LEAN AI

HOW TO CREATE SUCCESSFUL COMPUTER VISION SYSTEMS IN PRODUCTION

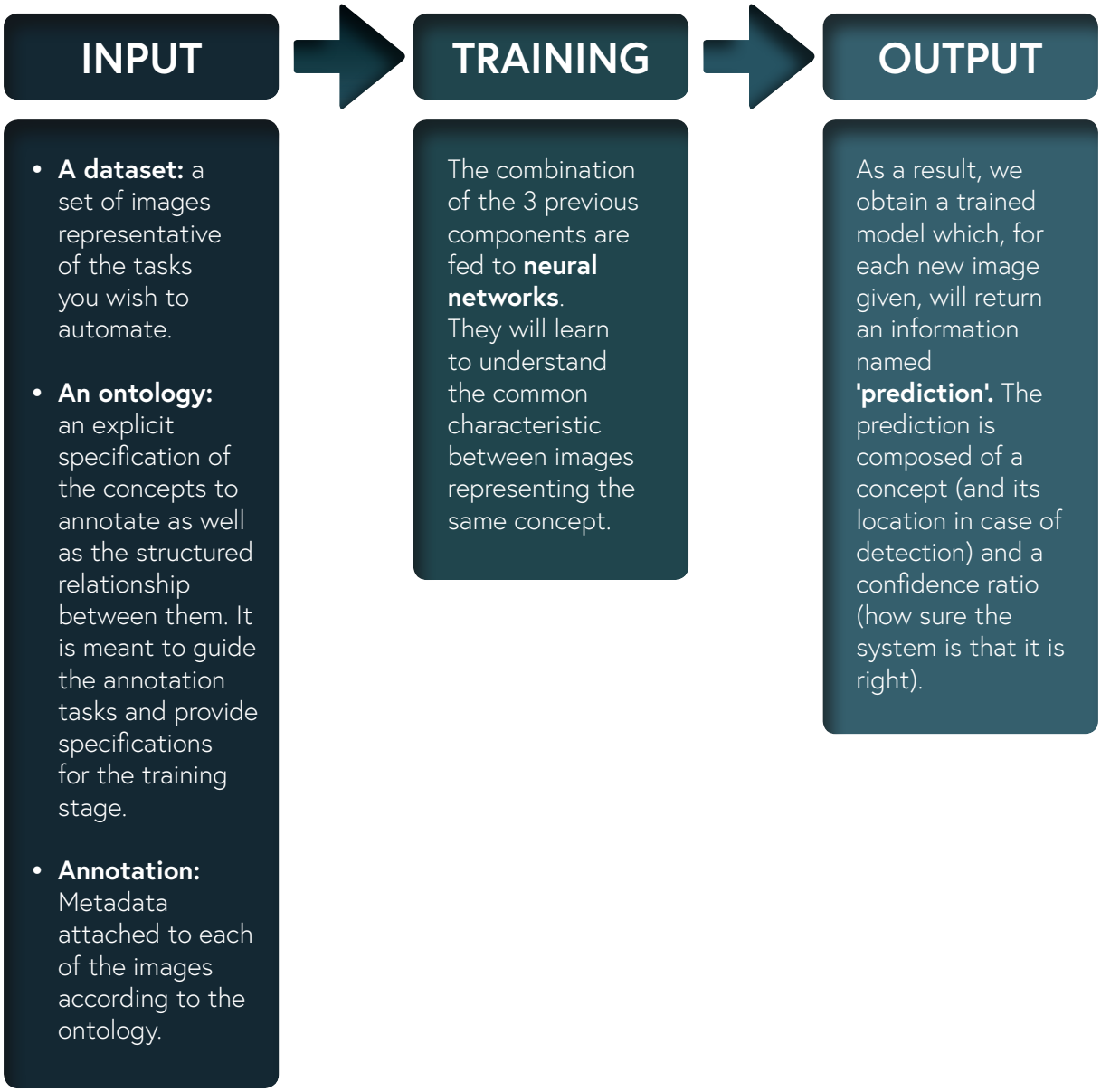
deepomatic

TABLE OF CONTENTS

BEFORE WE BEGIN...

