Machine learning agriculture pdf

Volume 119, July 2020, 104926Acultural supply chainSystematic literature studyView full text Our innovation analysts recently looked at new technologies and upcoming startups working on solutions for the agricultural sector. Since there are a large number of startups that work with a wide variety of solutions, we want to share our insights with you. This time we take a look at 5 promising deep learning startups. Heat Map: 5 Top Deep Learning StartupsFor our top 5 picks, we used a data-driven startup scouting method to identify the most relevant solutions globally. Global Startup Heat Map below highlights 5 interesting examples of 272 relevant solutions. Depending on your specific needs, your best choices may look completely different. Click to enlarge Which startups are developing the other 267 solutions? Beriqo – Agri Information Management SystemAgricultural productivity depends on a lot of environmental variables such as weather, groundwater reserves, and soil health. These cause differences in yield even in nearby fields or, sometimes, in different areas of the same area. Precision agriculture aims to minimise this variability by combining remote sensing with artificial intelligence (AI). The Canadian startup Beriqo is developing an information management system (IMS) for agriculture. The startup uses powerful deep learning algorithms to detect pixel-by-pixel changes in piles of satellite imagery collected weekly. Their analysis allows decision-making in precision agriculture by identifying parameters such as underperforming fields, nutritional deficiencies, pest damage, and missed manure streaks. AbuErdan – Animal follow-upAnimal welfare in the poultry industry focuses on the care of individual chickens on a farm. It alleviates certain ethical problems, improves community health, and increases yields. But round-the-clock manual monitoring of all chicken is an uphill task. Deep learning algorithms that analyze camera feeds to monitor all birds are attractive solutions. AbuErdan is a US-based startup that builds a poultry husbandry system. The system uses deep learning networks and predictive analysis to predict chicken performance from past performance and historical data while allowing for better breeding decisions to improve the quality of poultry. The start-up solution also helps breeders to plan and manage the entire poultry value chain. Klimazone – Climate RecipesVariability between different plants of the same crop causes differences in their growth and yield. Growing them in controlled environments minimizes this variability. However, it is not a viable solution for large areas of land. Vertical agriculture and climate recipes address restrictions on space and variations, respectively. A climate recipe is a set of environmental conditions that cause a desired result in the plants. German startup Klimazone creates big data-based solutions for vertical agriculture. Their deep monitoring monitoring allows growers to create inexpensive climate recipes for controlled growth environments. The solutions synchronize food production with sourcing to minimize waste and accurately monitor plant protection and growth for better yields. Klimazone's solution is easily scalable and makes vertical cultivation affordable. Blueberries – Intelligent Spot Spraying SystemFarmers use herbicides to eliminate weed growth in agriculture. However, their uncontrolled use has a significant environmental impact and contributes to the development of herbicide resistance among weeds. Solutions that replace carpet spraying of herbicides with more accurate use reduce costs and mitigate unwanted effects that come from their overuse. Bilberry is a French startup that develops an intelligent spray system for effective weed control. The startup uses cameras mounted on syringes and deep learning-based recognition algorithms for precise application of herbicides in real time. It also maps weed distribution to improve its accuracy with use and for better management of agricultural processes. AgroScout – Image-based Anomaly DetectionPlant diseases such as blight and gall destroy a lot of crops annually. Symptoms of these diseases are often first noticed in leaves as spots or lesions. But the wide variety of blades makes general detection algorithms impractical. Deep learning algorithms overcome this challenge and detect plant diseases by scanning leaves. Israeli startup AgroScout is working on an image-based anomaly detection solution to detect crop diseases from visual inspection of leaves. Their platform is cloud-based and collects images from both drones and smartphones. It also contains data from weather, satellite and other local sensors. The algorithms detect diseases at an early stage, thereby reducing the cost of using pesticides. What about the other 267 solutions? While we believe that data is the key to creating insights, it can be easy to be overwhelmed by it. Our ambition is to create a comprehensive overview and provide actionable innovation intelligence for your Proof of Concept (PoC), partnership or investment objectives. The 5 deep learning startups showcased above are promising examples of the 272 we analyzed for this article. To identify the most relevant solutions based on your specific criteria and collaboration strategy, get in touch. Those who work with livestock also experience time and cost savings as a result of ML-driven technology. ML technology currently improves agriculture and ranching activities in a variety of areas, including livestock health maintenance, dairy and egg production, herding, and selective breeding. Livestock Health Maintenance ML helps farmers maintain happy and healthy cattle herds. In one case of use, a company in Amsterdam uses sensors to monitor cow behavior. ML analysis of the