Computer Vision News

The magazine of the algorithm community

A publication by RSIP VISION

January 2019

Research Review:
Rethinking ImageNet Pre-training
by Kaiming He, Ross Girshick, Piotr Dollár

Upcoming Events

Challenge:
ABCD Neurocognitive Prediction

Women in Computer Vision

Artificial Intelligence Spotlight News

Project Management:
Annotating Data for Medical Projects

Tip of the Month:
Train Your Network (with codes!)

Computer Vision Project:
Grading and Sorting Software with Deep Learning

Contents

Did you subscribe to Computer Vision News? It's free, click here!

Computer Vision News

Editor:
Ralph Anzarouth

Engineering Editor:
Assaf Spanier

Publisher:
RSIP Vision

Contact us

Give us feedback

Free subscription

Read previous magazines

Follow us:

Dear reader,

Welcome to the first issue of **Computer Vision News** in **2019**! The main motivation behind the publication of this magazine for the 4th year in a row is that readers appreciate what we do: Computer Vision News and its Daily conference offsprings (CVPR Daily, MICCAI Daily, ECCV Daily and ISBI Daily) have generated an aggregate **1,100,000 page views** from all our readers together during 2018. Thank you! This magazine will continue to be 100% free, so please subscribe (if you haven't done it yet) and subscribe your friends and colleagues.

A word about RSIP Vision: as you might know, **we are a pioneering software house offering custom algorithms and R&D** to the industry. For us, 2018 has been a year of spectacular development in both business and technology. Enterprises love to work with us, as witnessed by the feedback we have just received from **Biosense Webster**, part of the **Johnson & Johnson** family of companies. You can read it on page 21!

Enjoy reading this new issue of Computer Vision News and, as always, **take us along for your next Deep Learning project!**

Ralph Anzarouth
Editor, **Computer Vision News**
RSIP Vision

Do we really need ImageNet pre-training? Read on page 4 our exclusive review of the new outstanding work by Kaiming He, Ross Girshick and Piotr Dollár of Facebook AI Research!

by Assaf Spanier

Every month, Computer Vision News reviews a research paper from our field. This month we have chosen **Rethinking ImageNet Pre-training**. We are indebted to the authors (**Kaiming He**, **Ross Girshick** and **Piotr Dollár**), for allowing us once again to use their images to illustrate our review. Their article is here.

Do we really need ImageNet pre-training?

This remarkable paper demonstrates competitive results on object detection and instance segmentation on the COCO dataset using standard models trained from random initialization. The results are on par with ImageNet pre-training. Training from random initialization is surprisingly robust; the results hold: 1) even when using only 10% of the training data, 2) for deeper and wider models, and 3) for multiple tasks and metrics. Experiments show that ImageNet pre-training speeds up convergence early in training, but does not necessarily provide regularization or improve final target task accuracy.

Introduction:

Starting with the RCNN articles, the early breakthroughs in using deep learning for object detection were achieved with networks pre-trained for image classification on ImageNet and then fine-tuned on the intended dataset for object detection. Following these results, most modern object detection networks and many other computer vision algorithms use the pre-training then fine-tuning paradigm. Some of the latest articles published push this paradigm even further, by pre-training on datasets 6 to 3,000 times the size of ImageNet (JTF 6×, ImageNet-5k 300×, and Instagram 3000×). The paradigm, while showing significant improvement for image classification training, only provides a little improvement when training for object detection tasks (up to about 1.5%). This improvement dwindles the larger the object detection task dataset is relative to the pre-training dataset.

Method and Innovation:

In this paper, the authors show:

1. Though ImageNet pre-training speeds up convergence, training from scratch achieves the same accuracy given sufficient time. Note, that in training from scratch the network must learn low-level and mid-level features (such as

edge, textures, etc.), which it usually learns in pre-training.

2. When pre-training paradigm results are reported as being more efficient, the pre-training time isn't always taken into account.
3. The authors show that competitive results can be achieved training on just 10% of the COCO dataset from random initialization, if hyperparameters are carefully selected to prevent overfitting. Using the same hyperparameter settings as pre-trained networks, random-initialization training will achieve the same results, even when trained on only 10% of the dataset.
4. ImageNet pre-training shows no benefit when the target tasks/metrics are more sensitive to spatially localized predictions.

In the context of the current state of the art, these results are surprising and should challenge the ImageNet pre-training paradigm's influence. ImageNet pre-training is and in the near future will continue to be the go-to solution, especially: 1) where developers have insufficient data or computational resources to train on their target task from scratch, and 2) since ImageNet pre-training is widely seen as a 'free' resource, thanks to the labeling and annotation efforts that have been done before, and the approachability and wide availability of ImageNet pre-trained models.

However, the authors' observations suggest that looking forward, when developers have sufficient data and as computational resources improve, training from scratch / from random initialization should be seriously considered. The paper demonstrates that collecting data and training directly on the target task is a solution that needs to be considered, especially in such cases where there is a significant disparity between the pre-training task and the target task. The new evidence provided and explored by the paper points toward a need for the community to discuss and reevaluate the pre-training -- fine-tuning paradigm.

Implementation:

Let's look at the architectures, training rate, optimization and normalization methods and hyperparameter settings used in this work.

Architecture

Mask R-CNN with ResNet, ResNeXt or ResNeXt plus Feature Pyramid Network (FPN) backbones were investigated.

Normalization

The normalization methods commonly used in training the standard pre-trained networks are less suitable for detection and segmentation training, since these normalization methods require loading very large volumes of data for training:

very high resolution images, with labeling for every pixel. This would result in only being able to process a very small number of images in each batch, making normalization excessively difficult. This difficulty is avoided by the fine-tuning paradigm networks, which take advantage of the normalization parameters learned during pre-training.

The authors adopted the following normalization methods for training from random initialization on detection and segmentation tasks.

1. Group Normalization (GN) -- GN performs computation that is independent of the batch dimensions. GN's accuracy is insensitive to batch sizes.
2. Synchronized Batch Normalization (SyncBN) -- an implementation of BN which increases the effective batch size for BN by using many GPUs, overcoming the problem of small batches.

Learning rate

The learning rate update policy was to lower the learning rate by 10x for the last 60k iterations. And by another 10x for the last 20k iterations. The authors showed that there is no need to lower the learning rate earlier than just before the very end of training. There is also no need to train at a low learning rate for a long time -- this only causes overfitting.

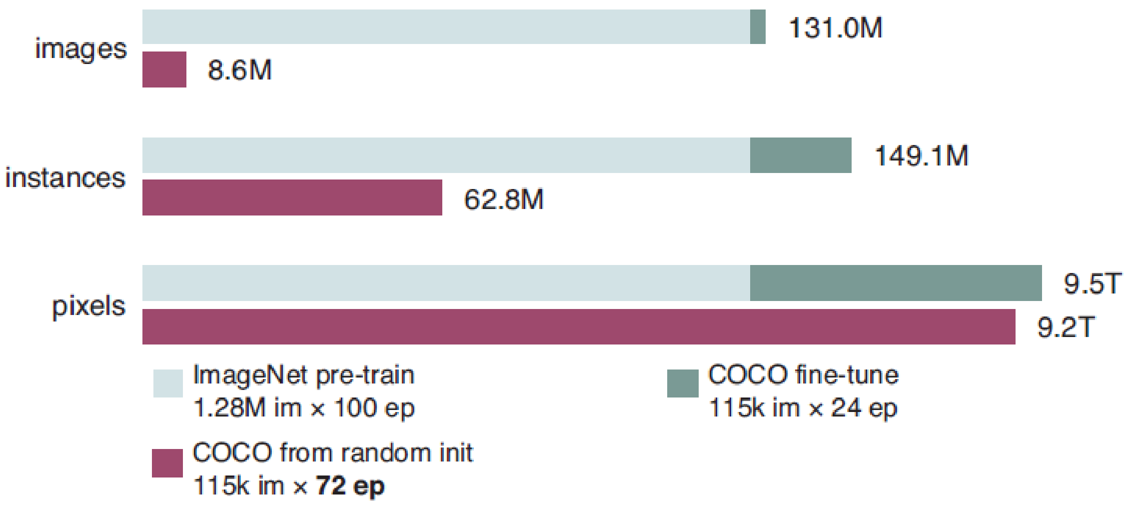
Hyper-parameters

All other hyper-parameters follow those in Detectron. Specifically, the initial learning rate is 0.02 (with a linear warm-up). The weight decay is 0.0001 and momentum is 0.9. All models are trained on 8 GPUs using synchronized SGD, with a mini-batch size of 2 images per GPU. Per Detectron's default, Mask R-CNN used no data augmentation for testing and only horizontal flipping augmentation for training. The image scale was 800 pixels for the shorter side.