If you are not familiar with computer vision programming, you can read **our guidebook** on **"How to build your computer vision systems in 6 steps"** before you resume your reading.

In short, a **computer vision system** consists of labelled images or videos that are put through a neural network in order to **train a model to automate a given task.**



INTRODUCTION

Deep Learning and its application to computer vision in particular, has generated a remarkable amount of attention these past years, due to the vast range of tasks it applies to and the level of performance reached. Every month, researchers seem to push the limits back by showcasing impressive new applications. The number of visual tasks seemingly solved by deep learning seems to grow exponentially.

However, despite all its promises, this technology still fails to produce the same results in the industrial world. Except for well-known examples such as autonomous vehicles, Amazon Go automated checkout system or face recognition, the applications of computer vision seem to get confined to anecdotal use-cases. And if companies advertise lab experiments from time to time, it is difficult to find examples of large scale industrialization of these systems.

After having helped many customers automate their visual tasks at industrial scale, we have witnessed how deep learning promises can be fulfilled and massively unlock business value. What we have found out is that the lack of successes in the industrial field is mostly due to an outdated methodology used by most organizations to drive their AI initiatives. As it turns out, it is very close to software development methodologies used in the first days of software development, and which have been replaced in the mid-90s.

The result of our work is summarized in the present document. We propose a new framework inspired by the recent trends in Agile, Lean and Devops, called **Lean AI.** The goal of this methodology is to minimize the risks of failure in the field of industrial AI, as well as maximizing the performance of AI systems. We start by exposing the outdated development methodology and the reasons it is preventing organizations to succeed in their efforts to generate business value using deep learning. We then introduce the **Lean AI** framework, and show how it can solve the many problems that plague AI in production. Finally, we discuss the challenges associated with implementing Lean AI.

Aloïs Brunel, CPO @ Deepomatic

WHY IMAGE RECOGNITION DOES NOT HAVE THE INDUSTRIAL SUCCESS IT DESERVES

We have found out that there is **one main reason behind the lack of** significant **industrial achievements** powered by modern computer vision and deep learning in general. This reason is the **wrong approach** taken by most industry actors to building and industrializing computer vision projects.

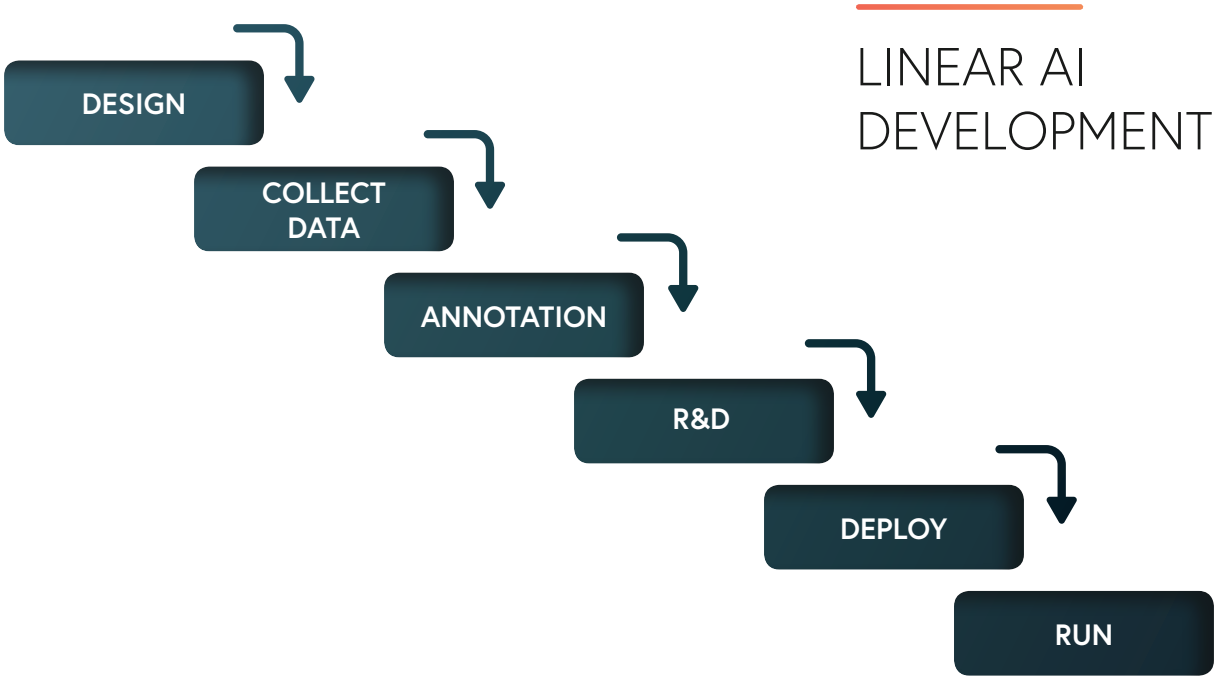
> Since the emergence of deep learning in business, the academic approach has prevailed. By focusing on optimizing model on fixed hypotheses, the resulting systems became abstracted from the real-world and failed to perform in production.

