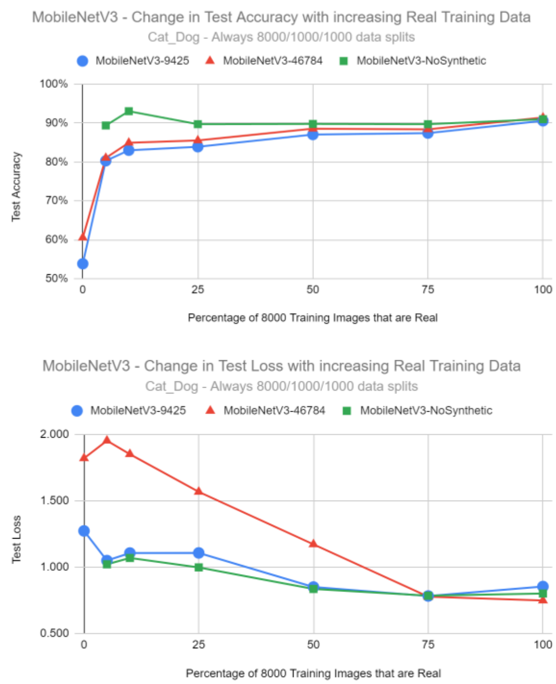


Figure 4.2: MobileNetV3-small comparison of three different training sets with their (top) test Accuracy and (bottom) test Loss. 'MobileNetV3-NoSynthetic' uses a training set made of only real images but ranges from 400 images (5%) to 8000 (100%) images. The plots show that, compared to using a smaller training set (green), using additional synthetic images has a negligible effect on test Loss, but a negative effect on test Accuracy.

Regarding the impact of having more synthetic images to sample (blue line circles vs red line triangles); having more synthetic images to sample shows a negligible effect on performance for the larger VGG16 model and a slight effect on MobileNetV3-small. There are a variety of reasons why having more synthetic data available to train from does not improve model performance. Uniform random sampling of synthetic images can skew the images towards one background, model, lighting, or camera angle. In addition, having more samples requires all those samples to be significantly different from one another. Minor image variations may not contribute significantly enough to the class's feature space.

The Welding classification task's RSR experiment shows mixed results depending on your desired confusion matrix classification. Model confusion matrix performance shows that as more synthetic images are used in an imbalanced training set each feature space is learned differently. More synthetic training samples reduced the high-cost FP classification (stating a weld is *Good* but is a *Defect*), but the binary classifier loses the *Defect* class's feature space (seen by the equalizing of TN & FN). The MobileNetV3-small model's confusion matrix performance shows similar behavior to VGG16. Using only real training images yielded the highest cost per test sample result. As more synthetic images are added to the training set, TP classifications increased, FP decreased, and TN & FNs began to equalize.

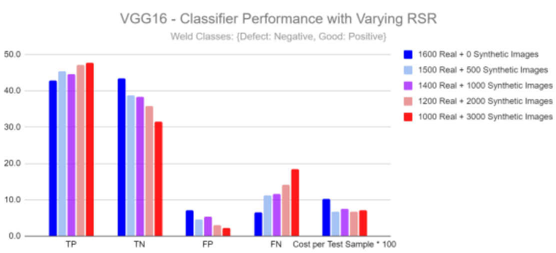


Figure 4.3: Bar chart showing how VGG16 models trained on different sets of training data performed on confusion matrix metrics. {Defect weld: Negative, God weld: Positive}. As the number of synthetic samples increased, the number of TN and FP classifications decreased. The number of FNs increased. As such the Cost per Test Sample decreased as the number since a FP has a much higher cost associated with it.

It is possible that one reason for the equalizing TN & FN rate could be that the *Good* classified *Minor Imperfections* synthetic image variant is not distinct enough from the Defect synthetic images. Thus, reducing the separation of the class feature spaces and making it harder to learn the specific visual features of each class.

Regarding the usability of this classifier to assist inspectors; decreasing the rate of FPs is critical to prevent *Defective* welds from being missed. However, the greater FN rate could reduce the user's confidence in the model.