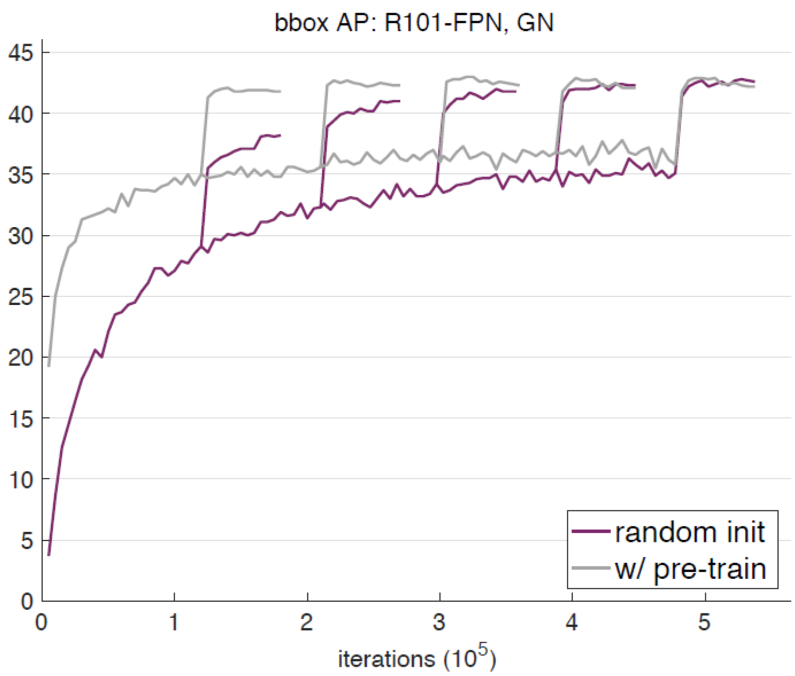
Results:

Given enough data, any network can be trained :-) as it can be seen in the graph that follows. The volume of data used for ImageNet pre-training is shown in light blue; The fine-tuning volume of data used is darker blue; and training from scratch volume of data used is in purple. The top bar is the number of images trained used for training; The middle bar is the number of objects (each image can include more than one object); The bottom purple bar shows you the total volume of pixels handled (image sizes vary between datasets), which translate to volume of data. You can see from the bottom purple bar that overall the network processes the same data volume whether pre-trained then fine-tuned or trained from scratch (random initialization).

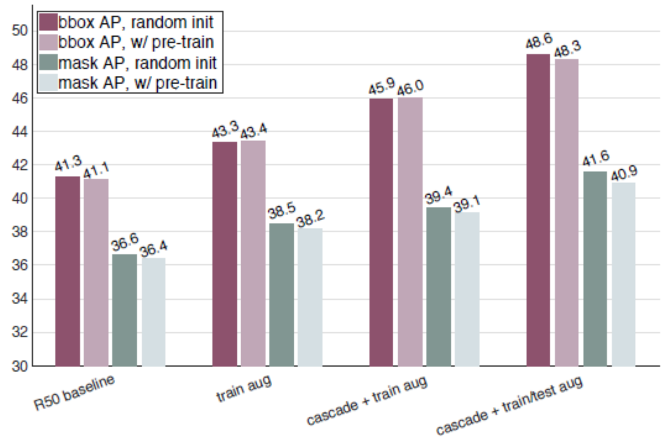
Given enough data, any network can be trained

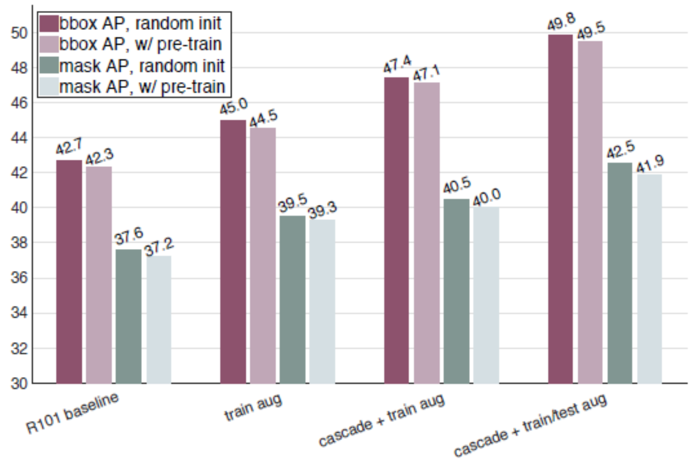


The validation bbox AP curves are shown side by side below: using GN with ResNet-101 (R101) backbone and using SyncBN with ResNet-50 (R50) backbone. Each figure compares the curves for models trained from random initialization vs. ImageNet pre-trained then fine-tuned.



The figure in the next page presents comparisons between training from random initialization vs. pre-training then fine-tuning on various systems using Mask R-CNN, including: 1) baselines using FPN and GN, 2) baselines with training time multi-scale augmentation, 3) baselines with Cascade RCNN and training-time augmentation and 4) plus test-time multi-scale augmentation - left: R50; right: R101.

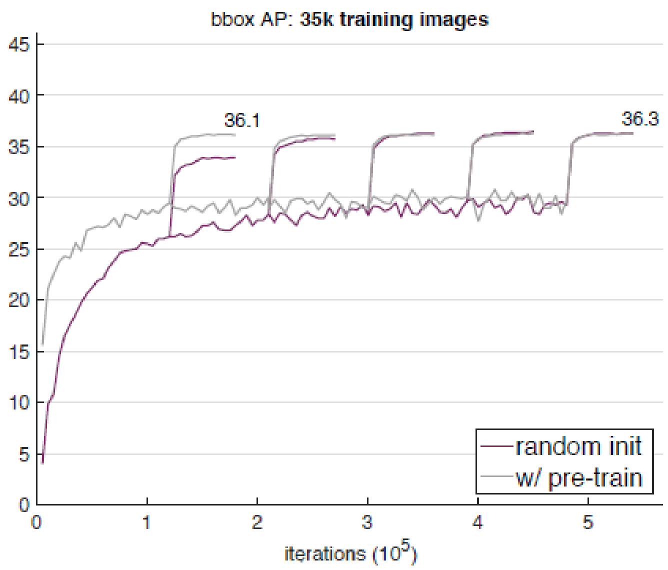




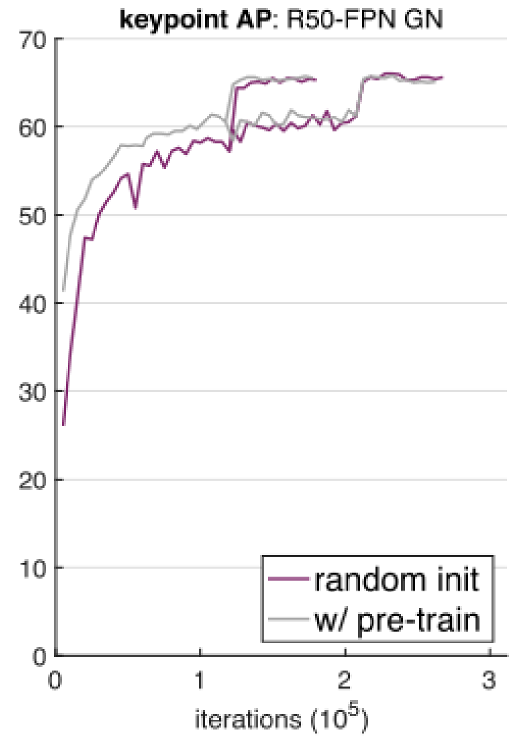
Repeat training using different methods, different configurations and various architectures, comparing the performance of networks trained from random initialization with those pre-trained then fine-tuned -- shows it cannot be just chance that the overall data needed proves equal again and again, whether starting from scratch or using pre-trained networks. The methods are equivalent.

schedule		2×	3×	4×	5×	6×
R50	random init	36.8	39.5	40.6	40.7	41.3
	w/ pre-train	40.3	40.8	40.9	40.9	41.1
R101	random init	38.2	41.0	41.8	42.2	42.7
	w/ pre-train	41.8	42.3	42.3	41.9	42.2

Another experiment ran by the authors consisted in training the pre-trained network to find the optimal hyperparameter settings. The authors used the hyperparameter settings discovered by training the pre-trained network to train their random-initialization network from scratch -- and achieved equal results using just one third of the data. The figure below presents training accuracy (purple shows random initialization, grey shows pre-trained then fine-tuned).



