RESEARCH REPORT
ANIMAL SCIENCE

Title: Developing an automated, real-time pig counting technology - **NPB # 19-080**

Investigator: M. D. Tokach

Co- Investigators: J. C. Woodworth, A. B. Lerner, K. F. Coble, S. Blazevich, J. M. DeRouchey, S. S. Dritz, R. D. Goodband, K. Duggal

Institution: Kansas State University

Date Submitted: 7/29/2020

Industry Summary: The development of artificial intelligence has allowed computers to be trained to complete a wide range of tasks, from simple to complex in nature. This is done by creating a model, or a set of algorithms (codes which tell the computer what to learn and how to make decisions on information) to solve a given problem. The goal of this project was to develop an automated system to count pigs. To begin, videos were collected of moving pigs by hanging a camera phone from the barn ceiling to get a top down view of pigs walking underneath the camera. These videos were then split into thousands of images. In each image, a computer scientist drew a box around the pig head. After feeding the images back to a computer, it began to "learn" how to identify pigs on its own. This is important because it needs to know how to differentiate a pig from the floor, wall, human handler, rattle paddles, etc. The next part of process involves the computer "watching" the subsequent images as pigs walk by and counting the number of pigs that pass underneath the camera. An equation was added to the model to adjust the running count if pigs turned around. Equipment needed for the system described in this report is 1) a mobile phone to run the user application, 2) an iPhone to hang above the alleyway where pigs will walk, 3) a small, screenless computer, and 4) a router. The router allows all the devices to communicate with each other on a local network, without having to useexternal internet. The mobile application allows the user to "start" and "stop" the count and displays the live video of moving pigs as well as the count on the home screen. Challenges that limited how well the computer was able to count pigs included pigs moving on top of or very close to each other, pigs moving at high speed, and changes in lighting. Further development (gathering more images to train the model) is needed on this foundational solution prior to being ready for use in commercial swine production.

For more information, please contact:
Name: Dr. Mike Tokach
Title: Extension Specialist, Swine
Address: 246 Weber Hall, Kansas State University, Manhattan, KS 66506
Email: mtokach@ksu.edu

Keywords: counting, technology, machine learning

Scientific Abstract: Three prototypes were developed and evaluated to assess the accuracy of an automated pig-counting solution. Key components of these prototypes consisted of hardware (camera and edge compute device), software (image processing, object detection, object tracking, and object counting), and a supporting mobile application. Prototype 1 utilized a YOLOv3 model ran on an NVIDIA Jetson TX2. While the model was functional, the edge device was not powerful enough to process images in real-time. Prototype 2 utilized an edge device with greater processing capability (NVIDIA AGX Xavier) with an SSD MobileNetV2 model. With this hardware, inferences were completed in real time. This second prototype also evaluated three cameras, of which the iPhone X was selected as the image gathering hardware due to its image quality and ability to adjust to various lighting scenarios, which improved overall accuracy of the model. Due to the fact that multiple cameras were still being evaluated, this prototype was only tested on a limited number of pigs. For the final and third prototype, additional model training was completed, and the model was optimized for the edge device and camera chosen. Percent of pigs counted ranged from <50 to >100% with an average of 79%. Several suggestions are detailed within this report to resolve challenges that decreased accuracy. With the rapid advancement of machine learning models and sophistication of hardware, we believe that an accurate pig counting solution is feasible, and that the prototype developed within this project can serve as a foundation for the development of that solution.

Introduction

Accurately counting pigs is a challenge in the U.S. swine industry. While the true margin of error in livestock counts is difficult to quantify, it is well understood that inaccurate counts of livestock inventory can significantly impact profitability and efficiency of labor. Imprecise pig counts may result in loss of revenue when pigs are bought or sold and inappropriate management decisions regarding nutrition, health, and marketing of pigs. Oftentimes, pigs are counted as they are moved between facilities within multi-site production systems, where production staff may also be performing other tasks such as vaccinations and sorting.

Automated animal counting is developing in a variety of species. In wild or migrating animals in which the lack of human proximity makes it difficult to obtain accurate counts, automated counting can be especially useful. Populations of wild ducks have been counted via decoy simulation and which demonstrated that drone images counted with semi-automated machine learning could have increased accuracy when compared with counts generated by human observers.[1] Several researchers have sought to use image processing and automation to track moving pigs for identification, behavior and welfare, or health purposes.[2,3,4] The objective of this project was to develop a pig counting model to be implemented with camera hardware and an edge computing device to automatically and accurately count pigs as they are moved during

normal production processes. Further, this process is designed to occur in real-time, with information deployed to various users, such that this solution could also be utilized for inventory data management.

Objectives

The overall objective of this project was to develop an automated, real-time, pig counting technology for all sizes of pigs.

The specific objectives of this project were as follows:

1. Develop a technology that can identify pigs and train the associated algorithms via machine learning to deliver a "count" of the animals that passed through the checkpoint.
2. Validate this technology in various production environments such as receiving and loading out of nursery and finishing barns, as well as market hogs received at the packing plant.
3. Develop a user-friendly interface for the technology that has the capability to interface a system of devices.

The intended outcomes included a successful pilot architecture comprised of a system of sensors that utilize "internet of things", machine learning, and edge capabilities to accurately count pigs being loaded or unloaded within a pork production system. A mobile application also was to be developed where information could be easily interpreted in real-time. A clear objective during the process was to develop a system at a price point that would make it accessible and implementable throughout the swine industry.

This report is intended to be a high-level overview of the pig counting solution. Appendix A and B are also provided in order to provide additional technical descriptions of the hardware, software, and algorithms to those seeking more details.

