PJT IA

Automated quality control

DANNI Youssef

SCUILLER Tristan

HOUKAD Mohamed

SAYARI Firas

REGAIEG Hadil

Encadré par M. CARDIN Ronan

Sommaire

Présentation du projet / Phase d’avant projet

Présentation du projet

Dans les environnements industriels modernes, les opérations de contrôle qualité jouent un rôle crucial pour garantir la conformité des produits finis et éviter que des pièces défectueuses ne soient transmises aux étapes suivantes de la chaîne de production. Ces contrôles permettent également de détecter précocement d’éventuelles dérives de fabrication, facilitant ainsi l’identification rapide de leurs causes profondes.

Une méthode couramment utilisée repose sur l’acquisition d’images des composants via un système de vision, suivie d’une analyse automatique destinée à vérifier certaines caractéristiques visuelles. Lorsqu’une anomalie est suspectée, une inspection humaine complémentaire est généralement sollicitée. Ce processus augmente la charge de travail et expose l’entreprise à des erreurs humaines, parfois coûteuses.

Dans ce contexte, le recours à des modèles de vision par ordinateur permet d’automatiser l’analyse de ces images, offrant une précision souvent supérieure à celle des opérateurs humains. Ces systèmes intelligents sont capables non seulement de classifier les pièces selon des défauts connus, mais également de détecter des anomalies rares ou inattendues, rendant leur intégration particulièrement pertinente dans des contextes industriels exigeants.

Objectif du projet

Ce projet est ouvert par le constructeur automobile VALEO, qui a mis à disposition un dataset comportant des images de composants électroniques prises en fin de ligne dans des conditions normalisées. Avec ces données, il est possible d'entraîner un modèle de Machine Learning permettant de :

- **Identifier** les pièces bonnes et les pièces défectueuses
- **Classifier** les types de défauts le cas échéant
- **Identifier les images anormales** non traitables par le modèle pour qu'un opérateur humain puisse effectuer manuellement le contrôle de la pièce.

Toutefois, bien que le projet soit initialement centré sur l'apprentissage automatique, nous avons jugé pertinent d'étudier l'implémentation concrète d'un tel modèle dans un processus de production. D'abord pour garantir que la solution proposée est réaliste d'un point de vue industriel, mais aussi pour des raisons d'idiosyncrasies au sein de l'équipe projet, tantôt industrielles ou numériques. Par conséquent, l'objectif de ce projet est double :

- **Entraîner** un modèle selon le cahier des charges du challenge de Valeo
- **Proposer un plan d'action** pour implémenter ce modèle dans un processus réel de production

Démarche

L'équipe projet se décompose en 2 sous-groupes :

- Un premier de **data-scientists,** dont la mission principale est d'entraîner le modèle de Machine Learning à partir des données fournies par le constructeur
- Un second d'ingénieurs en **performance industrielle**, qui se chargera d'établir le plan d'action pour le déploiement du modèle au sein de l'appareil de production

La collaboration et la coordination sera nécessaire afin de répondre au double objectif qui est non seulement d'entraîner un modèle, mais également de le déployer.

Calendrier

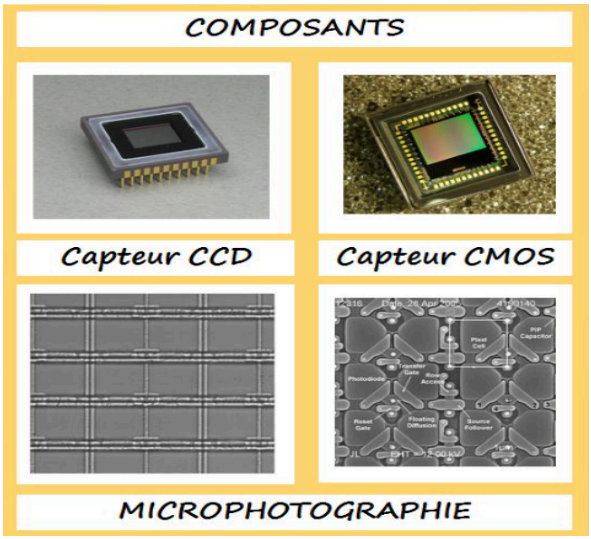
Voici le calendrier que nous nous sommes fixés pour ce projet :