4.II. Resampling to Balance Datasets (#5)

When handling an imbalanced classification dataset, there are several techniques to help prevent the classification model from skewing to much towards the majority class. These methods include (but are not limited to) Cost-Learning via training weights, undersampling, and oversampling. This research extends research done by Dr Gary Weiss, et al by investigating another method called Synthetic Padding, which adds synthetic images to the minority class in the imbalance. All four of these methods are applied to rebalance the imbalanced Welding training dataset. Specifically comparing the effect on model classification rates and resulting classification costs when the model is trained to maximize Test AUCROC [Weiss].

When trained with the VGG16 model, all resampling methods lowered model cost but had different effects on each type of classification. Oversampling decreased cost the most, but had the largest increase in costly FPs. Cost-Learning showed similar performance to the models in the previous section with a greater definition of the *Good* feature space, but the model lost understanding of the *Defect* feature space. The MobileNetV3-small model shows an opposite effect for the undersampling, oversampling, and synthetic padding methods. In the MobileNet tests, the model learned the *Defect* (negative) feature space to improve TN & FN performance over using an unweighted and unbalanced dataset.

Method	TP	TN	FP	FN
No Change	0	0	0.5	0.5
Cost Learning	0	0	0.667	0.334
Undersampling	0	0	0.5	0.5
Oversampling	0	0	0.5	0.5
Synthetic Padding	0	0	0.5	0.5

Figure 4.4: Matrix showing the cost of making each classification for the Resampling Experiment. 'No Change' has uniform error weighting but the training set used is imbalanced with 1200 Defect images & 600 Good. Methods undersampling, oversampling and synthetic padding have balanced training sets (1200:1200).

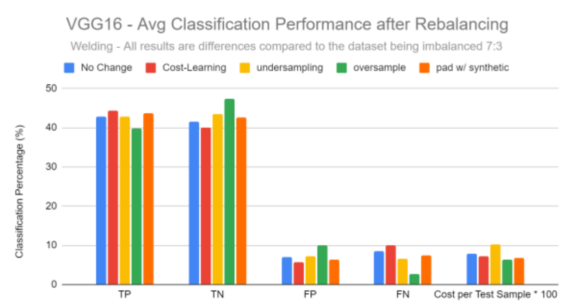


Figure 4.5: Confusion Matrix classifications for each rebalancing method. Only undersampling increased Cost per Test Sample.

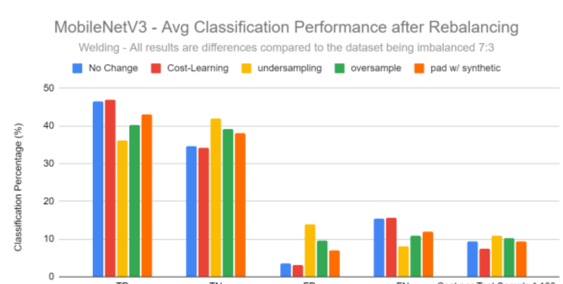


Figure 4.6: Confusion Matrix classification for each rebalancing method. Only oversampling decreased Cost per Test Sample.

5. Contributions

This work contributes to research related to using synthetic images in model training. First, having low quality 3D models (<10,000 polygons) can replicate parts of the feature space of complex shapes (animals) and minor alterations (welding defects). It is possible to train CV models on only low-quality 3D model images and produce significant classifier accuracy (as shown in Figure 4.1 when training with 75% real samples). Second, CV models with millions of parameters can train on small amounts of real images and yield significant classifier accuracy. This supports an alternative for needing thousands of images to train an adequate classifier. Lastly, if large amounts of real images are available, this research also shows that low-quality 3D synthetic images can reduce model loss and possibly increase model accuracy.