Building a team of experts, orientation on current status of pig counting, and aligning on a common goal

This section describes the key players and roles within the team, communication, orientation, and framework of the project.

Introduction to project and creation of innovation team

Due to the complex and interdisciplinary nature of this project, the first task to complete was orientating the core team that would be working on the project. The team consisted of several researchers from KSU, software engineers, computer scientists, and machine learning experts from Amazon Web Services (AWS), and production specialists from JBS. TensorIoT is another technology company that works closely on internet of things (IOT) and artificial intelligence (AI) projects with AWS, specializing in developing solutions that involve edge computing. They were added to the project by AWS. In addition, several team members from each partner were involved from a project coordination standpoint. In all, there were 10 to 15 core members of the team, with an additional 30 to 50 employees from all organizations that were involved to complete various steps in the project. The amount of collaboration and organization from

employees from all levels of each participating partner in this project cannot be understated. Specifically, the eagerness and willingness from the JBS organization to invite us into their facilities along with the challenges that a development-type project brings was outstanding and allowed for significant learning gains.

Communication

A KSU doctoral student, Annie Lerner, was appointed as the project manager, and was largely responsible for keeping the KSU, AWS, TensorIoT, and JBS counterparts aligned and in communication. This role was essential to the progression of the project due to the majority of the team working from different locations. Additionally, because of the unique sectors in which the various team members operated (swine production, technology development, and academia), an effort needed to be made often to "translate" information between fields. Specifically, technology development and swine production fields each have particular, highly technical jargon and nuances, and for the project to succeed everyone had to become at least familiar with certain aspects of the other field.

Due to the remote working environments and large team, communication occurred primarily through emails and biweekly conference calls. Particularly during the development periods between onsite visits, AWS and TensorIoT would update the group on the previous two weeks' outcomes, challenges, and progress.

This document contains highly technical jargon that is frequently utilized in the machine learning field. A glossary of frequently used terms, along with accompanying examples, can be found below.

Glossary of technical terms		
Term/Acronym	Definition	Example
Application Program Interface (API)	A toolset that programmers can use to help them create software. Simply put, an API specifies how software components should interact.[1]	A weather snippet on a smart phone uses API to display information from the database of a weather application.
Artificial intelligence (AI)	Artificial intelligence (AI), the ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings.[2]	A drone that surveys a field, processes images, and determine which areas need improvement.
Convolutional neural network (CNN)	A Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in	An algorithm used in facial detection, object classification for self-driving cars, hand-writing recognition, etc.

	the image and be able to differentiate one from the other.	
Edge device	Edge computing solutions place Internet of Things (IoT) devices, gateways, and computing infrastructure as close as possible to the source of data—and to the systems and people who need to make data-driven decisions.[4]	Autonomous tractors that can obtain and process information about the environment remotely, without having to connect to a network or server to make inferences.
Hypertext Transfer Protocol (HTTP)	A protocol used to transfer data over the web. It is part of the Internet protocol suite and defines commands and services used for transmitting webpage data.[5]	
Internet of things	The concept of connecting any device (so long as it has an on/off switch) to the Internet and to other connected devices. The IoT is a giant network of connected things and people – all of which collect and share data about the way they are used and about the environment around them.[6]	Collars fitted for dairy cattle that track activity.
Real-time transport protocol (RTP)	A network protocol which described how to transmit various media (audio, video) from one endpoint to another in a real-time fashion.[7]	
Real time streaming protocol (RTSP)	An application-level network communication system that transfers real-time data from multimedia to an endpoint device by communicating directly with the server streaming the data.[8]	

[1] https://apifriends.com/api-management/what-is-an-api/
[2] https://www.britannica.com/technology/artificial-intelligence
[3] https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53
[4] https://www.intel.com/content/www/us/en/edge-computing/edge-devices.html
[5] https://techterms.com/definition/http

[6] https://www.ibm.com/blogs/internet-of-things/what-is-the-iot/
[7] https://developer.mozilla.org/en-US/docs/Glossary/RTP
[8] https://searchvirtualdesktop.techtarget.com/definition/Real-Time-Streaming-Protocol-RTSP

Travel and onsite development

The first onsite team meeting involved bringing the technology experts from AWS onsite to several JBS facilities including: a weaned pig transfer station that processes 4,000 to 7,000 pigs per day, wean-to-finish barns, and a pork processing facility. These visits were critical to orient the AWS team members to the nuances of raising, moving, and counting pigs. It was most of the AWS team member's first experience to large-scale livestock production. In order to properly prepare them for each environment, we had prior discussions detailing the sights, sounds, and smells that they would encounter before going into each facility. This alleviated some of the questions that can occur upon the first-time seeing animal agriculture and allowed them to focus on the technicalities and factors of the environment that pertained to developing pig counting technology. Though this step was not necessarily detailed in our initial project planning, it was highly critical to successfully initiate the project and align all team members to the objective at hand.

Several trips onsite to JBS facilities followed, where a KSU researcher would be accompanied by one or two AWS experts. In all, six onsite visits were completed over a period of 11 months. These trips ranged from 3 to 5 days. Two on-sites were completed with the primary objective of capturing video data for observation and model training. Remaining on-site visits to the weaned pig transfer station were conducted to test prototypes.