Jalon	Phase	Objectifs
01/03/2025	*Exploration des données*	• Production de figures permettant de comprendre la distribution et la nature des données (on privilégiera les données chiffrées) • Visualiser les données
01/04/2025	*Entraînement d'un MVP*	• Avoir un premier modèle répondant au CDC, avec des performances correctes
01/05/2025	*Finalisation du modèle*	• Disposer d'un modèle déployable avec des performances optimisées
Jalon final	*Rédaction du plan d'action*	• Proposer un plan d'action permettant l'implémentation progressive du modèle dans l'appareil de production.

Phase 1 - Exploration des données

- **Deadline** : 01/03/2025
- **Livrables :**
    - Dossier contenant du code et des figures permettant de mieux comprendre les données mises à disposition afin d'orienter le choix d'une architecture et l'entraînement

Rapport d'exploration



Les pièces inspectées sont des circuits électroniques, plus précisément des composants semi-conducteurs comportant des structures métalliques complexes. Ces composants peuvent présenter plusieurs types de défauts visuels, parfois très subtils.

Les données mises à disposition par le constructeur sont scindées en 2 sous-ensemble :

- **Training dataset :** dataset que le constructeur souhaite qu'on utilise pour l'entraînement, représentant ~85% des données
- **Testing dataset :** dataset que le constructeur souhaite que l'on utilise pour le testing, ~15% des données

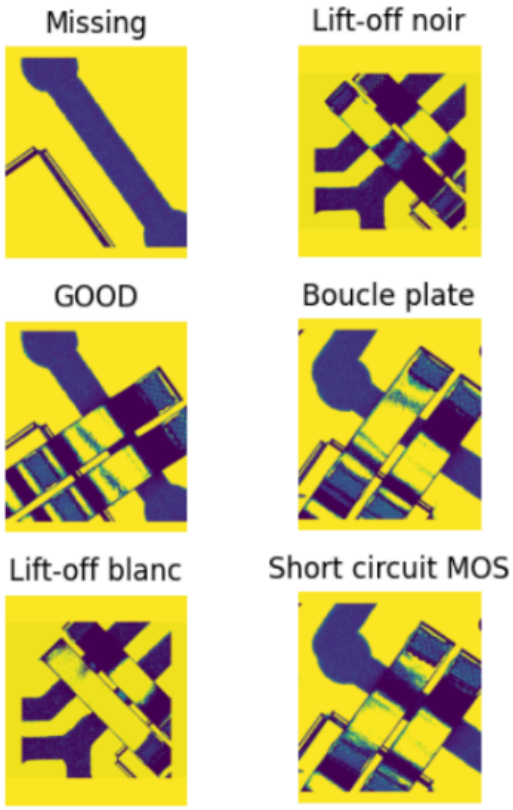
Training dataset :

Le jeu de données d'entraînement mis à disposition comprend 8 278 images, réparties en 6 classes :