Mask R-CNN training for the COCO human keypoint detection task: for this task, the random initialization network can learn more quickly than the pre-trained then fine-tuned network, even without additional training time. Keypoint detection is a task more sensitive to spatially localized predictions. This is evidence that ImageNet pre-training provides only weak training for spatially localized predictions and that, for these tasks, training from scratch may be perfectly equivalent.



Pre-trained networks converge much faster…

Summary of the paper's insights

- Training networks on their target tasks from scratch (random initialization) is viable, even with no changes to architecture.
- Training from scratch requires more iterations for convergence. Pre-trained networks converge much faster.
- In many different configurations and circumstances, training from scratch may achieve performance on par with that of pre-trained then fine-tuned networks. Including even training on COCO on just 10k images.
- ImageNet pre-training doesn't necessarily help reduce overfitting, except in the case of very small datasets.
- ImageNet pre-training is less useful if the target task is more related to object localization than classification.

Annotating Data for Medical Projects

RSIP Vision's CEO Ron Soferman has launched a series of lectures to provide a robust yet simple overview of how to ensure that computer vision projects respect goals, budget and deadlines. This month **Arik Rond** tells us about **Annotating Data for Medical Projects**. It's another tip by **RSIP Vision** for Project Management in Computer Vision.

Last month's lecture dealt with one of the main issues that a project manager must solve: **collecting and selecting data for medical projects**. This month we discuss about **how this data should be annotated**.

This too is a critical phase in deep learning based projects. The easier case is when the client requesting the work has available annotated data for the project. Another optimal case is when open datasets are available to use. When it happens, there is no need for further data annotation.

Still, this is not the common case, since **labeling medical data is at least as problematic as finding it**. Moreover, we have seen that training data should be as diverse as possible, to cover as many cases as possible in the real world. This adds further complexity to the annotation task. Greater data diversity generally entails the need for better trained annotators, with advanced medical expertise.

When the company has its own in-house doctor, the solution of this problem is much easier: the doctor can provide these annotations or at least supervise them and improve their accuracy. When limited medical eyes are available, the company needs to think of creative ways to secure accurate labels for its medical images.

> ***"...computers teaching computers..."***

One of the possible solutions is the use of **conventional (not deep learning) semi-automated computer vision techniques** to generate these labels. This is what Ron Soferman calls "**computers teaching computers**".

Indeed, it may happen that a previous version of the software, though slower and less precise, will be able to run offline and provide **sufficiently accurate annotations**. After some necessary corrections, these initial results will turn into accurate and reliable training data.

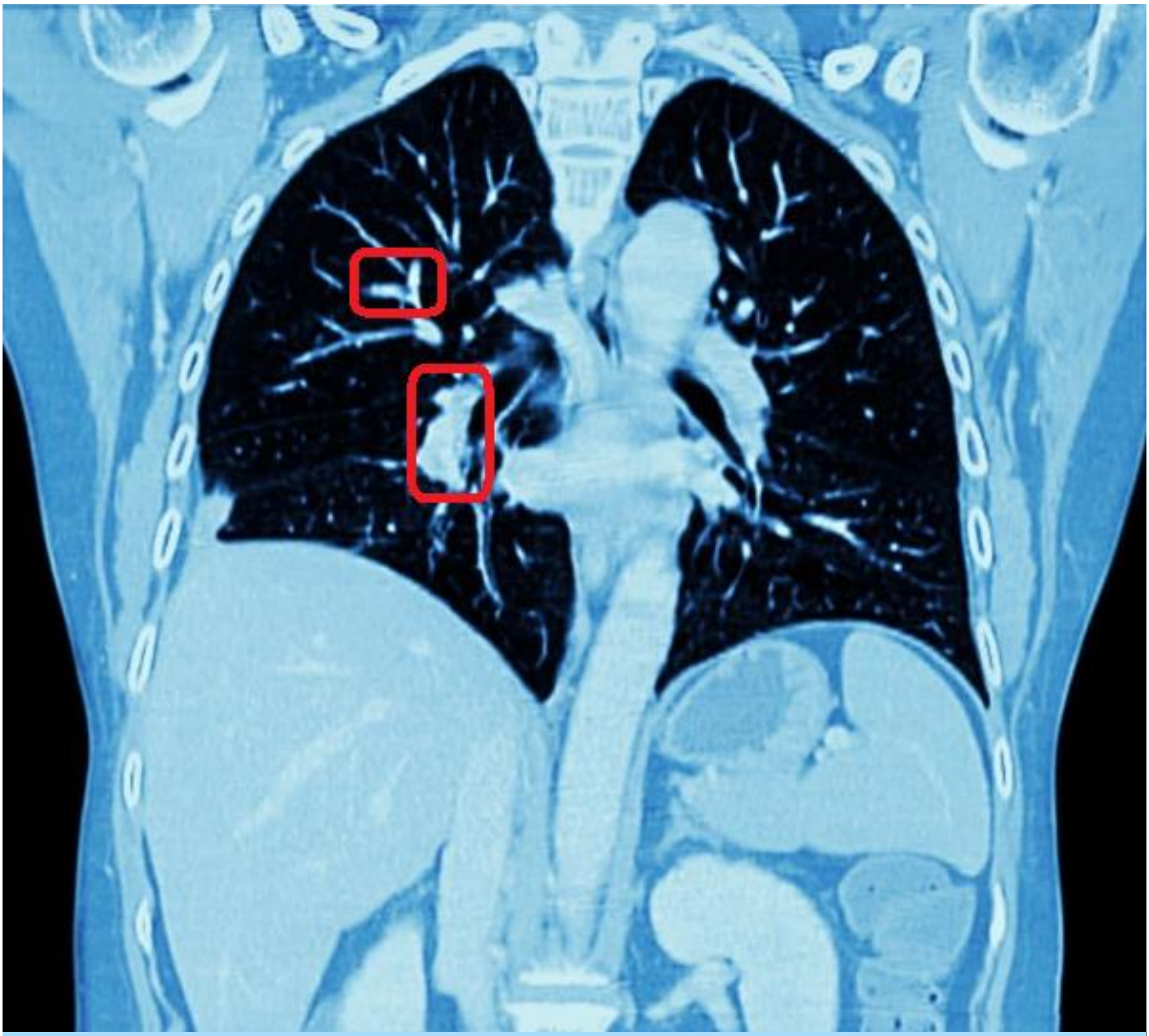
When even a previous version of the software is not available, it is still possible to write a simple program that supports the annotation task: for example, a simple threshold of the image sometimes suffices to provide a partial segmentation. This method will generate much noise, but at the same time it may simplify the task of the human annotator. In same cases, this hint can save up to 80% of the labeling effort.

...data diversity generally entails the need for better trained annotators, with advanced medical expertise.

Other times, a more complex program is needed, though not excessively complicate either.

In addition, at a certain point in a project we may have a neural network that is not accurate enough for the final task, but is accurate enough to **create initial annotations**. Once this part is done, it opens the possibility of letting non-medical experts perform the annotation, under the supervision of a doctor to oversee and eventually correct any mistake.