The deep learning flavor of computer vision first emerged as a success on the academic side. When this approach took over all other competing ones on research benchmarks, it started a deep learning craze which in turn led companies to try benefitting from this new technology. As the industry experience with deep learning was basically non-existent, companies started tapping into the pool of deep learning researchers and PhD students to lead their AI efforts, resulting in **methodologies with a strong emphasis on machine learning and prototyping.**

Academic research mostly focuses on developing new algorithms and techniques that will maximize the performance on given reference tasks. In the machine learning field, this means that the algorithms are trained on reference fixed datasets and evaluated on reference datasets. One such well-known dataset is ImageNet. It provides one of the most important research competitions: teams all over the world are fighting to improve the state-of-the-art on this dataset. As we can see, most of the academic deep learning activity is dedicated to improve (even slightly) these reference performances.

The dominant approach to developing industrial computer vision projects is directly borrowed by this academic way of doing things, which emphasizes on tuning the deep learning architectures to get the best results out of the biggest possible dataset. This Linear AI Development model looks like this:

- **Design** a system inputs/outputs (e.g, the classes you want to recognize) based on the current understanding of the problem.
- **Gather** as much raw **data** (images) as possible.
- **Annotate** this data to constitute a training dataset.
- Use **research** scientists to tune the deep learning architectures, develop new techniques that will get the best performance on a reference evaluation data set.
- **Deploy** the system obtained at the previous step in production.
- Evaluate the **performance** in real conditions to find out if it works as intended (spoiler: it often does not).



For those who are familiar with the history of software development methodology, you may recognize something similar to the old **Waterfall Model**, that reigned over software development before the advent of agile methodologies. The Waterfall model is a development process in a linear sequential flow in which the next phase of development begins only if the previous phase has been completed.

WHY IS LINEAR AI DEVELOPMENT FUNDAMENTALLY FLAWED IN A BUSINESS CONTEXT ?

1. The **conditions of production** are difficult to anticipate and evolve over time because of operational circumstances.

2. The **data set** used to train the algorithms does not have to be **fixed.**

3. **Developing** in-house **neural network** architectures is time-consuming and expensive, with little guarantee of improving performance.

4. It takes a lot of **time to go** from the design phase **to the production** phase, which makes it difficult to detect and fix any problems.

5. For high-impact AI projects, it is often required to **deploy** the AI systems **at the edge**, which implies spending more time on setting up an IoT infrastructure.

TOWARDS A LEAN AI METHODOLOGY AND TOOLS

FROM LINEAR SOFTWARE DEVELOPMENT TO THE LEAN & DEVOPS SOFTWARE REVOLUTION

Historically, software was developed in a linear fashion, following methodologies like the Waterfall Model. Years could separate two consecutive software releases. Strong assumptions were made during the design phase, followed by a long and costly R&D and software development. Finally, heavy testing phases were organized ahead of release time in order to avoid bugs. This model is in a sense very similar to the current dominant AI methodology that we described in the An outdated methodology section.

If "software is eating the world" (Marc Andreessen), this is in part due to the massive adoption of **lean, agile,** and finally **devops methodologies** by software companies after 2000. By emphasizing on **frequent releases, waste removal, quick iteration cycles, testing**, even small organizations were able to build amazing products that would change the world at an ever faster pace.