Initial observations and challenges in pig counting

From onsite visits, AWS observed five main challenges in video-based pig detection and tracking:

1. Sudden light changes that occur in the barns or different illumination during the day and night or when the lights are on/off;

2. Different backgrounds (color, shapes, objects, etc.) where the counting needs to be done;

3. Occlusions and/or shape deformation of pigs impact the ability to consistently track pigs individually across all frames;

4. Pigs are often very close to each other and large in number in a single frame, causing overlapping bounding boxes (described below), which makes labeling a difficult and time-consuming process, and challenges object detection algorithms;

5. Pigs don't always move in a single direction as they move back and forth through the corridor impacting the tracking and counting algorithms.

Pursuing the weaned pig use case and vision for solution

After surveying the entire pig production system and the variation in pig size, facility type, and scale of production, AWS suggested that the team focus on prototype development using the weaned pig and specifically the JBS weaned pig transfer station (WPTS). This decision was made in part by the evaluation that weaned pigs are often the most challenging to count, due to their faster and more crowded movements. Additionally, with the WPTS receiving 4,000 to 7,000 piglets per day, this large volume allowed for large scale and rapid testing (compared to a sow farm or a nursery barn that would typically send or receive fewer piglets with less frequency). Further, we hypothesized that training the model with a more specific set of images (weaned pigs only vs. pigs of all sizes) would make the model more accurate in that given scenario. Then, additional data or models for growing and finishing pigs could be based off of the initial weaned pig model. Focusing resources and efforts on this single use case allowed further progress to be developed than the approach of tackling multiple pig sizes with one model.

Following the initial onsite visits and running preliminary identification and counting models on all sizes of pigs, AWS concluded to set up a system based on the following model:

> Real-time pig counts determined from raw image capture based on the following inference flow:
>
> 1. A camera mounted to obtain a top view of the pigs at the point where they are to be counted capturing video in 1080 pixel resolution at 30 frames per second. Potential mount points are near the doors in the hallway where pigs walk mostly in one direction and less occlusions occur.
> 2. This real-time video is streamed via a Real Time Streaming Protocol (RTSP) application to the pig counting application. The image processor module in the IoT Edge hardware will normalize the size of the video and split the video into 512 x 512-pixel sized frames. These frames are sent to the object detection model to make inferences.
> 3. The pig detection function uses an object detection model to perform inference on each frame to detect all the pigs present in that image. The output of this function is the list of bounding boxes for every image.
> 4. The direction tracker maintains the state for every ten frames to capture the direction in which each of the bounding boxes aremoving.
> 5. The aggregated counts can be calculated from the time series data based on when the counting was started to when the counting was stopped.

Prototype Development

The approach taken to develop a testable solution took place over many months and resulted in three distinct prototypes. Once each prototype was ready, it was evaluated in the weaned pig

transfer facility. At each of these visits, challenges, successes, and observations were recorded to perform further iterations for the next prototype.

Prototype Components

The following section describes the development and details of individual components of the pig counting system. Additional technical information is provided in Appendix A. Figure 1 illustrates the complete pig counting system.

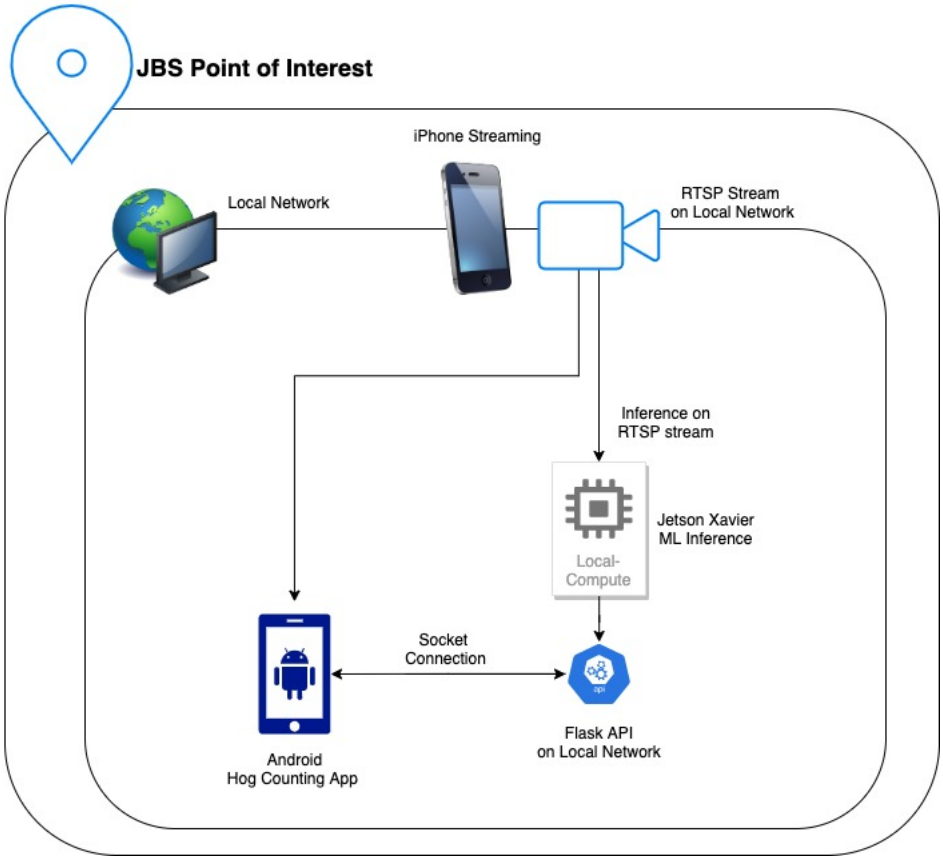

Figure 1. Various components of the pig counting system. JBS point of interest could be considered the barn. An edge device contains hardware and local network needed to share the video of the pigs between the iPhone and computer to provide real-time counts in the pig counting app on the android device.