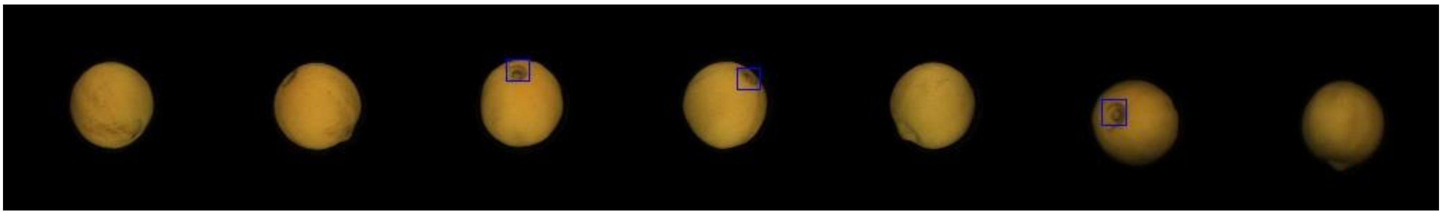
More project management lectures



Visible lung cancer on CT scan of chest and abdomen.
Data annotation too is a critical phase in deep learning based projects.
When limited medical eyes are available, the company needs to think of creative ways to secure accurate labels for its medical images.

Fruit Grading and Sorting with Deep Learning

Every month, **Computer Vision News** reviews a successful project. Our main purpose is to show how diverse image processing techniques contribute to solving technical challenges and real world constraints. This month we review a precision agriculture project by **RSIP Vision's**: **Fruit Grading and Sorting with Deep Learning**. This research is the result of a cooperation between **RSIP Vision** and **Sunkist Growers - Research & Technical Services (RTS)**, division of a leading not-for-profit marketing cooperative entirely owned by and operated for the California and Arizona citrus growers.



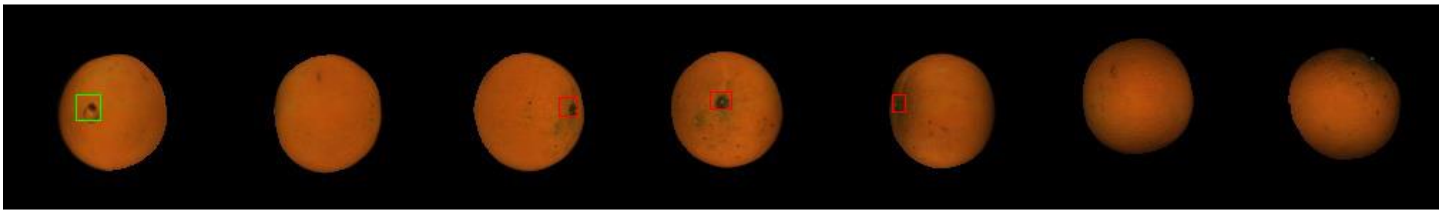
One of the most interesting challenges in handling the supply chain of fruits and vegetables lays in the ability to **sort and grade produce in the optimal way**, given the whole range of possible product features and the requirements coming from the distribution channels to the packaging house. **RSIP Vision** has been involved in several projects within this application field.

During the past three years, we have boosted precision agriculture work to drive it into the new Artificial Intelligence age. This specific project was conducted in partnership with **Sunkist Growers RTS**, division of a growers cooperative dedicated to run traditional growing practices while pioneering innovative solutions.

Together, we launched **a new approach** to AI that enables to control result and at the same time solve any issue that might arise while performing this task. **Sunkist RTS** had developed a large sorting machine, with great field results. But they wanted to perfect them, by developing the machine of the next generation. Achieving the best possible grading results necessarily entails integrating new technologies like **Deep Learning**.

A major requirement set by **Sunkist RTS** for such an AI system was that the Deep Learning networks would provide specific reports about specific features and events, rather that one unique result for the whole produce, which might be like a black box hiding the detailed results.

Other challenges needed a solution, like **time constraints**: in agriculture, due to its nature of massive throughput needed by the clients, there is very little time to handle each produce. The fruit must be processed and graded **in real time** as they flow through the sorting/grading machine - **many fruits per second**.



... very improved results with respect to the traditional method.

Also, **the packaging house** needs to have control over the final grade of the produce as well as each sub-grade, and these controls need to be modified often without re-training or even stopping the system, which would perturbate the operations at the site.

One example of issue that **we detected and solved**: differentiating the stem and blossom areas of the fruit (which do not downgrade the fruit) from scars and blemishes on the fruit surface (which do). In that way, when grading the spots in that area, stem and blossoms are ignored.

Using one step that will give one grading answer to each input image, might be very problematic: it would need a lot of retraining **when the conditions of the field change** and phenomena like plant pests and diseases appear in the data.

Thus, **RSIP Vision** and **Sunkist RTS** developed this stepwise approach based on AI that finds the specific features that define the quality of the fruit and provides a report **for each specific area**.

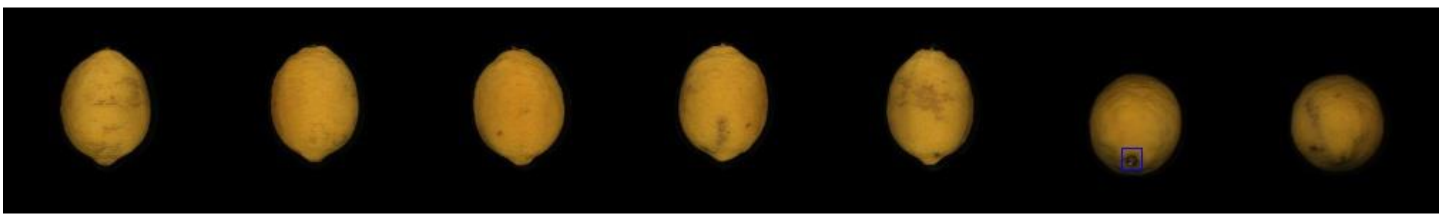
The **final score** is obtained by aggregating each individual score, thus enabling to grade the produce according to market requirements.

The Deep Learning method has to be adapted according to every specific feature and phenomena, so that we are using the optimal network: the one that gives the best accuracy result in the detection task of the relevant features - stains, diseases, stems and blossoms. Once adapted, we train the system accordingly **to obtain the most robust results**, that will not change during the season, even when ripeness of produce is changing.

This approach gave **Sunkist Growers** exceptional results, enabling the successful detection of new phenomena, the control of the result and the analysis of any issue arising during operation.

The great novelty of this system introduced by **Sunkist RTS** and **RSIP Vision** is the ability to utilize **state-of-the-art AI in a robust and controlled setting**. The outcome is very improved results with respect to the traditional method.

Take us along for your next Deep Learning project!