LEAN, AGILE, DEVOPS METHODOLOGIES : WHAT'S THE DIFFERENCE ?

- **Agile** : advocates **early delivery, continuous improvement,** and **focuses on rapid responses to changes**. It focuses specifically on software development.
- **Lean** : theorized the **Build-Measure-Learn**[1] **loop** to build products that were actually solving user problems and got better as they were used.
- **Devops** : (and in particular continuous integration and continuous delivery) unleashed the full potential of the lean and agile methodologies by providing the technical capabilities to test and **deliver software instantaneously in production.**

It is clear to us that AI should have its own agile/lean/devops revolution in order to reach a state of **maximum business value production**. We propose the **Lean AI** methodology, as a step forward in that direction.

[1] The Lean Startup, Eirc Ries

deepomatic

LEAN AI

Lean AI borrows from both lean manufacturing and lean startup methodologies:

- **Lean Manufacturing** emphasizes on **"How to build better products",** in a context where the product we want to build is well defined. The emphasis is put on the **continuous improvement of the product** building process by removing waste[2].

- **Lean startup** emphasizes on **"How to find the right product to build"**, in a context of market uncertainty. What does the final user want? It focuses on the notion of validated learning obtained via a Build-Measure-Learn loop.

In the context of **AI systems**, the final product is generally already well understood: we know how it is going to be used in production and the business value it is meant to generate. The uncertainty lies on how to produce the right AI that implements that product, and the uncertain (or changing) conditions of production in which the system is going to operate. By employing **Lean AI**, we seek to **improve the product building process by removing certain waste while navigating the uncertainty of production conditions.**

THE THREE MAIN INGREDIENTS OF LEAN AI ARE:

- THE MINIMUM VIABLE AI
- THE LEAN AI LOOP
- SHORT CYCLE IMPLEMENTATION

[2] In this context, the term waste encompasses anything that is a non value-added process.

deepomatic

MINIMUM VIABLE AI

To kick-start the Lean AI Loop, it is crucial to get the **first version of the AI system in production as early as possible**: this is the **Minimum Viable AI** (or **MV.AI**). This first version is based on a set of assumptions (the visual ontology and the set of images to annotate first) that should be chosen on the basis of a well-thought analysis, while keeping the size of this first batch of data contained. Keep in mind that since the ontology and the set of images could change drastically after a few loop iterations, the build phase should not take too long. A good MV.AI should:

- **be based on reasonable assumptions about the production conditions**
- **take only a few days to build**