Hardware

Development, requirements, and testing

This section describes the hardware selection process to implement the prototype solution. There are four main components required for the solution: an input camera, an edge gateway device, a device hosting a user interface. Provided all three devices will be in range, a

standard WiFi router is sufficient for support network hardware to connect the devices via the same local network.

Camera

The automated pig counting workflow starts with a digital camera providing a live video stream. The image source device should be capable of capturing and streaming video over a network or via direct connection to the edge device where inferences are performed. If an IP camera or a phone is used, video should be streamed as MJPEG/H.264 over RTP, RTSP, HTTP. A video capture client can then connect to the stream and convert it to frames that can be used for inference.

The major considerations for choosing a camera include:

1. Objects that need to be inspected, in this case pigs

2. Capture frame rate required, i.e. how fast is the object moving

3. Image resolution required

4. Performance across expected environmental conditions, such as lighting, temperature, humidity, etc.

5. Network connectivity (local network and/or Internet)

6. Cost of portable or fixed installations

Several cameras (IP camera, camera on raspberry pi, multiple iPhone cameras) were evaluated to understand how image quality impacted model accuracy. The iPhone X (dual lens, 12 megapixel) was selected as the hardware that streams video of pigs for the rest of the network to use. Choosing the iPhone X came down to a few key components:

- Dual lens camera
- Auto-adjusting for adverse lighting conditions
- Auto-adjusting for focus on main objects in frame

Edge Gateway

The IoT Edge Gateway hosts the portion of the pig counting solution that runs on-site. This includes ingestion of video streams, frame extraction, machine learning inference, and IoT Cloud interface.

The video processing and machine learning inference functionality are the major drivers in edge gateway device selection for this application. Based on the object detection and tracking requirements, the edge gateway device needs to provide GPU support. Cloud

interface and solution deployment was handled by AWS IoT Greengrass, which is largely device independent.

The NVIDIA Jetson AGX Xavier was selected as the hardware for the edge gateway. The Xavier is a powerful computing device well equipped for heavy image computation, boasting a 512 GPU Core board. It has the ability to analyze a video stream of pigs in real-time, approximately 50 frames per second (FPS), with an optimized Tensor RT model.

The Jetson Xavier doesn't come packaged with the ability to connect to a Wi-Fi network. To do so, an M.2 Wi-Fi card and antennae were installed on the Xavier.

Device hosting user interface

User interface (mobile application) is designed to run on an Android device. From this mobile device, the user can start or stop pig counting processes, as well as view the live stream of pigs being counted and generate a running count.

Software

This section describes the software components required to implement the prototype solution. There are four main components: image processing, object detection, object tracking, and object counting.

Image processing

Image processing obtains source video in real-time and splits it into individual image frames using a library such as OpenCV. The images are resized to a particular width and height required by the downstream machine learning models. Image processing is implemented on the edge gateway device.

Object detection

Object detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain class (such as humans, buildings, animals, or cars) in digital images and videos. Object detection is used to identify all the pigs in a given image and to output pixel coordinates of the bounding boxes of all pigs detected.

Several different methods of object detection were investigated (Figure 2). Object segmentation where the whole pig is identified by shape rather than a crude rectangle is very costly time wise and computationally. Bounding boxes around the pig body or pig head were also tested. The bounding box of the pig head was as accurate as the object segmentation model with less computational expense and was thus selected.

An object detector is typically more computationally expensive, and therefore slower, than an object tracking algorithm. Examples of object detection algorithms include Haar

cascades, HOG + Linear SVM, and deep learning-based object detectors such as Faster R-CNNs, YOLO, and Single Shot Detectors (SSDs).

The model was created in a SageMaker environment and deployed onto the edge device using Greengrass. A Greengrass Lambda listens to the messages published by the image processing routine and runs object detection inference against it. It sends the list of bounding boxes for every frame as a local MQTT message to another topic that triggers object tracking.

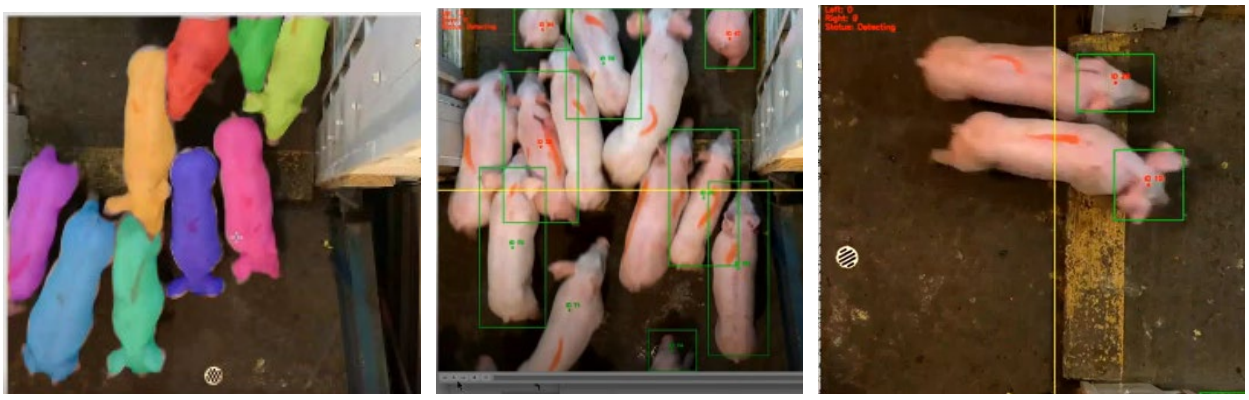

Figure 2. Examples of object detection with segmentation or bounding boxes on the body or head.