6. Conclusions

Variation. The key to a successful CV classifier is having a dataset with a diverse feature space to learn from. Through the documented RSR experiments, this research can conclude that there are benefits to supplementing a CV training set with synthetic Unreal Engine images. For particular RSRs, replacing training set real images with synthetic images can decrease your CV model's loss and narrow the training-test accuracy gap. This also allows developers to move valuable real samples from the training set to the validation & test sets. Investing in processes to produce synthetic data can reduce the work, cost, and risk of collecting real images manually.

Future research can expand upon several areas. First, how to reduce the domain gap between real and synthetic images, such as the relationship between classifier accuracy and factors such as model resolution and using UE pre-built environments. Second, these tests can be repeated to explore how synthetic images can introduce object variations unknown to your model (reducing the Curse of Dimensionality). Finally, work can be done to improve the image pipeline software to create images with bounding boxes that can train more advanced CV models.

6.I. The Importance of Synthetic Image Variation

This research agrees with previous research published by Dr Thornström, Dr Gastelum, and the ECCV submission stating that synthetic image variation is the key to training on synthesized images. Producing large quantities of synthetic images with small variations does not guarantee feature space expansion, rather images with significant variations contribute to model classification. Based on these results, it is the author's opinion that synthetic image Quality, rather than Quantity, contributes to feature space expansion. This work, and the others cited, suggest that future developers may see more success if they approach gathering varied images from a different perspective than capturing every statistical combination of image variation. Proportional sampling of generated synthetic images, producing high quality & significantly different images, and using various UE pre-built environments for synthetic image capture may increase synthetic image effectiveness.

Please consider the following recommendations to improve collected synthetic data. First, utilize randomness in every facet of 'setting up' a screenshot (like the Welding collection), instead of collecting images on a set track with hardcoded values (like the Cat_Dog collection). Second, in addition to sampling the synthetic collections proportionally (like what was done in this paper), a developer should proportionally sample the collection's meta-data (model, lighting, etc). Proportionally sampling based on the collection's meta-data can help prevent the oversampling of one 3D model over another to maintain image variation. Lastly, try incorporating more expansive UE environments during image collection. Only the UE test environment is used in this experiment, there are fully fleshed out landscapes, cities, and environments ready to insert the desired 3D models into.

6.II. Commentary on using GANs for Image Production

Regarding the use of GANs in synthetic image creation. As of the date of publication, open-source tools can produce highly detailed images with user provided text prompts that can be locally saved. In-house developed GANs can be trained to provide slight variations, such as the ECCV paper's use to improve the image 'realism' of created synthetic images [ECCV]. This is an alternative to this report's process of creating a UE environment then setting up the pipeline to generate synthetic images. It is recommended that developers understand the workflow and business strengths and risks of using GANs or UE environments for their image pipelines.

GANs produce images by selecting from learned distributions of features, thus creating pseudo-random images that are difficult to recreate (once trained, fast, but low reproducibility and control). If a third-party GAN is used, consider how to retain images and create more if the API were to suddenly change (see API changes for Reddit and Twitter in 2023). Image pipelines based on UE environments allows users to reproduce images using the exact variations that they want and offers a nimble way of generating new images as customer requirements change (fast & high reproducibility, but requires prior knowledge of rendering engines). Both methods have pros and cons.

Source Code

Please see the following Git repository for all source code related to this report. The repository includes 1) the python base class to implement a UE python callback, 2) proof of concept python scripts to edit a UE environment for any reader's reference, 3) python scripts to train the VGG16 and MobileNetV3-small model and 4) any other assisting scripts.

Link:

https://github.com/smutnyjw/training_cv_models_on_ue_images_jws

Appendix

A.I. Supporting Tables

Image Type	Cat	Dog	Good Weld	Defect Weld
Real	12,500	12,500	600	1400
Synthetic	23,932	23,932	3,264	4,896

Figure A.I.1: Class distribution of the used datasets for both the Cat_Dog and Welding Defect Detection use cases. Numbers show the number of found real and generated synthetic images.