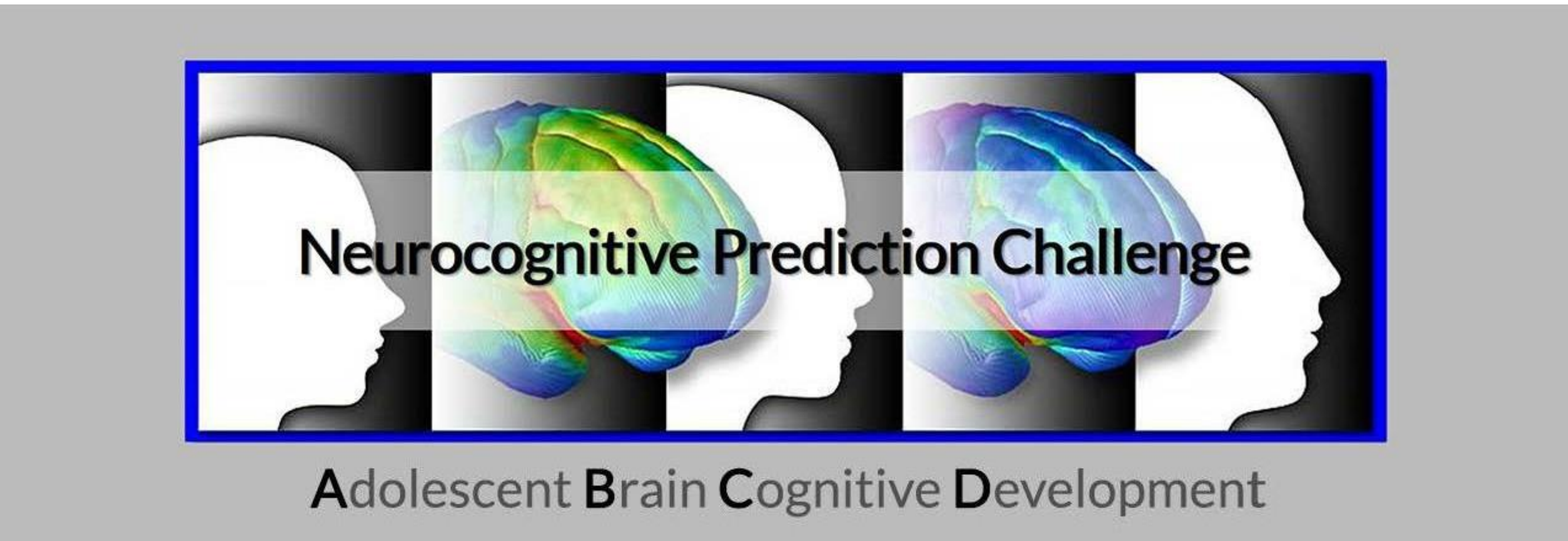


The ABCD Neurocognitive Prediction Challenge (ABCD-NP-Challenge 2019) represents the largest long-term study of brain development and child health in the United States. Chairs Kilian Pohl and Wesley Thompson, with co-chair Susan Tapert, discussed with us about this cutting-edge project which aims to examine how childhood experiences like sports, video games, social media, unhealthy sleep patterns, smoking, and so on affect brain development and social, behavioral, academic, health and other outcomes.

Can you tell us more about the ABCD Neurocognitive Prediction Challenge?

Kilian: ABCD is the largest long-term MRI and neuroscience study in the United States. They recruited over 11,500 participants between the ages of 9 and 10. They do all kinds of neuropsychological testing in addition to acquiring structural, diffusion functional MRI. Wes and I thought we should propose to do something with this data to entice people outside the community, that are not necessarily part of ABCD, to work with this dataset. We came up with an idea. If I give you a lot of MRI scans, how accurately can you predict the fluid intelligence score of those subjects? In this challenge, you get about 4,000 datasets for training. We are in the process of uploading that data now. We process the data. It is skull-stripped. You get grey matter parcellations. You also get residual fluid intelligence scores that Wes computed. Then we give you about 3,500 scans. At the end, it'll be around 6,000 scans. We ask you, based on those scans, to provide us with the fluid intelligence scores of those subjects. Then we will measure the error to what we have, the fluid intelligence score, so we can compute it based on the neuropsychological testing and report the error in between.

Wes: ABCD releases the data publicly on a yearly basis for analysis. The images actually are slightly different. The images that Kilian is working with are actually released almost immediately. There's a delay of a few weeks, maybe. [*Kilian's smiling*] Maybe it's a little bit longer than that. The idea



Kilian Pohl is Program Director at SRI International

then is that we've already released data on 4,500 or so children's neuropsychological assessments including their fluid intelligence scores to the public to analyze. Those, merged with the skull-stripped images, the T1 images, that Kilian has put together constitute the training dataset for the challenge. Those are publicly available to anybody with any application. They don't have to be in our competition. The timing for the next data release is March 15, 2019. That's the current time slotted for the next release of public data including the neuropsych scores. We're soliciting people to submit their prediction algorithms before that date so that they do not have access to the test dataset, which is the training score, the neurocognitive scores, for the people who are currently not in the current version of the baseline data. That's the structure of the challenge.

Kilian: This is kind of outside the format of how, for example, MICCAI proposes challenges normally. The reason is not driven by us. The reason for this difference is basically the release schedule of data by ABCD.

Wes: That's not in our control, of course.

What is the motivation behind the challenge?

Wes: From my perspective, the justification for the contest is twofold, maybe threefold. The first reason why I was eager to join the challenge is because I wanted to promote the usage of the ABCD data outside of the consortium. A large chunk of how we, as a consortium, will be evaluated is how many people outside of our internal consortium members actually end up using the data in a productive way. I felt, and I'm sure Kilian felt the same way, that this would be a very nice way to really spur people to use the data. Motivation #2 is that I'm really curious to see how well we can describe the relationship between the brain and behavioral measures. Personally, I settled on the neurocognition because that's something that clearly has a seat in the brain. There's been a lot of research on that showing a lot of brain morphometry and function and how that's associated with neurocognition in different ways. Susan can speak more to that in terms of a scientific perspective. Also, the fluid intelligence measure is very normally distributed, so it's something that I think there's a lot of evidence for. I think there's a chance that we'll be able to predict it with a meaningful amount of variance explained. The third reason is basically promoting methods development for modern machine learning approaches to the brain prediction problems. I'm just really curious to see how well we can do. what are the methodologies that really work well in this context? Is


Susan Tapert is Professor of Psychiatry at UC San Diego

it deep learning? Susan can talk about the whole scientific rationale looking at the relationship between brain structure and neurocognitive performance.

Susan: I'm very excited to do this challenge so we can find some interesting ways to see how neurocognition relates to brain structure in kids. One thing that I think is very important is that we're very focused at ABCD and looking at how the brain developed over time and the factors that influenced how that brain development might be different in some kids than in others. A really important consideration is: does it make any difference? The brain is used for many things, but one is for thinking and memory tasks. If we pull in the neurocognitive performance data collected in ABCD, we can help to snap on the meaning of brain developmental differences.

Do you have any tips that you want to give to the participating teams?

Susan: I guess a little bit of literature review would be a really good idea! Think through what we know about brain anatomy and cognition. I think that would be a helpful hint.

Are there any mistakes to avoid?

Susan: I think if there's a finding that's very different from what's seen in the literature, you'd want to be more cautious.

Wes: We're trying to have people avoid making trivial predictions based on things that are not really relevant. One of the things that we residualize on is brain volume or head volume. We don't want to key in on how big the head is or how big the brain is. Kilian can explain that better. We're also residualizing socio-demographic factors because we don't want people to focus on things that might be correlated with the brain structure, but not intrinsic to the performance of the brain.

We're also residualizing on data collection site because it is a multisite study with 21 sites. We don't want people to key in again on site differences that would be irrelevant for making external inferences about what the relationships are that people are finding. We're trying to make sure, as much as we can, that people don't come up with algorithms that are senseless in the sense that they don't generalize or say anything mechanistically about how the brain is related to neurocognition. Having said that, I agree with Susan completely that a good tip might be to review the literature and see what people know about the relationship between brain and neurocognition. How could it be built into a machine algorithm to improve its performance for its interpretability

Wes Thompson is Associate Professor of Biostatistics at the University of California in San Diego and the Associate Director of the Data Analysis and Informatics Core for the Adolescent Brain and Cognitive Development Study, in charge of consortium and biostatistics

or both?

Kilian: While it's not tested for in the challenge, it would be really nice if people don't just apply approaches to do prediction on the score, but also to let us know how they came to those results. What changes in anatomy cause to drive the prediction of the fluid intelligence scores? That would be really helpful for ABCD and, having said that, there is no way of testing for that in the challenge so we didn't attempt to do that, but it gives additional brownie points.

Wes: We're not observing changes in brain matter. We're observing differences in brain matter between children. This is the baseline data. As Susan said, this is a developmental study, a longitudinal study. We'll be repeating brain scans on the same children multiple times over 10 years. In fact, every other year we'll get another full brain scan with all the different modalities that Kilian mentioned before. We'd like to, hopefully, depending on how well-received this initial challenge is, to do future challenges where we actually do individual-level predictions. That remains to be seen how we actualize that, but then we would use longitudinal imaging data on kids to predict their future values on some quantity, maybe not neurocognition. Maybe it has something to do with psychopathology or something else. Going forward, we'd like to continue these contests using the longitudinal data and see how well we can predict how trajectories change within a person.