Object tracking

Object tracking entails locating an object in successive frames of a video by assigning it a unique ID and tracking it as it moves in a video stream. The object location in the next frame is predicted based on various attributes of the frame such as gradient, optical flow, etc. Multi-object trackers were used so that multiple pigs could be counted as they cross a threshold (or returned across the threshold).

Object counting

The objective of this component is to persistently track multiple moving objects from frame-by-frame object detection inputs and count how many objects entered, left, or re-entered the field of view. The object detection algorithm doesn’t assign a unique identifier to the detections, which means object motion is unknown frame-to-frame. Object tracking ties those bounding boxes across frames to analyze the direction of the object's motion by using the distance between centroids approach or Intersection-over-Union approach.

This information is used, along with whether the centroid of a given pig has crossed a pre-defined threshold, to understand whether the object left the frame, re-entered the frame, etc. producing object counts.

Final model selection

The final model selected was an SSD MobileNetV2 that utilized 512 x 512 pixels. Optimization was completed on TensorRT framework.

Mobile application

The Pig Counter mobile application is intended to serve as the interface by which the machine learning models deployed on the edge streams the pig videos that the model produces for counts. From a technical perspective, the application is required to communicate with the edge hardware as well as the cloud via APIs and video streaming protocols. From a usability perspective, it was critical that the solution can be supervised and augmented by human operators, and that the application's features be straight forward and intuitive for users of a variety of education levels and backgrounds.

A user can download the app onto their phone or tablet. Once a user has progressed past the authorization forms, they are directed to a simple form for identifying the site at which they will be conducting a count, or run, and the device that will be streaming the video to the application. The user enters metadata about the count, some of which can be drawn from the truck manifest (e.g. voucher number) or context (e.g. loading or unloading), and is provided with a button to begin the count. When the user presses this button, the edge hardware records the metadata and begins streaming video to the model. At the run screen, the user is provided with video streamed from the device along with data about the counts progress. As pigs are counted by the machine learning model, a data visualization represents how many pigs have filled the current pen based on the pen max previously identified by the user, as well as overall progress such as how many pens have been filled and how many total pigs have been counted out of the total expected. Once the user identifies the count has finished, they are taken to a review screen where they are presented with the count details and can either submit the count or edit fields indicating why the expected count did not match the actual count. While this covers the core uses for a standard user, additional features were supplied to support managers and administrators of the application.

Managers can review previous counts run by users at their sites, filter the history by various parameters, or review, edit, and submit counts. To ensure appropriate users only access data relevant to them, there are additional administrator screens for user management and site management. The user management screens allow administrators to modify users' roles, as well as modify the sites they're associated with and have access to. On the site management screens, the administrators can select a site and add or remove the users which are associated with the site.

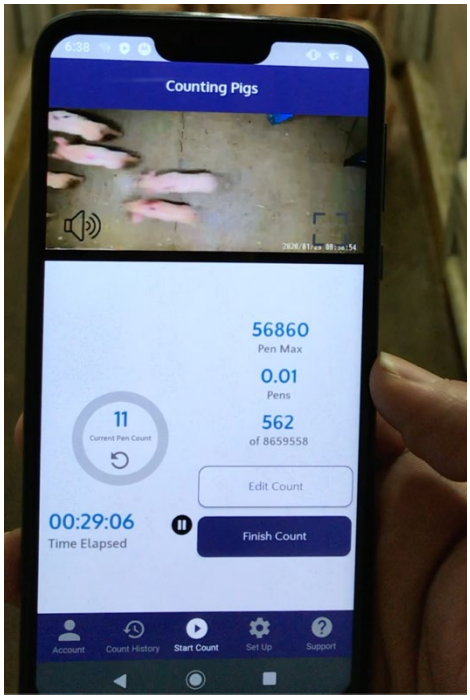

Figure 3. Mobile application. See appendix B for details on the app.

The mobile application has been implemented natively on Android and iOS written in Java and Swift following MVVM and VIPER architectures respectively. Appendix B describes the Android application's features related to each screen. Briefly, the app records site, barn, pen, number of pigs in that pen and total for the barn (Figure 3). It shows the pigs counted or subtracted as they cross the threshold and cumulative time of the counting period. The user can also edit the count if a pig was later removed and to compare the count to expected pig numbers. The results can then be loaded to the cloud to be incorporated into other data management systems.

Prototype results and current status of solution

The following sections outline the summary of the team's work in creating a solution for counting weaned pigs and suggestions for improving the counting accuracy of the solution further. Overall, three prototypes were developed and evaluated, with each iteration using learnings and challenges from the previous to improve the final solution.

Prototype 1

The initial iteration completed the base solution implementation that remained core to all subsequent iterations. This includes cloud and on-premise backend infrastructure, with iterations focusing on the machine learning workflows. Different state-of-the-art object detection models were investigated and analyzed for precision and speed. The models considered included SSD and YOLOv3 with different base networks like ResNet50, MobileNetV1, DarkNet53, ShuffleNetV2, etc. The two top models evaluated were YOLOv3 with DarkNet53 base (most accurate) and SSD with MobileNetV1 base (fastest inference).

Similar to object detection, different tracking algorithms were evaluated for performance when a bounding box was provided in the first frame, and predictions were generated for future frames. For each new detection made, the center of each bounding box is used and compared to the memory of previous bounding box centroids. The Euclidean distance between the previous

frame's centroids and each new centroid is computed and whichever old centroid and new centroid has the shortest distance between points is said to be the same object. For objects not spoken for or with distances over a certain threshold, they take on new ids and are then added to the tracker's state memory. This is done with each successive frame. The methodology that performed best for the project use case is the DLIB correlation tracker with centroid tracking.