Model Classification	Actual Classification	Meaning
Positive	Positive	Predicted and a *Good* weld
Positive	Negative	Predicted *Good* weld but *Defect*. **(HIGH COST)**
Negative	Positive	Predicted weld *Defect* but *Good*
Negative	Negative	Predicted and a weld *Defect*

Figure A.I.2: Description of the Welding Defect Detection classification terminology. In this research, a Good weld is a POSITIVE classification and a Defect is a NEGATIVE classification so that the models can train to maximize AUCROC metrics that tune for a high TP and low FP rate.

A.II. RSR Experiments – Additional Graphs

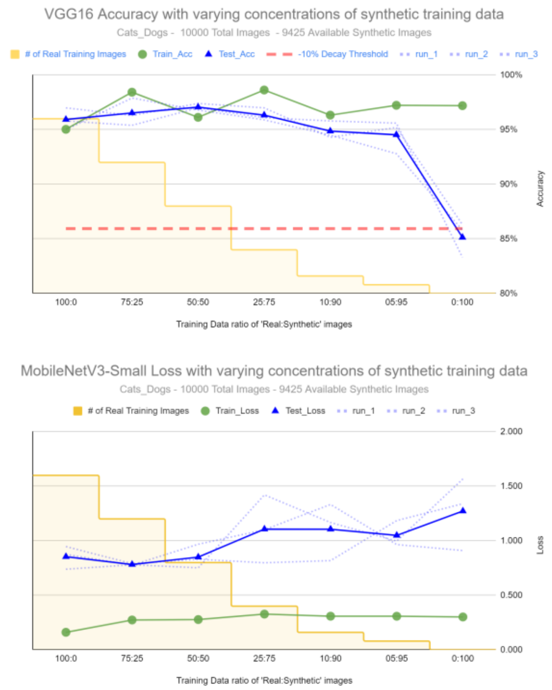


Figure A.II.1: VGG16 Test & Train Accuracy & Loss plots as the ratio of real training images decreases when 9425 synthetic images could be sampled.

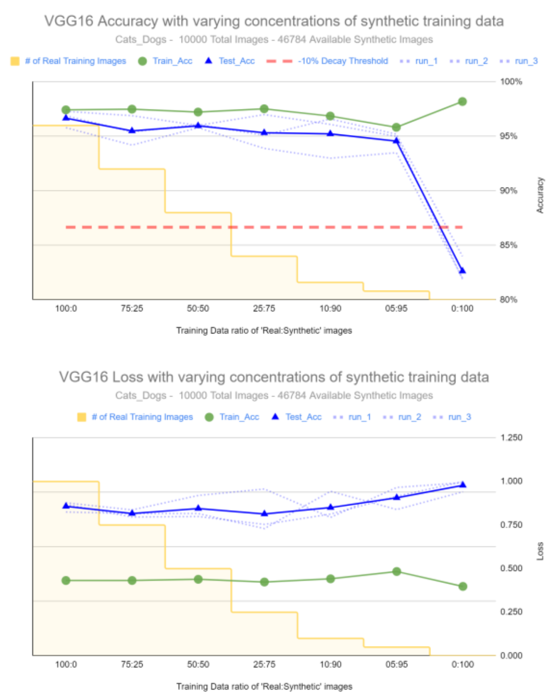


Figure A.II.2: VGG16 Test & Train Accuracy & Loss plots as the ratio of real training images decreases when 47864 synthetic images could be sampled.