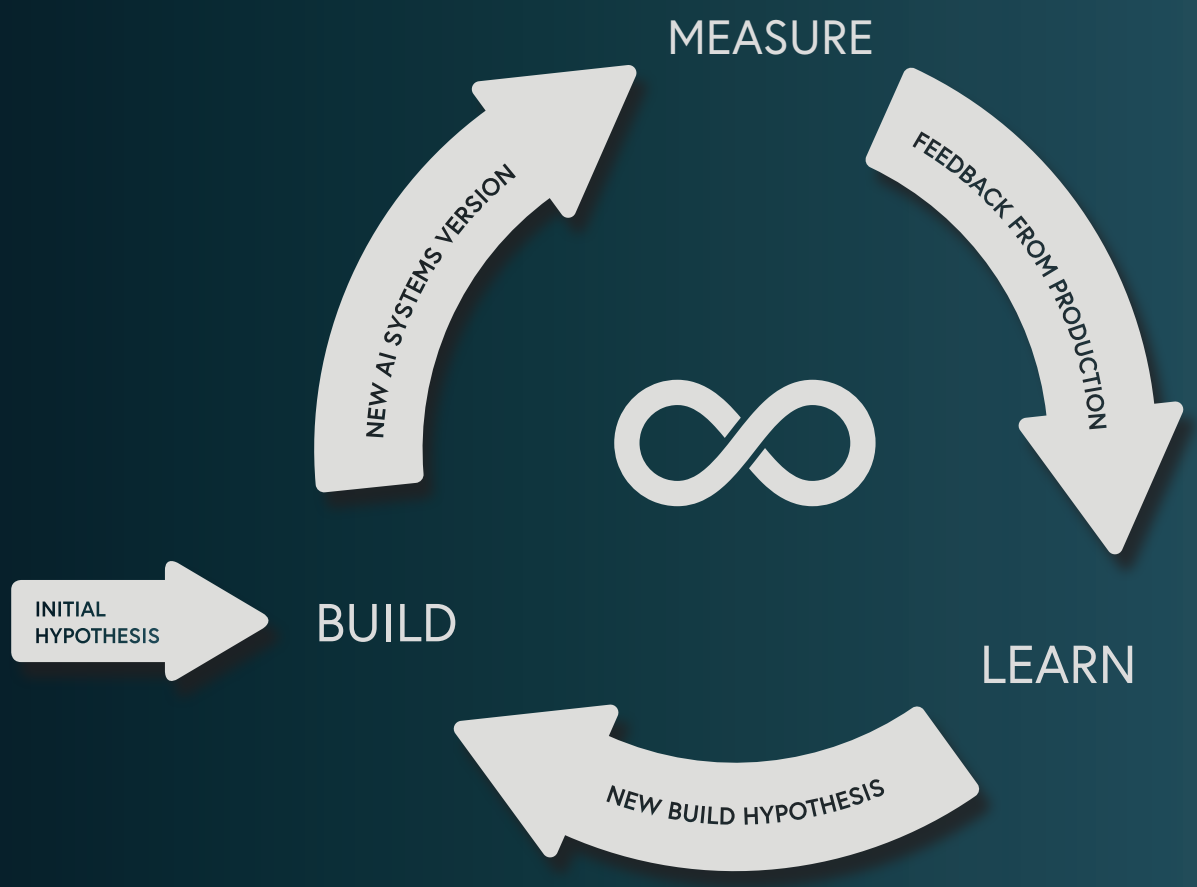
This implies that if the problem to solve is very complex, the MV.AI could very well only **solve a part of this problem**. Moreover, the MV.AI should follow the general rules given in the next section to **minimize the time spent on building** it.

Let's give an example: An AI system is used to detect abnormal and violent behavior in a car park in order to be able to alert security immediately in case of an emergency. The MV.AI would then be trained to recognize only one or a few suspicious attitudes. For instance, it could be tested on recognizing masked people and people lying on the floor. In this case, not all dangerous behavior is identified but it is enough to validate that the system can work in production. Later on, the system will be taught to recognize more complex concepts such as someone trying to break into a payment terminal.

deepomatic

LEAN AI LOOP

The Lean AI methodology is based on the Lean AI Loop. It is made of a series of **three steps borrowed from Lean methodologies**, which together implement an iterative process.



By repeating these steps over and over, the **performance** of the system is **improved iteratively.**

Let's quickly review the whole cycle before diving deep into each step. First, the loop is kickstarted with the following **hypotheses**:

- the **annotation ontology (**which should be chosen according to the business problem we are looking to solve)
- the **set of images** to annotate.

As we loop through the cycle and we learn how the system behaves in production, these hypotheses will evolve.

- **Build**: given your ontology and your set of images, **annotate** it accordingly while minimizing annotation errors. Then **train your models** and assemble them into a proper AI system.

- **Measure**: given a new version of your system, confront it to production. This means **deploying the system where the action takes place** and then **evaluating its performance** in production mode. This is where you want to see the impact of your loop iteration.

- **Learn**: this last step enables the possibility of continuous improvement. By gathering **feedback from the production**, we understand **where the AI system should improve**. This step may very well uncover operational conditions that you did not anticipate, and that will be integrated into the next version of your system. This feedback materializes as a **new set of hypotheses** that will be given to the next Build phase:
  - **Change your ontology**: classes are going to be merged, split, or created
  - **A new set of images to annotate**: the images on which the system is uncertain should be brought back to annotation. Among these images, those that will improve your system most should be prioritized. **Images representing specific real-world conditions** of interest will also be introduced.

deepomatic

SHORT CYCLE IMPLEMENTATION

One of the key ingredients of the Lean AI methodology is to **minimize the time needed to go through each loop iteration**. Particular **points of friction** are to be carefully handled.