Highly accurate object trackers combine the concept of object detection and object tracking into a single algorithm, typically divided into two phases:

Phase 1 – Detection - During the detection phase the computationally more expensive operation is executed to (1) detect if new objects have entered the field of view, and (2) see if objects that were "lost" during the tracking phase can be found. For each detected object, an object tracker is updated or created with the new bounding box coordinates. Due to its computational load this phase is only run once every N frames.

Phase 2 – Tracking - Tracking occurs when the detection phase is not active. For each detected object, an object tracker is created to track the object as it moves around the frame. The object tracker should be faster and more efficient than the object detector. Tracking continues until the N-th frame is reached, at which point the object detector is executed. The entire process then repeats.

Phase 3 – Counting - Once the state of the object is tracked across frames, it can be counted based on centroids of the bounding boxes crossing a given threshold.

Prototype 1 results:

Prototype 1 results were encouraging. Percent of pigs counted ranged widely, but the highest accuracy of the solution was 94.6%. With the NVIDIA Jetson TX2 as the edge device the full object detection, tracking, and counting was performed at 10.4 FPS, which was not "real-time".

Prototype 2

After additional benchmarking, the edge gateway device was changed from an NVIDIA Jetson TX2 to an NVIDIA Jetson AGX Xavier as it provided better real-time performance.

This iteration was implemented with the following configuration:

- **Models**
  o SSD MobileNetV2
- **Image size**
  o 512 x 512 pixels
- **Edge Device**
  o NVIDIA AGX Xavier
- **Cameras**
  o iPhone X (dual lens), iPhone XR (single lens), Vivotek IP camera

- **Optimization**
  o TensorRT framework

The input from the camera was an unobstructed top view of the pigs, which captured video in 1080p resolution and 30 FPS. Placement points were near the doors in the hallway where pigs walk mostly in one direction, and less occlusions occur. For this iteration, the camera was placed right after the vaccination station in the weaned pig transfer station.

1. Real-time video was streamed via an RTSP application to the pig counting application installed on the Jetson Xavier.
2. Image preprocessing normalized the size of the video and split the video into 512 x 512 pixel frames. These frames were sent to the object detection model to perform inference.
3. Object detection was inferred on every frame to detect pig heads present in that image. The output of this function was a list of bounding boxes around all pig heads for a given image.
4. Object tracking maintained the state of all of the detected pigs for every 10 frames to capture the direction in which each of the bounding boxes was moving. Once the bounding box crosses the threshold in the center of the view, the count was increased by one for movement from left to right and vice-versa.

Prototype 2 results:

The highest accuracy observed for one truckload of weaned piglets was 94.38%, but ranged from 76 to 99.10% for smaller groups of 50 to 150 piglets.

Three different cameras were tested in real-time along with various combinations of lighting to observe the impact to accuracy. Tests were performed over three days, with the following results for the final day:

	Total Predicted			Difference, head		Accuracy, %	
	iPhone	IP Cam	True Count	iPhone	IP Cam	iPhone	IP Cam
Truck 1	1310	1217	1434	124	217	91.35	84.87
Truck 2	1326	1345	1450	124	105	91.45	92.76
Truck 3	1395	1354	1478	83	124	94.38	91.61

Listed below are some of the challenges observed with the first two prototypes:

1. Pig crowding occurs for several reasons, such as pigs getting backed up when more than one person is standing near the vaccination pit, or pigs don't move, and so on. Crowding is seen to be the biggest factor affecting accuracy because the model fails to identify many pigs in a given frame due to visual occlusion of pigs.
2. Loss of ID or ID switching of the bounding boxes that causes miscounts. An ID is assigned to each new pig and is tracked as the pig moves. The ID is lost due to failed detection or ID switch when occlusions occur. As a result, the object will be in an incorrect state causing miscounts.

3. Multiple object detections where a single pig is identified as two pigs occurs. This contributes to ID switching, where a counted object is considered either uncounted or vice-versa, leading to miscounts. For example, the head and butt of the same pig is counted as two pigs if the confidence threshold is too low.
4. Smallest of the weaning pigs are not getting detected at times. Augmenting training data with smaller pigs will help.
5. Faster pigs often carry their ears flat towards the back of their heads and because the ears are an essential part of the object identification process, this resulted in missing pigs. Augmenting training data with faster pigs will help.
6. False positives and false negatives occur from the object detection model. Having false detections or no detections will affect the counts as the application is entirely dependent on the object detection model to identify pig heads. Given the required accuracy is high, margin for error is very low.

Prototype 3

Leveraging the results from the previous two prototypes, the latest iteration was comprised of the following main parts:

1. An iPhone, which streams video of pigs to an IP address on the local network using the RTSP.
2. A custom end-user application running on an Android device, which allows operators to view the live video and live machine learning counts coming from the edge gateway device (Jetson Xavier).
3. A Flask application installed on the Jetson Xavier which exposes an API used by the Android app to receive counts from the machine learning inference.
4. The machine learning models and code installed on the Xavier, which perform the detection, tracking, and counting of pigs in the video stream generated by the iPhone.

In this iteration, video from the previous work was processed into images to produce 2,200 training samples.

Videos taken during the operation of Prototype-2 were parsed into static images. These images were manually analyzed to determine which were likely to cause problematic inference results. These images were labeled accordingly, and converted into a TensorFlow format, tfrecords.