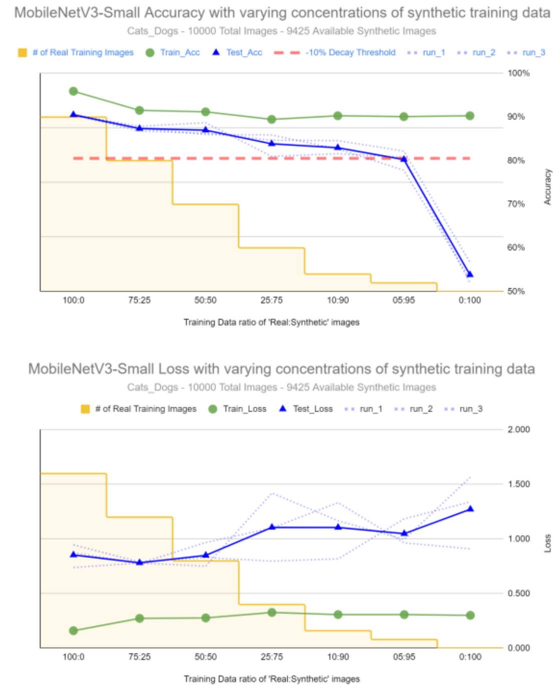


Figure A.II.3: MobileNet Test & Train Accuracy & Loss plots as the ratio of real training images decreases when 9425 synthetic images could be sampled.

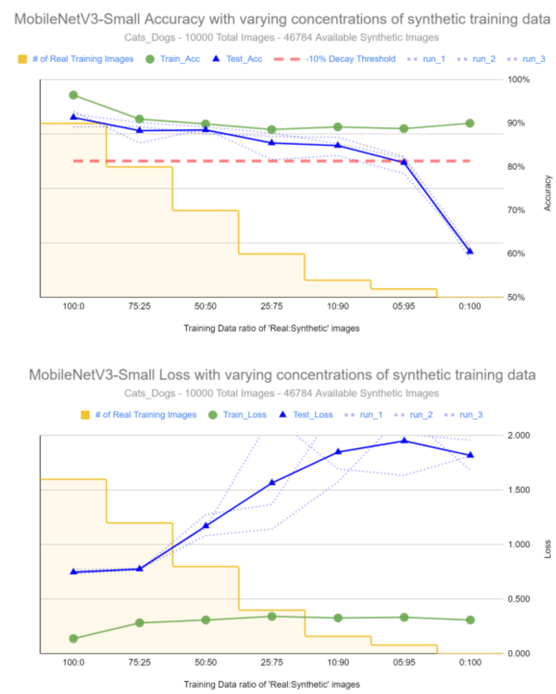


Figure A.II.4: MobileNet Test & Train Accuracy & Loss plots as the ratio of real training images decreases when 47864 synthetic images could be sampled.

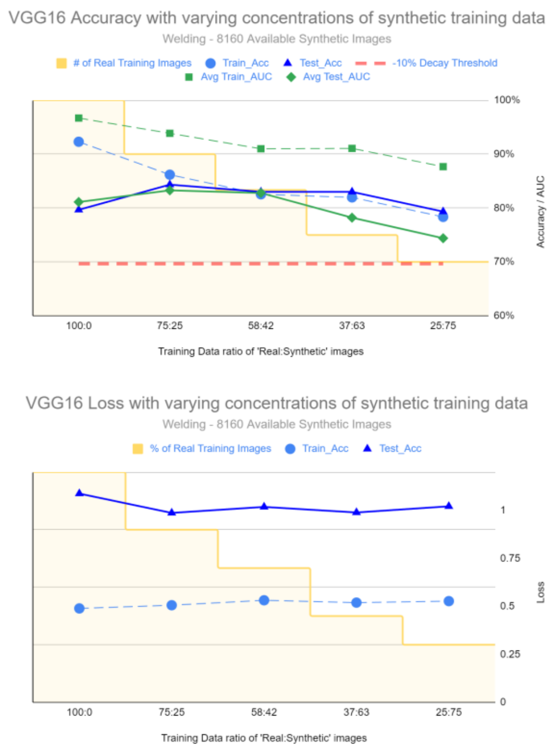


Figure A.II.5. VGG16 Accuracy, AUC, and Loss when training on an imbalanced Weld defect training set as the ratio of real training images decreases.