1. **Build**:
   - **The size of the batch of images** to annotate should be kept reasonable (depending on the task, a few hundreds to a few thousands), so that the annotation can be done in a short amount of time while focusing on their quality.
   - **The time spent on doing R&D**, i.e: tuning the deep learning architectures **should be zero**, and state-of-the-art architectures used instead. As the system becomes better and stops progressing and for organizations that can afford it, R&D can be considered to go get the last drops of performance. In any case, R&D should be done in parallel with the Loop and should never constitute a blocking task.

2. **Measure**: once a new version of the system has been built, it has to be **redeployed on the field**. Since high-performing systems are often composed of several neural networks, this means being in the **capacity of remotely updating** heavy applications through a network of connected devices, despite partial connectivity and low bandwidth.

3. **Learn**: it is crucial to **feed back** both the raw **data points** which confuse the system and the data points on which we know the system was wrong. They are going to constitute the **prime material** for building a **better version of the system.** In the context of IoT applications (which constitute the majority of high impact applications), this means that this **data should be gathered and stored in situ**, before being **sent back to a central location** when sufficient bandwidth becomes available.

deepomatic

IN BRIEF

	BUILD	MEASURE	LEARN

ACTIONS	• Annotate
	• Train
		• Deploy
		• Evaluate
			• Collect feedback data from production and then :
			  ◦ Change ontology
			  ◦ Add images

POINTS OF FRICTION	• Size of the batch of images to annotate • Time spent on R&D	• Capacity to update applications taking into account connectivity and bandwidth constraints	• Feedback data has to be stored locally before being centralized

CONCLUSION

AI as an academic field has opened exciting new possibilities that have yet to materialize in the industrial world. Our experience has repeatedly shown that **going from a linear development model to an iterative and agile one has the capacity to unlock high-impact applications**. Other companies such as Tesla have already shared the same observations[3]. In addition, if Lean AI is particularly well suited for the computer vision expression of deep learning, we believe it applies almost immediately to the other applicative fields of deep learning such as text processing. As such, **Lean AI has the potential to unleash the true potential of AI for businesses.**

However, if some organizations will be able to develop internally the tools needed to implement Lean AI, most of them will not be able to implement it from scratch. This is why we want to insist on the need for a **new generation of software platforms**. The combination of **Deepomatic Studio®** and **Deepomatic Run®** allows the full realization of Lean AI for companies intending to **accelerate** their **AI business initiatives.**

Businesses use **Deepomatic platform** to build computer vision applications that **automate tasks on the field**. Thanks to an easy-to-use interface, operationals transfer their knowledge to an AI on **Deepomatic Studio®** and then monitor its performance. Meanwhile, developers and IT teams benefit from **Deepomatic Run®** to seamlessly deploy computer vision applications while keeping them connected to **Deepomatic Studio®** for feedback reception.

[3] Building the Software 2.0 Stack, by Andrej Karpathy, Director of AI @ Tesla

PRATICAL IMPLICATIONS

BUILD

ANNOTATION & AUTOMATED TRAINING

The task of training models should be done in conjunction with annotation, as a kind of small iterative loop. By breaking down the annotation batch into smaller sub-batches and training models between each of them, this allows to spot annotation quality problems and understand inherent ambiguities in the ontology that can be course-corrected on the way. These problems can be detected thanks to model predictions and data science metrics such as confusion matrices, for instance. See Learn Section.

Our take on training is that you should definitely not spend time within the AI Loop to perform research-level tuning of neural network architectures, and rather use state-of-the-art models and techniques, that will provide best-in-class performance without too much effort. Setting up an on-demand training service, built-in with classic techniques such as hyperparameter search and data augmentation will be a better time investment as it will reduce drastically the time spent on each iteration.