The input data set was a consolidation of the labeled images and data from the previous prototypes. Additional image augmentation was performed in order to improve the accuracy of the model by exposing it to additional variations. Learning rate and optimizers were analyzed to improve model performance. Learning rate controls how quickly or slowly a neural network model learns a problem. The learning rate was modified multiple times in order to test the model training.

Optimizers update the weight parameters to minimize the loss function, which provides feedback the optimizer on whether it is moving in the correct direction to reach a global minimum. The optimizer was switched to Adam from RmsProp. While RmsProp is a simple nonlinear activation

function, Adam is an adaptive activation function which modifies the learning rate by decaying it using several factors as the training progresses.

Newly generated models were compiled to run with hardware acceleration via the TensorRT engine on the NVIDIA Xavier device. This implementation allows the model inference to complete faster, and keep pace with the frame rate of the live video.

Prototype 3 results

Overall, 17,000 pigs were counted during the last onsite validation process. Each group of pigs was recorded to be recounted following the real-time count to calculate the ground truth. The average of accuracy of the pig counter was 79%, with a range of percent of pigs counted from < 50 to > 100%. Figure 4 describes the frequency of percent of pigs counted on nearly 300 groups of pigs counted is depicted below. Values can be greater than 100% when the application counted more pigs than those that actually passed underneath the camera (ground truth).

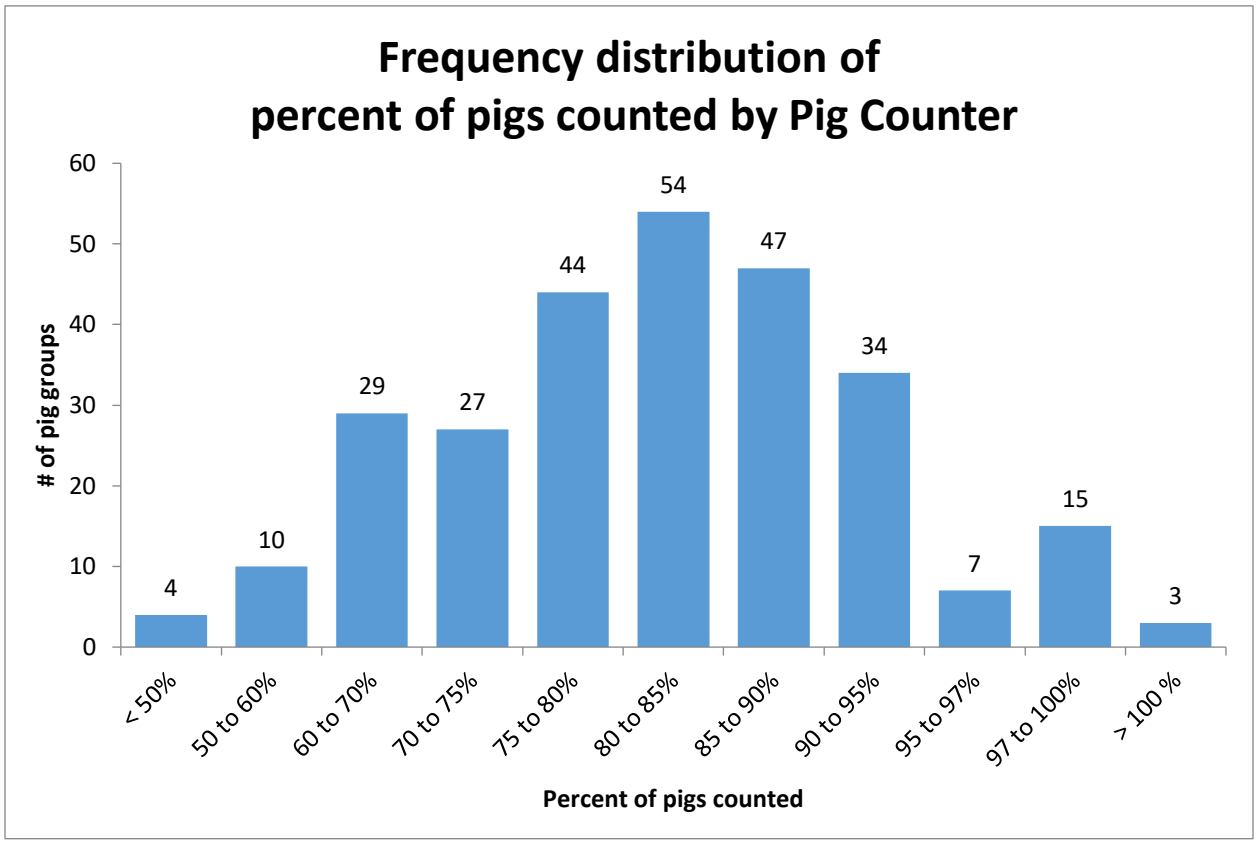

Figure 4. Frequency distribution of the percent of pigs counted of prototype 3 of the pig counting device.

Remaining challenges

A considerable amount of learning was achieved during this project. With the approach taken, funds available did not allow full delivery of a model that met all objectives of the project. Accuracy remains the biggest challenge. Oftentimes, human intervention of pig movement could cause crowding and occlusion, but still even if pigs are or are not crowded, acceptable accuracy was not achieved. Thus, even though the solution may sometimes achieve accuracy that is near 100%, this is certainly not the norm and is therefore not ready for implementation.

Several improvements are recommended to enhance accuracy. While these actions will result in improved model performance, this does not imply the measures will definitively result in adequate accuracy for all pig-counting use cases. In addition, and aside from technology feasibility, it's difficult to estimate the effort needed to attain and maintain the solution to that level of accuracy. That is, the cost-to-benefit rationale for additional investment in the solution may not make it viable. And in fact, cost viability was an underlying constraint for the prototype technology approach which translated primarily in the trade-off for edge compute processing power versus cost (under one to two thousand dollars per solution unit). We wanted to ensure this solution be affordable for producers.