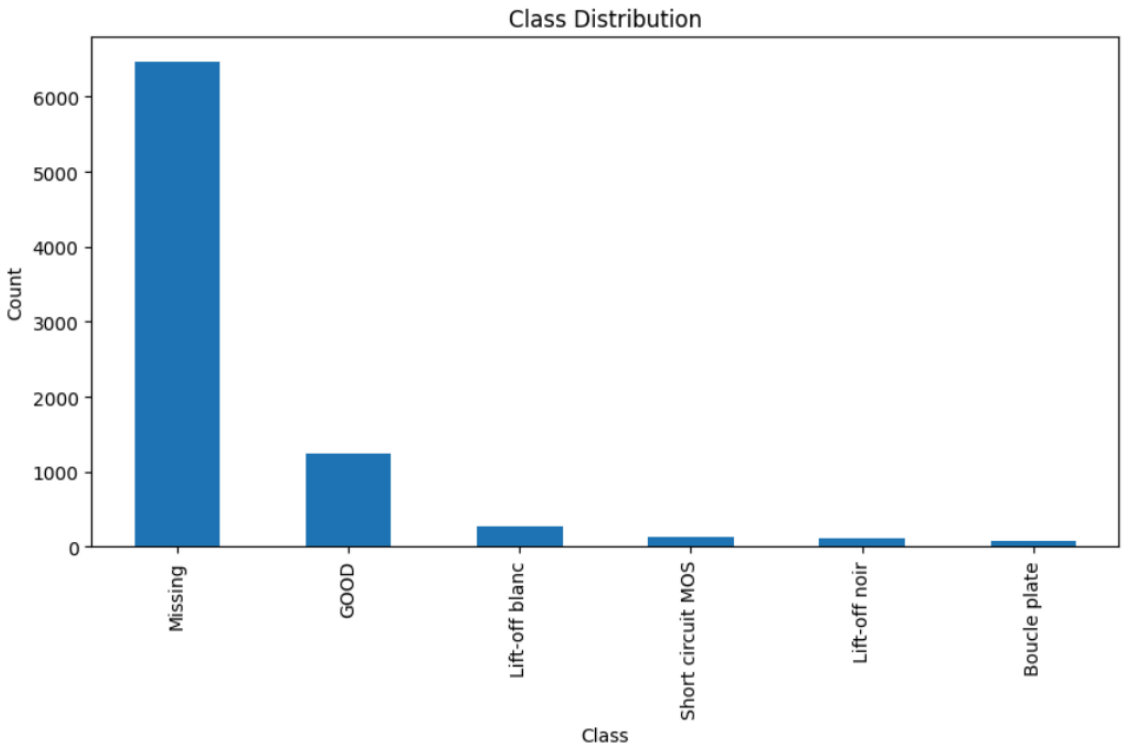
- 1 235 images de pièces conformes (label 0),
- Et 7 043 images présentant un défaut connu (labels 1 à 5).

Classe	Description	Nbr d'images
GOOD	Pièces non defectueuses	**1235**
Flat loop	Ecrasement de boucle	**71**
White lift-off	Décollage de soudure avec reflet clair	**270**
Black lift-off	Décollage sombre	**104**
Missing	Elément absent	**6472**
Short circuit MOS	Court-circuit entre grilles de transistors	**126**

Les images ci-dessus montre des exemples de défauts présents sur les composants électroniques, tels que le Missing, le Lift-off (noir et blanc), le Flat loop, ou encore le Short circuit MOS. Certains défauts sont facilement visibles, comme le Missing, tandis que d'autres sont plus subtils, comme le Flat loop. La classe GOOD représente une pièce sans défaut, servant de référence pour l'inspection automatique.



Les classes présentent un fort déséquilibre. Par exemple, la classe “Missing” représente près de 80% des images.



Testing dataset :

Le jeu de test contient 1 055 images réparties dans les mêmes 6 classes + une classe supplémentaire :

- "Drift" (label 6) : cette classe regroupe les images hors distribution, correspondant soit à des défauts rares, soit à des anomalies liées à l'acquisition (images floues, mal exposées, mal orientées, etc.).

Dans les 2 fichiers csv on a deux indicateurs supplémentaires qui enrichissent les jeu de données.

- lib : Elle identifie l'un des 4 types de composants inspectés, chacun ayant des formes, tailles et orientations différentes.