PERFORMANCE REGRESSION TESTING

When we build a new version of the AI system, it is very important to ensure that this system is at least as good as the previous one in terms of performance, before deploying it in production. This is why a certain form of regression testing is needed. In classical software development, regression testing typically checks that your new software does not break any existing features. In the case of AI, you want to check that the production performance will only be impacted positively. The best way of doing it is to constitute a set of historical records coming from production and a simulator that gives the ability to replay the new system on this past data so that you can evaluate the theoretical performance your system would have had if deployed in production. See Performance evaluation for recommendations on choosing your performance metrics. When human / machine interaction is involved in the daily use of the system, it may be required to model the action of a human in order to be able to replay the set of historical records.

Example 1:
An organization builds a barrier-free, camera-only highway toll system. The set of historical records may be from a sample of several tolls, with video sequences and the list of vehicles which passed by the toll (as well as the information associated: the type of the vehicle, the number of wheels, etc.). A new version may be tested by confronting it to this historical record and checking that its performance is as high as the previous versions.

Example 2:
An AI system is used within a weighing machine for fruits and vegetables in supermarkets. It detects the type of fruit or vegetable and weighs it automatically. If the system has a doubt, it provides the two best guesses to the user, who has to choose the right one. In that case, it may be important to model the human behavior (is she always going to choose the right item if a choice is provided?) in order to simulate the performance of a new system on historical records.

MEASURE

CONTINUOUS DELIVERY & IOT MANAGEMENT

The first step of measuring the quality of the AI system is to deploy this system in production. In some cases, deploying to the cloud could be an option, but this is very often not the case because of connectivity, privacy or cost constraints.

In order to perform quick Lean AI Loop iterations, an organization needs to reach continuous delivery. In the IoT context, this means overcoming challenges about remote updates and version rollbacks. As deep learning based systems typically are heavy applications, and the conditions at the edge possibly being not favorable (low bandwidth, intermittent connectivity), it must be taken care of carefully.

CORRECTIVE FEEDBACK

Depending on the nature of your AI system, it may be possible to get corrective feedback. Corrective feedback are typically human validations that confirm or contradict the outcome proposed by the system. They can be gathered as part of the natural use of the system (e.g: through a human/machine interface), or as part as an organized validation campaign. This corrective feedback is a precious source of improvement for your system.

Example 1:
an AI system is used for automated checkout in a supermarket. As part of the checkout experience, the customer is asked to validate the ticket produced by the system. In that case, the correction is total as every single generated ticket is either validated or corrected.

Example 2:
an AI system is used to automate the quality control on automotive part manufacturing. When the system detects defects on a production line, the line is stopped and checked by a human expert. The expert can confirm the presence of the defect or tell the system it was wrong in that case. Here, the correction is partial as defects that are missed by the system are not spotted by the human operator, thus not corrected.

PERFORMANCE EVALUATION

Measuring the performance of deployed AI systems is key to implementing the Lean AI loop, and this should be well thought ahead of the first deployment. To understand how an AI system performs, it is important to distanciate oneself from purely data-science related metrics. Indeed, such a system is used to solve a business-specific problem and it should be evaluated on how well it solves it.

In the context of Lean AI, a good performance metrics should be:

- **Comparative:** we should be able to compare the performance with previous periods of time. In the context of an ever expanding fleet of devices, the use of cohort analysis is a good idea.
- **Responsive:** the chosen metrics should be easy enough to compute so that it minimizes the lag between execution and measurement, hence ensuring it does not slow the cycle iteration.
- Linked to **business value:** the chosen metrics should be directly contributing to a specific business KPI. This helps your organization to better understand the ROI of the system and make sure the Lean AI Loop optimizes for the right thing. In particular, data-science based performance metrics are often not the right choice here.

Corrective feedback can provide a good way of evaluating your system, as it gives an instantaneous photography of how well your system is performing. If the system is used through a human/machine interface, it could be a good idea to think about how corrective feedback can be part of the natural interaction.

Example 1: ❌
An AI system is used to help technicians during the maintenance of optical fiber networks, by detecting potential defects. The organization monitors the improvement in terms of the number of calls for on-site re-intervention. This is good business KPI but it is not responsive, hence does not allow to drive rapid Lean AI Loop iterations.