Kilian: We added a lot of information on our website. We will also make available the scoring algorithm, every component of it. If people have any questions, that's something to go on to. It takes a little bit of time to get access to the data because NDA (National Institute of Mental Health Data Archive) requires a sign-off by the institution, not by the participant, but by the institution of the participant to get access to it. People should start early. This can take several weeks. Often the institutions that people are affiliated with take a while to fill out these forms. Keep that in mind when trying to participate in the challenge. There is also an email list people can sign up to and a little tutorial on how to apply for access to NDA. I hope we'll see a lot of the people at MICCAI 2019 in October when we are organizing a workshop about the results of the challenge.

Susan: We're really interested to see what kind of algorithms we generate. This is really exciting data. **Good luck!**

Women in Computer Vision

by Ralph Anzarouth

Women in Computer Vision (also called Women in Science) is a series of interviews conducted by Ralph Anzarouth. New interviews are regularly published on all RSIP Vision's publications: Computer Vision News and the Daily magazines (CVPR Daily, MICCAI Daily and many more).

Find now on the project page the direct links to almost 100 interviews... **at the click of a button**

Leadership

Mentoring

Competence

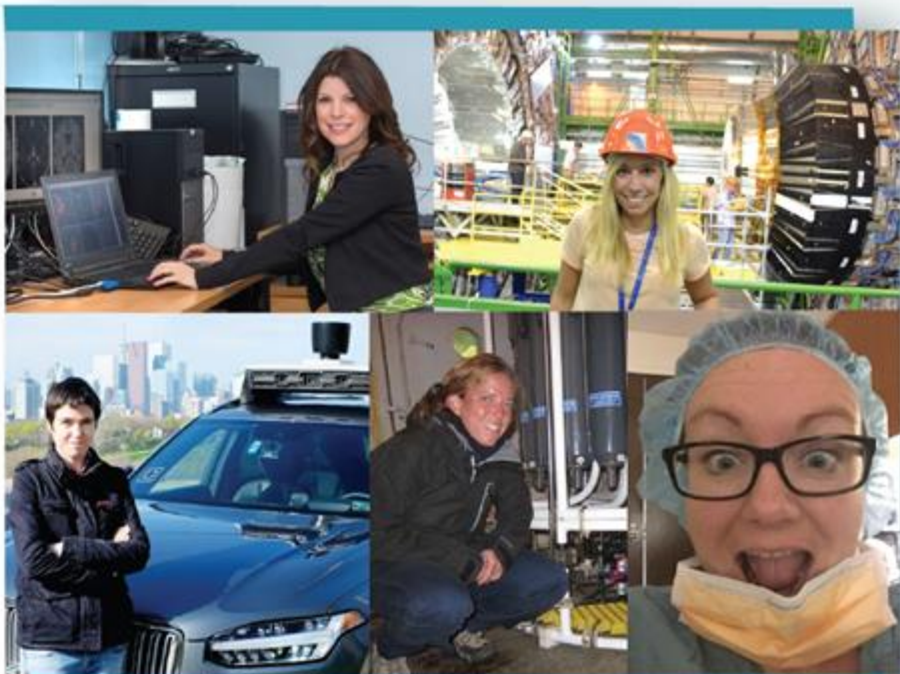
Confidence

Community

"The only way to succeed is to really start believing in yourself!"
Michela Paganini

"Most of all, you have to believe that you can do it!"
Laura Leal-Taixé

"It may look like a long list of names, but behind each name there is a fascinating world in which we were let in."
Ralph Anzarouth



Did you miss an interview? No worries, you can find them all in the **Women Scientist section of RSIP Vision's website**

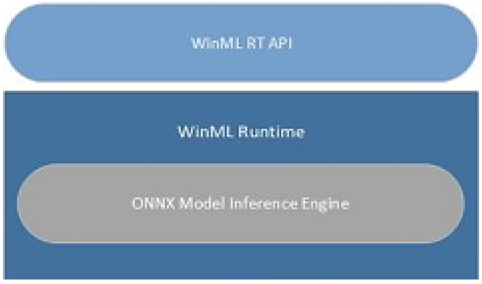
A Full Hardware Guide to Deep Learning:

Let start with something useful. **Very useful!** More often than not, you know what software to use for your project. If you don't, ask **RSIP Vision**. But deep learning is very computationally intensive: do you always know how powerful should hardware be to make it happen? GPUs, CPUs, RAM, HD/SSD, PSU... This Guide Knows...

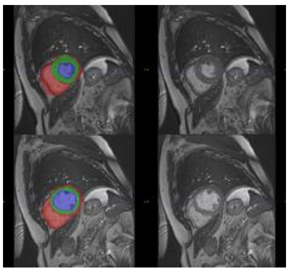
The industrial tech to watch in 2019:

Edge computing, **computer vision**, and "cobots" will have a huge impact on industrial manufacturing in the coming year and beyond. How do we know? **Stacey on IoT** tells us in her review of a new report by **CB Insights**. Not sure we agree with the quadrants. Do you? Read Here...

Microsoft open sources the ONNX runtime engine for ML:



Following its alliance with **Facebook** around the **Open Neural Network Exchange (ONNX), Microsoft** open sources the inference engine at the heart of its Windows machine learning platform. They are making it available on **GitHub**. Microsoft also made its **Azure Machine Learning** service generally available. Enjoy!



Will We Ever Solve The Shortage of Data in Medical Applications:

In the age of deep learning, data has become an even more critical resource than it used to be. The case of **medical data** is even more complicated, mostly because of patient privacy, which is protected by patient data laws that differ from country to country. What are the solutions? Three suggestions here...

Google's Drone Delivery Wing starts a pilot in Finland:

Wing, now a full company in the **Alphabet** group, will begin a drone delivery pilot in **Helsinki**, delivering goods and packages of up to 1.5 kilograms within a distance of up to 10 kilometers. The service is initially free and orders are placed via the app. Read More...

An Eye-Scanning Lie Detector and a Baby-Sitter Screening AI:

Two separate articles about two separate apps. What do they have in common? Well, oftentimes artificial intelligence and computer vision solve important problems. Sometimes, they are employed in a sort of **creepy, uncomfortable way**. What do you think of this app and of that app?

Another nice interview with **Yann LeCun** here. Did you read the one we did?
Another nice interview with **Yoshua Bengio** here. Did you read the one we did?
AI in art? We liked the term "Artificial Stupidity"... Read a curator's opinion here...

by Assaf Spanier

These tools help you follow all the training processes and results

The tip of this month will review two tools to help you train your neural network more systematically. These tools help you follow all the training processes and results in one place in a very user-friendly way. The first tool is actually a website -- **comet.ml** -- which interfaces with your code to enable you to see the outcomes of your runs online. The second tool is called **Hyperas** and it allows you to define what network configurations you would like to run and test, using standard **deep learning** libraries, such as Keras.

Let's start with comet.ml

Comet enables you to track your **Machine Learning** experiments, facilitating comparisons you may want to make and collaborations. It allows tracking hyperparameters, metrics, code, stdout, etc. and supports standard libraries -- Keras, TensorFlow, PyTorch, scikit-learn out of the box, and other libraries with the manual API.

Let's try it out to see how it works.