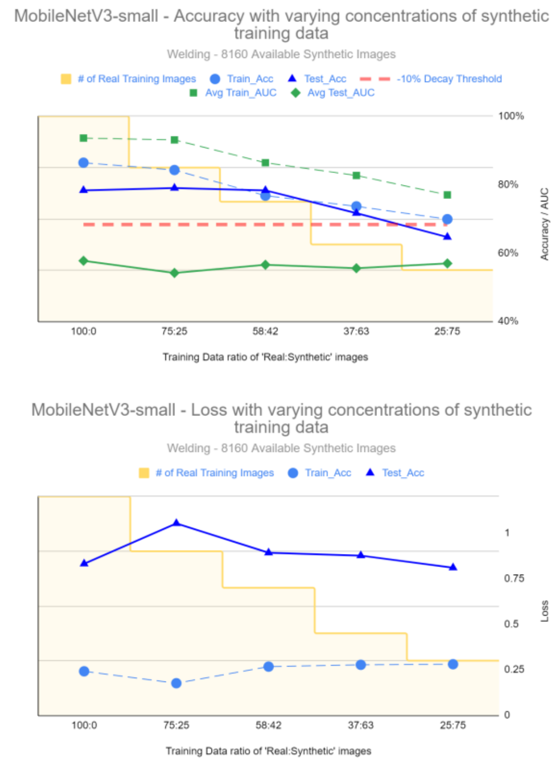


Figure A.II.6. VGG16 Accuracy, AUC, and Loss when training on an imbalanced Weld defect training set as the ratio of real training images decreases.

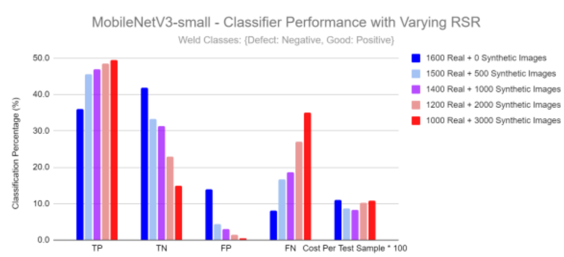
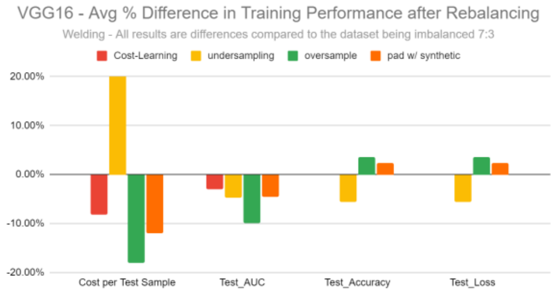


Figure A.II.7: Bar chart showing how MobileNetV3 models trained on different sets of training data performed on confusion matrix focused metrics. {Defect weld: Negative, Good weld: Positive}. Highlights how more synthetic samples in the dataset skewed the MobileNetV3 models to classify most samples as Positives (aka. Good welds)

A.III. Rebalancing Experiments – Additional Graphs



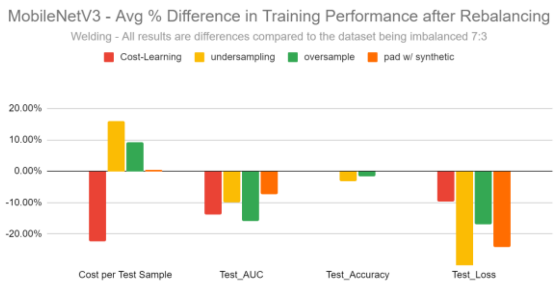


Figure A.III.1. VGG16 and MobileNet bar charts showing how the various rebalancing methods for Weld Defect Detection affected training and testing metrics. Graphics show that the effectiveness of each resampling method depends on what metrics are most important to your use case.

Acknowledgments

Thank you to Dr Creed Jones, Dr Ryan Williams, and Dr Jason Xuan for being a part of my committee, especially to Dr Jones for giving me their ear and guidance throughout this project.

Additional thank yous to …

Northrop Grumman Corporation for funding my Master's Degree at Virginia Tech University.

To the numerous teachers that I consulted with throughout this project. To Brandon Padayao with Baltimore OpenWorks for allowing me to sit in on a welding course and learn from you about metal welding. To Riley Van Etten and Deion Waddell for their expertise in 3D graphic design. To numerous others helping me brainstorm through ideas and barriers.