collected predicts fertility patterns, diagnoses eating disorders and warns farmers of signs of heat stress.4 In another example, the health of a milk cow can be evaluated by applying learning algorithms to images of white blood cells extracted from the cow's blood or milk. This process provides indications of certain health issues earlier than simple observation provides. Farmers can then initiate appropriate wellness measures before antibiotics become necessary.4Dairy and Egg Production ML-driven data analysis also helps ranchers optimize operations to more accurately and efficiently manage the production of milk, eggs, and other foods.2 For example, the same technology that collects and analyzes data on dairy activities also allows farmers to make operational decisions that improve milk production by up to 30%.5Animal Herding Autonomous robots are not just monitoring fields of crops for weeds and sick Plants. They are also trained to be cattle and sheep. In addition to physically shepherding animals toward a desired destination, these ML-powered devices can pull heavy objects from one place to another and collaborate with drones to convey important information to farmers.6Selective Breeding Selective breeding involves the use of genetic data to optimize livestock pregnancy rotations and encourage perpetuation of beneficial properties, such as milk quality, disease resistance, fertility, and more. Selective breeding is not new. Ranchers have relied on observable factors to produce livestock lines that have desirable properties for centuries. Today, using ML-supported sensors and applications, a large amount of data on genetic molecular markers, environment, feed makeup, birth patterns, and more can be analyzed, and ranchers can make livestock-mating decisions with far more accurate results. Author: Chris Padwick, Director of Computer Vision and Machine Learning at Blue River TechnologyHow did agriculture affect your day today? If you live in a city, you may feel disconnected from the farms and fields that produce your food. Agriculture is a core piece of our lives, but we often take it for granted. A prototype from 2017 by See & Spray, Blue River Technology's precision tensioning machine Farmers, today faces a huge challenge – feeding a growing global population with less accessible land. The world's population is expected to grow to nearly 10 billion by 2050, increasing global food demand by 50%. As this demand for food grows, land, water and other resources will come under even more pressure. The variation that is natural in agriculture, such as changing weather conditions, and threats like weeds and pests also have knock-on effects on a farmer's ability to produce food. The only way to produce more food while using less resources is through smart machines that can help farmers with difficult jobs, offering more consistency, precision and efficiency. Alex Marsh, one of our Field Operations Specialists, pictured with a self-propelled syringe. We work with large machines on the Blue River — Alex is 6'4 tall and is roughly on par with the top of the deck. Agricultural RoboticsAt Blue River build the next generation of smart machines. Farmers use our tools to control weeds and reduce costs in a way that promotes sustainability in agriculture. Our weeding robot integrates cameras, computer vision, machine learning and robotics to make an intelligent syringe running through fields (using AutoTrac to minimize the load on the driver) and quickly targets and sprays weeds, leaving the crops intact. The machine must make real-time decisions about what is a crop and what is a weed. As the machine drives through the field, high-resolution cameras collect high-frame images. We developed a convolutional neural network (CNN) with PyTorch to analyze each frame and produce a pixel-accurate map of where crops and weeds are. Once the plants are all identified, each weed and crop is mapped to field sites, and the robot only squirts weeds. This whole process takes place in milliseconds, allowing the farmer to cover as much land as possible because efficiency matters. Here is a great See & Spray Video that explains the process in more detail. To support Machine Learning (ML) and robotics, we stack built an impressive computing unit, based on the NVIDIA Jetson AGX Xavier Edge AI platform. Since all of our inference conference happens in real time, uploading to the cloud would take too long, so we take server farms to the field. The total computational power on board the robot that has just engaged in visual inference and spray robotics is on par with IBM's super computer, Blue Gene (2007). This makes this a machine with some of the highest computational capacity of any moving machine machine in the world! Building weed detection modelsMy team of scientists and engineers is responsible for training the neural network model that identifies crops and weeds. This is a challenging problem because many weeds look just like crops. Professional agronomists and weed researchers train our marking workforce to label the images correctly — can you spot the weeds below? You're looking at cotton plants and some weeds. Can you tell the difference? In the photo below, the cotton plants are in green and the weeds are in red. The cotton plants are in green and the weeds are in red. Machine learning stackOn the machine learning front, we have a sophisticated stack. We use PyTorch for training all our models. We have built a set of internal libraries on top of PyTorch that allows us to perform repeatable machine learning experiments. Responsibilities my team falls into three categories: Build production models to deploy on the robotsPerforma machine learning experiments and research with the goal of constantly improving model performanceData analysis / computer science related to