Incremental experiments should be done to iterate towards these answers. Lastly, these improvements may be augmented further by additional techniques discussed in the final section.

1) Dataset Improvements

The most important aspect to improve the model is to increase the number of images used for training. The set of images used for training consists of ~4000 images enriched through augmentation to a full dataset of ~17,000. The deep learning models used will benefit from a much larger dataset and much more diverse image sets covering environmental conditions (color, reflective index, brightness, shadows) and technical aspects (cameras used, resolution). Using a larger variety of image sources will allow the training to avoid overfitting of the model to particular settings. We estimate that gathering additional videos and labeling at least 100,000 diverse images will improve the model regardless of the underlying convolutional neural network (CNN). We estimate that 50 h of video of moving pigs was collected by or provided to AWS and TensorIoT during the development process. Videos were then split into thousands of images for training. The choice to focus on limited video from the weaned pig transfer station may have limited the robustness of the dataset.

2) Object Detection Improvements

The underlying CNN can also be improved. The current network chosen is optimized for edge inference performance. By the end of the project, real-time requirements were met, and the solution is now at a stage where more complex deep learning networks could be evaluated and optimized. Initial results for SSD ResNet50 FPN or YOLOv3 Darknet53 look promising. Using R-CNNs (like Faster-RCNN) would potentially be an option, but require more detailed labeling of data. Therefore, more effort and benchmarks are needed to show if they can achieve the real-time requirement with the hardware limitations in place.

3) Object Tracking Improvements

Other methodologies for the object tracking inference could be investigated to evaluate improvement when tuned for direction tracking. Some areas of interest are:

1. Adjusting the distance function between the centroids to track the movement of the bounding box across two consecutive frames.
2. Retain unmatched objects in memory for some time in case of missed detections.
3. Tweak the confidence scores of object detections to validate the impact on tracking two close detections.
4. Predict position by computing the velocity vector to rematch the missing object detections.
5. Use a tracker and detector in parallel to identify missing detections.
6. Use a stereo vision camera which gives information of depth as well.
7. Use heat maps along with CNN-based detections to accurately identify bounding boxes.

4) Inference Pipeline Improvements

Performance improvements may be realized by multithreading the inference pipeline, and separating the video frame buffer processing from inference. An example would be to validate an implementation around NVIDIA's DeepStream SDK, which also includes and optimized optical flow tracking algorithm. Building on top of that framework would free up more pipeline time for actual inference, enabling the use of larger and more complex models.

5) Input Selection and Sensor Fusion

Lastly, incorporating additional data inputs from a variety of alternative sensors could be joined with the existing computer vision model to enhance accuracy.

Some examples include:

1. Use a stereo vision camera to add an image depth data channel.
2. Evaluate the use of LIDAR / point-cloud based networks like PointPillars or PointRCNN.
3. Utilize a combination of infrared cameras, sensors, filtering and lighting, to both reduce the impact of variable environmental conditions, and to provided augmented data. Infrared thermography could provide an additional data channel by tracking pig bodies.
4. Investigate the possibility of modifying the existing processing workflow to incorporate other markers, other than pig shape, that could be used for object detection, such as adding uniform paint stick markings.

Other findings

The prototype pig counting solution described in this report was built completely from scratch and at the time of the grant there was no commercial solution available. Given the novelty of this use case and the application of this type of technology to pig production, several lessons were

learned along the way that may be useful to future projects that seek to design a technology to increase efficiency in pig production.

1. Scope: The project scope and resources needed must be carefully evaluated by experts who have understanding to the limitations that exist in both technology development and animal production. Our approach of wanting to build a model to count all ages of pigs was too broad, and we significantly increased progress by focusing on the weaned pig use case. We also may have underestimated the initial budget required to complete a project of this type. Although AWS achieved 97% accuracy or better several times, the "average" number of times did not meet our success criteria. AWS did show continual improvement of accuracy the more training that occurred. AWS is of the opinion that – if more funds were available – AWS can achieve the accuracy goal.

2. Plug-and-play: The expectation that a machine learning solution could be ready for use and implementation with high accuracy outcomes appears to be unrealistic in the case of pig counting. Our findings conclude that a model that performs with satisfactory results on initial testing must be continually evaluated and trained, at least for a period of time. This could allow the users to provide the model feedback (counts are accurate or not). Further, more data collection can increase the robustness of the model. We strongly believe that an affordable, accurate, and real-time pig-counting solution is on the horizon for the swine industry, but that continued development after project launch must be included in order to deliver an acceptable solution.

3. Trade-offs: Oftentimes in machine learning, model accuracy may be at odds with rate of inference. Thus, the objective of a project would be to optimize both accuracy and model speed, which may mean you are giving up efficiency in one for the other.

4. Outlook and technology advancement: Within the course of completing this project, we have observed technology that has rapidly advanced. The initial models used in the prototype already had improved versions within months of development. Oftentimes, we found that connectivity in rural areas was a roadblock, yet we predict that in the near future, connectivity will significantly improve. Additionally, as the models and technology continue to become more sophisticated, more this advanced technology will become more accessible and affordable for industries like animal agriculture. Just like several other pig counting technologies came online during the course of this project, we anticipate that a pig counting technology is very realistic and on the horizon for pork producers. However, they must be evaluated for accuracy and feasibility of implementation. A follow up-project to this report intends to test the available systems.