This project gave me the chance to learn about one of the most powerful graphics engines - one of many that fueled my imagination as a child.

References

R.I. Primary Sources

3D Ace. “Does Polygon Count matter in 3D Modeling for Game Assets?”. Nicosia, Cyprus. Jan 3rd, 2024. https://3d-ace.com/blog/polygon-count-in-3d-modeling-for-game-assets/

DowlingSoka, Ryan. “Python in Unreal Tips”. Published on Mar 18th, 2024. https://ryandowlingsoka.com/unreal/python-in-unreal/

ECCV-18 Unknown author submission ID 2903. “Learning image classifiers from (limited) real and (abundant) synthetic data.” Semantic Scholar. 2018. Corpus ID: 53472409. https://www.semanticscholar.org/paper/and-(abundant)-synthetic-data/1f759dbd74ad3907457dcbf50d4be54b470e33f4. Citation possibly for Nguyen, Tan, Chen, H, et al. “Learning image classifiers from (limited) real and (abundant) synthetic data.” Published by Rice University. https://tannguyen.blogs.rice.edu/current-research/. 2018.

Gastelum, Zoe Nellie, and Shead, Timothy. How Low Can You Go? Using Synthetic 3D Imagery to Drastically Reduce Real-World Training Data for Object Detection. United States: N. p., 2020. Web. doi:10.2172/1670874.

He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

Howard, Andrew, et al. "Searching for mobilenetv3." Proceedings of the IEEE/CVF international conference on computer vision. 2019.

Man, Keith, and Javaan Chahl. “A Review of Synthetic Image Data and Its Use in Computer Vision.” Journal of imaging vol. 8,11 310. 21 Nov. 2022, doi:10.3390/jimaging8110310

OpenWorks, Baltimore City. Mar 16th, 2024. Metal 3 with Brandon Padayao. Articles: “Metal 3: MIG Welding” and “Metal 3: TIG Welding”. 1400 Greenmount Ave, Baltimore, MD 21202

Padayao, Brandon. “Metal 3”. Baltimore Openworks. Mar 16th, 2024.

Thornström, J. “Domain Adaptation of Unreal Images for Image Classification (Dissertation).” Linköping University Department of Electrical Engineering, Computer Vision (Sweden). 2019. URN: 165758.

Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

Weiss, Gary M, McCarthy, Kate, and Zabar, Bibi. “Cost-Sensitive Learning vs. Sampling: Which is Best for Handling Unbalanced Classes with Unequal Error Costs?”. Department of Computer and Information Science, Fordham University. 2007. https://storm.cis.fordham.edu/~gweiss/papers/dmin07-weiss.pdf

R.II. Datasets

15DEM20F. “Welding Images Dataset”. Dataset 01. Publisher Roboflow. Roboflow Universe. Jul 2022. https://universe.roboflow.com/15dem20f-ab0x1/welding-images

afiys. “Proyek Akhir - OFA5 Dataset”. Dataset 01. Publisher Roboflow. Roboflow Universe. Oct 2023. https://universe.roboflow.com/afiys-oxbaf/proyek-akhir-ofa5

Celebal Workspace. “Welding_Detection_v2 Dataset”. Publisher Roboflow. Roboflow Universe. Dec 2022. https://universe.roboflow.com/celebal-workspace/welding_detection_v2

Elson, Jeremy and Douceur, John (JD) and Howell, Jon and Saul, Jared. “Cats vs Dogs”. Proceedings of 14th ACM Conference on Computer and Communications Security (CCS). October 2007. Published by Association for Computing Machinery, Inc. Distributed by Microsoft. https://www.microsoft.com/en-us/download/details.aspx?id=54765

Welddefect. “Weld_detections Dataset”. Publisher Roboflow. Roboflow Universe. Jul 2023. https://universe.roboflow.com/welddefect-4pvic/weld_detections/dataset/1

Weld 1. “Final Dataset Dataset”. Dataset 01. Publisher Roboflow. Roboflow Universe. April 2023. https://universe.roboflow.com/weld-1-mba6s/final-dataset-rc2vm