machine learning, A/B testing, process improvement, software engineeringWe chose PyTorch because it is very flexible and easy to troubleshoot. New team members can quickly pick up speed, and the documentation is thorough. Before we started with our team used Caffe and Tensorflow extensively. Diligently. In 2019, we made a decision to switch to PyTorch and the transition was seamless. The framework allows us to support production model workflows and research workflows at the same time. For example, we use the torchvision library for image conversions and tensor conversions. It contains some basic functionality and it also integrates really nicely with sophisticated augmentation packages like imgaug. Transforms the object into torchvision is a piece of cake to integrate with imgaug. Below is a code example with the Fashion MNIST dataset. A class called CustomAugmentor initiates iaa. The sequential object in the constructor then calls augment_image() in __call__ method. CustomAugmentor() is then added to the call to transforms.Compose(), before ToTensor(). Now the train and choice data loader will apply the augmentations defined in CustomAugmentor() when the kits are loaded for training and validation. In addition, PyTorch has emerged as a favorite tool in the computer vision ecosystem (look at Papers With Code, PyTorch is a regular submission). This makes it easy for us to try new technologies like Debiased Contrastive Learning for semi-supervised training. On the model education front, we have two normal workflows: production and research. For research applications, our team runs PyTorch on an internal, on-prem compute cluster. Jobs running on the local cluster are managed by Slurm, which is an HPC batch job based scheduler. It's free, easy to set up and maintain, and provides all the functionality our group needs to run thousands of machine learning jobs. For our production based workflows we leverage an Argo workflow on top of a Kubernetes (K8s) cluster hosted in AWS. Our PyTorch training code is deployed to the cloud using Docker.Deploying models on field robotsFor production deployment, one of our top priorities is high speed interference on the edge computing device. If the robot needs to run slower to wait for conclusions, it may not be as effective in the fields. For this purpose, we use TensorRT to convert the network to an NVIDIA Jetson AGX Xavier optimized model. TensorRT does not accept JIT models as input, so we use ONNX to convert from JIT to ONNX format, and from there we use TensorRT to convert to a TensorRT engine file that we deploy directly to the device. As the tool saccharine develops, we expect this process to improve as well. Our models are distributed to Artifactory using a Jenkins construction process and they are distributed to remote machines in the field by pulling from Artifactory.To monitor and evaluating our machine learning runs, we have found the Weights & Biases platform to be the best solution. Their API makes it quick to integrate W&; B logging in an existing codebase. We use W&B to monitor ongoing training, including live curves of training and SGD vs Adam ProjectAs an example of using PyTorch and W&; B, I will run an experiment and compare the results of using different different in PyTorch. There are a number of different solvers in PyTorch — the obvious question is which one should you choose? A popular choice of looser is Adam. It often produces good results without having to set any parameters and is our usual choice for our models. In PyTorch there is this looser under torch.optim.adam. Another popular choice of solver for machine learning researchers is Stochastic Gradient Descent (SGD). This solver is available in PyTorch as torch.optim.SGD. If you're not sure about the differences between the two, or if you need a refresher, I suggest reviewing this write up. Momentum is an important concept in machine learning, as it can help solver to find better solutions by avoiding getting stuck in local minima in the optimization space. Using SGD and speed the question is this: Can I find a momentum setting for SGD that beats Adam? The experimental setup is as follows. I use the same training data for each run, and evaluate the results on the same test set. I'll compare the F1 score for plants between different runs. I set up a number of runs with SGD as looser and sweeping through momentum values from 0-0.99 (when using momentum, making anything greater than 1.0 that the solver deviates). I set up 10 runs with momentum values ranging from 0 to 0.9 in increments of 0.1. After I performed another set of 10 runs, this time with momentum values between 0.90 and 0.99, with steps of 0.01. After looking at these results, I also ran a set of experiments on momentum values of .999 and .9999. Each run was done with a different random seed, and was given a tag by SGD Sweep in W&; B. The results are shown in Figure 1.Figure 1: On the left side, f1 points for crops are displayed on the x-axis, and the driving name is displayed on the y-axis. On the right side, the F1 score for plants is shown as a function of momentum value. It is very clear from Figure 1 that greater values of momentum increase the f1 score. The best value of 0.9447 is at the momentum value of 0.999, and drops to a value of 0.9394 to a momentum value of 0.9999. The values are shown in the table below. Table 1: Each run is shown as a row in the table above. The last column is the momentum setting for the run. F1 points, precision and recall for Class 2 (crops) are shown. How do these results compare to Adam? To test this I ran 10 identical runs with torch.optim.Adam with only default parameters. I used the Tag Adam runs in W& B to identify these runs. I also tagged each set of SGD runs for comparison. Since a different random seed is used for each run, the