First, you need to install the package locally, on your own computer, like this:

```
pip install comet_ml
```

To use comet.ml, you need an `api-key`, which you'll receive after you set up an account on their website. You use your `api-key` as an `Experiment class` parameter -- as seen in the snippet below. This is what links your code to the website.

```
from comet_ml import Experiment
experiment = Experiment(api_key="YOUR_API_KEY")

# Your code.
```

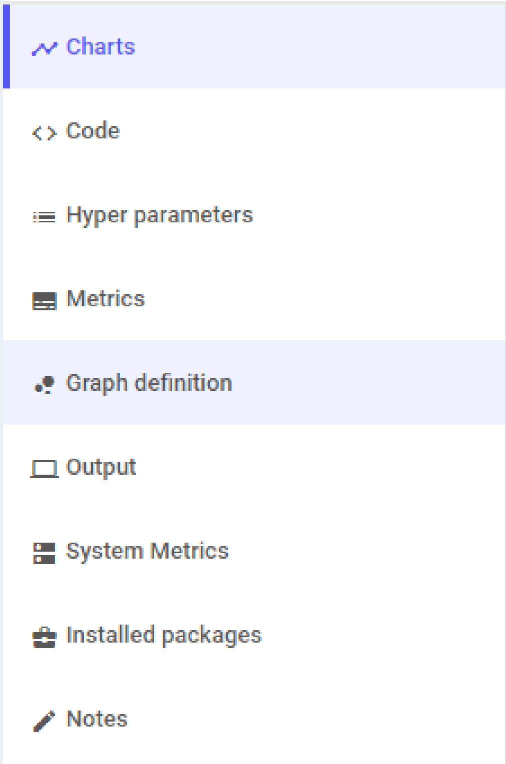
For each run of your code, comet.ml will report run outcomes, results and the values of whichever parameters you define. You list the parameters you want comet.ml to monitor using the `log_parameter` function. For example, in the snippet below we instruct comet.ml to monitor `batch_size`:

```
from comet_ml import Experiment

#if "my project name" does not already exist, it will be created.
experiment = Experiment(api_key="YOUR_API_KEY",
                        project_name="my project name")
batch_size = 4

experiment.log_parameter("batch_size", batch_size)
```

Comet.ml allows us to monitor and analyze training and testing separately, using the functions `Experiment.train()` and `Experiment.test()`, which define separate contexts for training and testing; there is also an equivalent function for validation -- `Experiment.validate()`.

Now, we'll look at a complete example, using Keras (in the code below we omit the import and data loading -- to save space). The code starts by defining the `Experiment class`, which connects to the comet.ml website. We then define 6 parameters for monitoring (batch size, epochs, etc). Next, we define a really basic two-layer network. And finally we run training and then testing sessions using the appropriate `Experiment.'session'()` functions.

```
#create an experiment with your api key
experiment = Experiment(api_key="YOUR_API_KEY",
                        project_name='mnist',
                        auto_param_logging=False)

batch_size = 128
num_classes = 10
epochs = 20
num_nodes = 64
optimizer = 'adam'
activation = 'relu'

#these will all get logged
params={'batch_size':batch_size,
        'epochs':epochs,
        'layer1_type':'Dense',
        'layer1_num_nodes':num_nodes,
        'layer1_activation':activation,
        'optimizer':optimizer
}
model = Sequential()
model.add(Dense(num_nodes, activation='relu', input_shape=(784,)))
model.add(Dense(num_classes, activation='softmax'))

model.compile(loss='categorical_crossentropy',
              optimizer=optimizer,
              metrics=['accuracy'])

#will log metrics with the prefix 'train_'
with experiment.train():
    history = model.fit(x_train, y_train,
                        batch_size=batch_size,
                        epochs=epochs,
                        verbose=1,
                        validation_data=(x_test, y_test),
                        callbacks=[EarlyStopping(monitor='val_loss',
min_delta=1e-4,patience=3, verbose=1, mode='auto')])

#will log metrics with the prefix 'test_'
with experiment.test():
    loss, accuracy = model.evaluate(x_test, y_test)
    metrics = {
        'loss':loss,
        'accuracy':accuracy
    }
    experiment.log_metrics(metrics)

experiment.log_parameters(params)
experiment.log_dataset_hash(x_train) #creates and logs a hash of your data
```



Now, once the code run ends, you can go to the website at any time and see the experiment outcomes report. The outcomes of each run are stored separately, and for each run the report is organized as a navigation bar like the one to the left, which shows all the parameters you can look at.

Under Charts, you can find graphs for the accuracy and loss metrics, and any other parameter you choose to define for monitoring in the `log_parameters` function -- the graphs are very similar to those of TensorFlow's TensorBoard. Additionally, comet.ml stores the code of a run alongside the report of that run (under Code); the output is stored (under Output) and so are all the packages that were installed for that run (under Installed packages). This systematic storage of all possible data, from installed packages, the code, the network graph, the precise parameters next to the output -- enables you to better analyze your data.

It is not unusual in the course of development to keep only the run outcomes, without meticulous notation of every last detail of all parameter and network settings, not to mention the different versions of packages installed. The result is that it becomes difficult to impossible to reliably reproduce your results later.

Hyperas

Hyperas is a convenient and simple package that is built for training deep learning network using Keras. Hyperas allows you to quickly train networks in different configurations and with different hyperperameter for selecting the optimal configuration for your data and network. Hyperas allows you to use the power of hyperopt without having to learn its syntax. All you need to do is set up your Keras model in the usual way with only a simple template of scroll brackets (which we'll immediately demonstrate) to define options and ranges for evaluation. .

First, you need to install the package locally, on your own computer, like this:

```
pip install hyperas
```

Let's say you have the following network set up using Keras that you want to train:

```
def create_model(x_train, y_train, x_test, y_test):
    model = Sequential()
    model.add(Dense(512, input_shape=(784,)))
    model.add(Activation('relu'))
    model.add(Dropout(0.2))
    model.add(Dense(512))
    model.add(Activation('relu'))
    model.add(Dropout(0.2)
    model.add(Dense(10))
    model.add(Activation('softmax'))
```

Using Hyperas, you need nothing more than scroll brackets to define the range of values or options you want evaluated for each hyperparameter or network configuration element, and Hyperas will tune them for you. For our example, we shall evaluate `Dropout` values changing at a uniform rate between 0 and 1, compare two different activation functions (`relu` and `sigmoid`), and 3 possible values for number of neurons in the dense layer: 256, 512 and 1024. Three optimization algorithms will be evaluated: adam, rmpProp and SGD. All of this code will be written inside a `create_model` function, which will both compile the model and train it. This function will be sent to `optim.minimize`, which will return the best performing model.


```
   model = Sequential()
    model.add(Dense(512, input_shape=(784,)))
    model.add(Activation('relu'))
    model.add(Dropout({{uniform(0, 1)}}))
    model.add(Dense({{choice([256, 512, 1024])}}))
    model.add(Activation({{choice(['relu', 'sigmoid'])}}))
    model.add(Dropout({{uniform(0, 1)}}))

    # If we choose 'four', add an additional fourth layer
    if {{choice(['three', 'four'])}} == 'four':
        model.add(Dense(100))

        # We can also choose between complete sets of layers

        model.add({{choice([Dropout(0.5), Activation('linear')])}})
        model.add(Activation('relu'))

    model.add(Dense(10))
    model.add(Activation('softmax'))

    model.compile(loss='categorical_crossentropy', metrics=['accuracy'],
                  optimizer={{choice(['rmsprop', 'adam', 'sgd'])}})
```

Finally, let's see how you can use both tools together: In the following example we will evaluate a number of network architectures using Hyperas and preserve the outcomes of the evaluation in a systematic way for future reference, using comet.ml. When you run the code below and its run ends, you will be able to go online to the comet.ml website and see the run outcomes for each run, that is for each network configuration.