Example 2: ✔
An AI system is used to provide alerts when non-conforming waste is about to enter a sorting machine and hence could break it (a non-conforming event). Humans are already monitoring several sorting lines using cameras. Two performance metrics should be evaluated:

- The % of missed events
- The % of events that were not detected by humans, that were detected by the system

These metrics are directly related to business metrics as the cost of each non-detected event is known, and can be made responsive (computing these metrics require thinking about how humans interact with the alerting system).

LEARN

The main use of the learn phase is to generate a new set of assumptions for the next cycle of the Lean AI Loop. Remember that the assumptions are composed of:

- A new set of images to annotate
- Modifications to the ontology

CLOSING THE LOOP

The obviously crucial component of implementing the Lean AI Loop is of course the ability to **close the loop.** This means that once a system is in production, it has the possibility of sending back data to a central location, in order to constitute the new set of images to annotate. In the context of IoT devices, this may be a challenge and choices usually have to be done:

- **How much data is sent back.** If you can, sending back the entire set of interactions with the system is obviously the best choice. This means the raw data (images or videos), the predictions made by the system, and the corrections (as mentioned in the last section) if available. However, if it is not possible, you may consider only feeding back a sample of this information
- **Feedback selection.** In the case where sending all the data is not an option, you may consider filtering it to only surface the most important data. This means prioritizing:
  - The data points on which the AI system is confused.
  - The data points on which the AI has been wrong (only possible if corrections are available).
  - The data points that satisfy an operational criterion that you know is of interest.
- **When the data is sent back**. Depending on the way the AI system is used, it may be smart to send back the data only at specific times. Example: if an AI system is used to provide automated checkout in stores, the data can be sent back during the closing times of the stores.

ACTIVE LEARNING

Finally, given the amount of data generally sent back by the systems in production, it is important to prioritize the raw data to annotate. It allows your annotators to take care first of the data points which were the most problematic for the AI system, and which are going to contribute the most to the improvement of the overall system. This can be done in many ways, and generally by techniques under the "active learning" umbrella: the images that maximize the confusion of the system, or that trigger negative corrective feedback (See Corrective feedback). The same techniques can be used to identify the possible weaknesses of your current ontology: by identifying the classes which often get confused by the systems in production, the ones that never get predicted or very rarely. From a general point of view, exploring this data using sorting techniques will help you get a better grasp of what is going on in the field.

DOWNLOAD OUR WHITE PAPER TO DISCOVER THE 6 STEPS TO BUILD A VIDEO VISION SYSTEM

ABOUT DEEPOMATIC

Deepomatic enables companies to automate their business-specific processes through computer vision. Our end to end solution leads to increased productivity, allowing both employees and companies to focus on their key expertise.

Through our products, heads of operations and operational staff can build their own AI applications and operate them at scale, within 3 months and without coding skills.

Every day, our clients use our technology to run the most advanced use cases in the world.

- In their restaurants, the Compass Group, world leader in contract foodservice, has implemented a checkout system identifying and pricing in real time each item on the meal tray. This results in increased productivity, less waste, and enhanced guest experience in every restaurant.
- Bouygues Telecom equips and trains its technicians with real-time image recognition in the field. As a result, the company can check 100% of the interventions by detecting defects and send non-conformity information directly to the work or after-sales teams.

Discover more success stories on our website.

TALK ABOUT YOUR PROJECT WITH OUR SALES DIRECTOR, **FILIP MORAEL-PONIATOWSKI**

CONTACT US

deepomatic

END-TO-END SOLUTION

From design to large-scale production, our products and partners support businesses every step of the way, whether it is annotating data, training AIs, or installing and maintaining AI-specific hardware.

OPEN THE BLACK BOX

All of the applications developed, in particular datasets and algorithms, belong entirely to the client.

PRODUTION-READY IN 3 MONTHS

Our easy-to-use software allows operationals as well as data scientists to quickly create best-in-class, production-ready AI applications. Businesses can expect a ROI in less than 3 months by deploying an AI on an industrial scale.

EDGE DEPLOYMENT

We help entreprises deploy and monitor AI-ready edge devices at scale to comply with hardware and security hardware constraints.

THEY TRUST US

AIRBUS
SNCF
OSCARO
Valeo
INDIGO
SANOFI
emovis
an Abertis company
RATP
COMPASS GROUP
BELRON
bouygues TELECOM