solver will initiate different each time and will end up with different weights at the last epoch. This produces slightly different results on the test set for each run. To compare them I will need to measure the spread of values for Adam and SGD running. This is easy to do with a box plot grouped by tag in W B.Figure 2: Dissemination of values for Adam and SGD. The Adam runs are in the left of the graph in green. SGD runs appear as brown (0.999), teal (0-0.99), blue (0.9999), and yellow (0.95). The results are shown in chart form in Figure 2, and in Table 2. The full report is available online as well. You can see that I haven't been able to beat the results for Adam by just adjusting momentum values with SGD. The momentum setting of .999 produces very comparable results, but the variance on the Adam runs is tighter and the average is also higher. So Adam seems to be a good choice of looser for our plant segmentation problem! Table 2: Run table showing f1 points, optimizer and momentum value for each run. PyTorch VisualizationsWith the PyTorch integration, W&B picks up the inclinations of each layer, so we can inspect the network during training. W&B experiment tracking also makes it easy to visualize PyTorch models during training, so you can see the loss curves in real time in a central dashboard. We use these visualizations in our team meetings to discuss the latest results and share updates. As the images pass through our PyTorch model, we seamlessly log predictions to Weights & Biases to visualize the results of model training. Here we can see predictions, ground truth and labels. This makes it easy to identify scenarios where model performance doesn't meet our expectations. The ground truth, the predictions and the difference between the two. Crops appear in green, while weeds appear in red. Here we can quickly browse the ground truth, predictions and the difference between the two. We've noticed the crops in green and grass in red. As you can see, the model does a pretty reasonable job of identifying crops and weeds in the picture. Here is a short code example of how to work with data frames in W&; B: Reproducibility and traceability are important features of any ML system, and it's hard to get right. When comparing different network architectures and hyperparameters, input needs to be the same to make runs comparable. Often individual practitioners on ML teams save YAML or JSON config files — it's excruciating to find a team member running and wade through their config file to find out what training sets and hyperparameters were used. We've all done it, and we all hate it. A new feature like W&; B just released solves this problem. Artifacts allow us to track the inputs and exits of our training and evaluation runs. This helps us a lot with reproducibility and traceability. By inspecting the Artifacts section of a run in W&B, I can tell you which datasets were used to train the model, which models were produced (from multiple runs), and the results of the model evaluation. A typical use case is as follows. A data staging process retrieves the latest and greatest data and stages it to disk for training and testing (separate datasets for each). These datasets are set as A training run takes the exercise set artifact as input and exits a trained model model an output artifact. The evaluation process takes the artifact test set as input, along with the trained model artifact, and the output an evaluation that can include a set of metrics or images. A targeted acyclic graph (DAG) is formed and visualized within W&B. This is helpful because it is very important to track the artifacts involved in releasing a machine learning model into production. A DAY like this can be formed easily:One of the major advantages of the Artifact feature is that you can choose to upload all artifacts (datasets, models, evaluations) or you can choose to upload only references to the artifacts. This is a nice feature because moving lots of data around is time consuming and slow. With the dataset artifacts, we simply store a reference to these artifacts in W& B. whereas it enables us to maintain control over our data (and avoid long transfer times) and still gain traceability and reproducibility in machine learning; Leading ML teams looking back on the years I've spent leading teams of machine learning engineers, I've seen some common challenges:Efficiency: As we develop new models, we need to experiment quickly and share results. PyTorch makes it easy for us to add new features quickly, and Weights & Prejudices gives us the visibility we need to troubleshoot and improve our models. Flexibility: Working with our customers in the fields, every day can bring a new challenge. Our team needs tools to keep pace with our ever-changing needs, which is why we chose PyTorch for its thriving ecosystem and W&B for the easy, modular integrations. Performance: At the end of the day we need to build the most accurate and fastest models for our field machines. PyTorch enables us to iterate quickly, then production customize our models and distribute them in the field. We have full transparency and transparency in the w&b development process, making it easy to identify the most performant models. I hope you've had this short tour of how my team uses PyTorch and weights and biases to enable the next generation of intelligent agricultural machines! About the authorI'm director of computer vision and machine learning at Blue River Technology. We build robots that separate crop from weed in an agricultural field and then just spray weeds. I've been working on Blue River for four and a half years. My background is in Physics and Astronomy and in the grad school I helped build a telescope to measure the cosmic background radiation. Check out our career page, we're hiring! Hire!