- Lines 1-11: Upload the modules
- Lines 13-20: Upload the data
- Line 23 defines the link to comet.ml
- Lines 26-43 define the model of the network to be trained, using scroll-quotes to define hyperparameter settings, ranges and configuration element alternatives to be evaluated using Hyperast -- Hyperas will run all combinations specified.
- Lines 46 to 58 call the functions for training the network; and to store parameters on comet.ml.
- Lines 66 to the end: This is the MAIN, which executes the entire code.

```
1.  from __future__ import print_function
2.  from comet_ml import Experiment
3.
4.  import numpy as np
5.  from hyperopt import Trials, STATUS_OK, tpe
6.  from keras.datasets import mnist
7.  from keras.layers.core import Dense, Dropout, Activation
8.  from keras.models import Sequential
9.  from keras.utils import np_utils
10. from hyperas import optim
11. from hyperas.distributions import choice, uniform
12.
13. def data():
14.     (x_train, y_train), (x_test, y_test) = mnist.load_data()
15.    x_train = x_train.reshape(60000, 784) ; x_test =
    x_test.reshape(10000, 784)
16.    x_train = x_train.astype('float32') ; x_test =
    x_test.astype('float32')
17.    x_train /= 255 ;  x_test /= 255 ; nb_classes = 10
18.    y_train = np_utils.to_categorical(y_train, nb_classes)
19.    y_test = np_utils.to_categorical(y_test, nb_classes)
20.    return x_train, y_train, x_test, y_test
21.
22. def create_model(x_train, y_train, x_test, y_test):
23.    experiment = Experiment(api_key="",
24.                            project_name="", workspace="")
25.
26.    model = Sequential()
27.    model.add(Dense(512, input_shape=(784,)))
28.    model.add(Activation('relu'))
29.    model.add(Dropout({{uniform(0, 1)}}))
```

```
    model.add(Dense({{choice([256, 512, 1024])}}))
    model.add(Activation({{choice(['relu', 'sigmoid'])}}))
    model.add(Dropout({{uniform(0, 1)}}))

    # If we choose 'four', add an additional fourth layer
    if {{choice(['three', 'four'])}} == 'four':
        model.add(Dense(100))
        model.add({{choice([Dropout(0.5), Activation('linear')])}})
        model.add(Activation('relu'))

    model.add(Dense(10))
    model.add(Activation('softmax'))

    model.compile(loss='categorical_crossentropy', metrics=['accuracy'],
                  optimizer={{choice(['rmsprop', 'adam', 'sgd'])}})

    with experiment.train():
        result = model.fit(x_train, y_train,
                 batch_size={{choice([64, 128])}},
                 epochs={{choice([50, 100])}},  verbose=2,
                 validation_split=0.1)
        print(model.summary())
        params = {'batch_size': result.params['batch_size'],
                 'epochs': result.params['epochs'],
                 'layer1_type': 'Dense',
                 'layer1_num_nodes': model.get_layer("dense_1")
                      .get_config()['units'],
                 'layer1_activation': model.get_layer("activation_5")
                   .get_config()['activation'],
                 'optimizer': type(model.optimizer)
                 }
    #get the highest validation accuracy of the training epochs
    validation_acc = np.amax(result.history['val_acc'])
    print('Best validation acc of epoch:', validation_acc)
    return {'loss': -validation_acc, 'status': STATUS_OK,
                  'model': model}

if __name__ == '__main__':
    best_run, best_model = optim.minimize(model=create_model,
                                          data=data,
                                          algo=tpe.suggest,
                                          max_evals=1,
                                          trials=Trials())
    X_train, Y_train, X_test, Y_test = data()
    print("Evalutation of best performing model:")

    experiment = Experiment(api_key="",
                            project_name="", workspace="")

    with experiment.test():
        print(best_model.evaluate(X_test, Y_test))
    print("Best performing model chosen hyper-parameters:")
    print(best_run)
```

Osmo is an award-winning digital gaming system that allows children to play games using their hands and physical objects, while interacting with a smartphone or tablet. For the past four years, Arnaud Brejeon has led the Computer Vision team at Osmo. Brejeon spoke with Computer Vision News about his experience in developing this fascinating new way for children to learn and play digitally.

"A mix of legacy computer vision and machine learning"

At **Osmo**, they make games for children, but with a twist! Rather than using objects with electronics inside, instead, **the game uses computer vision** to detect the movement and placement of objects as the children play with them in front of a tablet screen or smartphone. **Everything runs on the device** and it doesn't require sending images to the cloud. This makes their product good for feedback as well as for privacy.

Brejeon heads a small team of four people in a company of about 50 employees altogether. Their team overcomes many challenges. First, they must meet the high expectations of game designers. At the same time, kids themselves seek a high level of accuracy in the games. Otherwise, they will lose interest and give up. That means that Brejeon and his team must create games with a **very high accuracy, between 95% to 98%**, that kids will find interesting to play.

The third challenge involves figuring out how to **run algorithms on the devices** with enough speed and accuracy. The technology behind their product works in the area of robotics and uses **ad hoc algorithm pipelines tailored for each game**. In some cases, games can have false positives while other games cannot. Prior to creating each game, Brejeon and his team discuss with game designers to decide whether a game can include false positives, along with other features such as the hardware.

Osmo started in 2013 and released their first game in 2014. Since, they try to launch three to four games each year.

"*We've been quite busy!*" laughs Brejeon. In one of their earliest games called **Words**, two players compete to guess and spell on-screen images. If an elephant appears on the screen, for example, each player tries to spell the word first. The device will recognize the words written on the pages to decide which player gets a point. In some of the harder rounds, the word that they need to spell may not seem so obvious. Perhaps a mother elephant appears, and the kids will need to spell the word "*mom*" or "*mother*".

In the **Numbers** game, children add, count, and multiply tiles to match bubbles and free the fish. **The kids practice math without even realizing it**. *"We always try to put these kinds of tricks, so they can learn"* says Brejeon.

Osmo also has its **Hot Wheels MindRacers** game, a competition between two players in which kids send toy cars racing down the ramp into all kinds of digital worlds. Meanwhile, the **Monster** game teaches kids to draw and create animated activities.

Osmo targets children between the ages of 5 and 12. They sell their games for personal use at home and to schools throughout the United States and Europe that use the interactive games in their curriculums. As Brejeon puts it, *"We believe that by playing with tangible, real objects using digital, you can learn a lot from playing games."*

At first, Osmo games worked solely on **iPads**, but today, the product also supports the **iPhone** and **Amazon Fire**. The games work in a number of different languages including English and most European languages.

During the early stages of development, the gaming technology did not use **machine learning** at all. Brejeon and his team began adding machine learning little by little. Although they would like their games to use mostly machine learning in the future, currently the product uses what Brejeon calls *"a mix of legacy computer vision and machine learning"*.

Brejeon would like to incorporate the new ideas they have come up over the years to their games developed several years back. They can rework some of these games to make them even better. For example, they can consider ways to build on the Words game by using the same digital objects.

Looking ahead, Osmo has an **SDK** in case anyone wants to contact them to assist in the development of future games. They are always looking for talented people, and Brejeon invites anyone interested in joining their team remotely to get in touch.

WELCOME TO Fabulous COMPUTER VISION EVENTS

WACV: IEEE Winter Conference on Applications of Computer Vision
Waikoloa Village, Hawaii — Jan 7-11 — Website and Registration — MEET US!

CES 2019 - Consumer Electronics Show
Las Vegas, NV — Jan 8-12 — Website and Registration — MEET US!

PrecisionAg VISION Conference
Seattle, WA — Jan 14-16 — Website and Registration

RE•WORK Deep Learning Summit
S.Francisco, CA — Jan 24-25 — Website and Registration

SPIE Medical Imaging
S.Diego, CA — Feb 16-21 — Website and Registration — MEET US!

ICPRAM Intl. Conf. Pattern Recognition Applications & Methods
Prague, Czech Republic — Feb 19-21 — Website and Registration

BMVA: British Machine Vision Association - Deep Learning in 3D
London, UK — Feb 20 — Website and Registration

VISIGRAPP - Computer Vision Imaging and Computer Graphics
Prague, Czech Republic — Feb 25-27 — Website and Registration

TRI-CON - International Molecular Medicine Tri-Conference
S.Francisco, CA — Mar 10-15 — Website and Registration — MEET US!

IAPR Computational Color Imaging Workshop
Chiba, Japan — Mar 27-29 — Website and Registration

Did we forget an event?
Tell us: editor@ComputerVision.News

FREE SUBSCRIPTION

Dear reader,

Do you enjoy reading Computer Vision News? Would you like to receive it **for free in your mailbox** every month?

Subscription Form
(click here, it's free)

You will fill the Subscription Form in **less than 1 minute**. Join many others computer vision professionals and receive all issues of Computer Vision News as soon as we publish them. You can also read Computer Vision News in **PDF version** and find in **our archive** new and old issues as well.

We hate SPAM and promise to keep your email address safe, always.

Did you read the Feedback of the Month? It's on page 21

FEEDBACK

Dear reader,

If you like Computer Vision News (and also if you don't like it) we would love to hear from you:

Give us feedback, please (click here)

It will take you only 2 minutes. Please tell us and we will do our best to improve. Thank you!