- window : Elle indique la zone d'inspection ciblée (2 zones par composant).

id	filename	window	lib	Label
0	15b3bab7c186fd35b65df777890c427dd243feacbb85dd...	2003	Die01	Missing
1	1856617e1ac2d821a46a41b938818f0169342226a78f93...	2003	Die01	GOOD
2	19066cce773b3a092ebf4311b11858aa653da6f8274957...	2003	Die01	Missing
3	19c10caf4b24284e1748caed62d94cbb689d6b379b1cf5...	2003	Die01	GOOD
4	1a627426d55a668df8bcd381a7fa87b620481995b6755f...	2003	Die01	Missing
...	...	...	...	...
8273	8c2897ec93e00bd24f68ac7f80152c514f623ef34b7cc4...	2005	Die04	GOOD
8274	8d428498436a83b76aa7e982432fb61e427ee8fb7b30b7...	2005	Die04	Missing
8275	9cd1a03a4a6c5decbf700e64f78e02daa86891ec1577d0...	2005	Die04	Lift-off blanc
8276	e5d19ca1824341c6725858da59f5b856ef3a7a0c3566e2...	2005	Die04	GOOD
8277	e66f1f5b57fa9bf046d4f6088027af10bd139121965e86...	2005	Die04	Missing

Démarche pour la suite du projet

L'entraînement du modèle présente 2 aspects :

- Problème de **multi-class classification** pour la classification des pièces entre la classe bonne, défaut 1, défaut 2...
- **Détection des outliers**, autrement dit des images non traitables par le modèle et rangement dans une classe "drift".

L'exploration des données a permis de mettre en lumière :

- la nécessité d'un **preprocessing** sur les données avant l'envoi dans un modèle d'apprentissage automatique
- une forte **disparité des classes** dans les données d'entraînement
- une **taille importante des données** images (~1Go au total) donc la nécessité :
    - d'utiliser une plateforme de **cloud computing** à défaut de disposer d'un GPU en hardware
    - de **batchifier l'entraînement** pour ne pas saturer la mémoire RAM

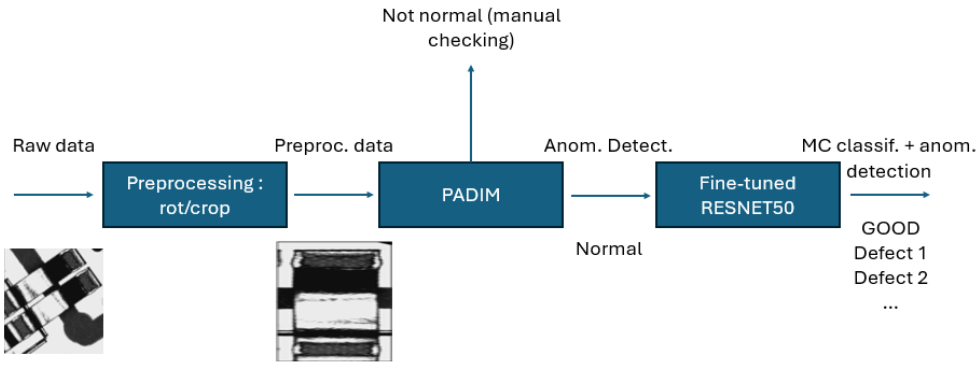
Les objectifs de la phase 1 sont remplis, le projet peut désormais passer à la phase 2.

Phase 2 - Entraînement d'un MVP

- **Deadline :** 15/03/2025
- **Livrables :**
    - Fichier contenant le modèle et ses paramètres permettant de réaliser le problème de multi-class classification.

Choix d'architecture

L'architecture benchmark pour ce genre de problème de classification est la suivante :



On a d'abord la donnée burte issue de la caméra qui arrive en input du modèle complet. Cette donnée est dans un premier temps preprocessed pour qu'elle soit interprétable et significative pour un modèle de computer vision. Vient ensuite et dans un premier temps le modèle PADIM dont le but est d'identifier les outliers pour éviter de produire des prédictions sur des images non interprétables, puis vient ensuite un RESNET50 avec les poids du dataset ImageNet qu'on a fine-tuné en ajoutant des couches superficielles pour :

- **Adapter** les facultés de vision du modèle à notre dataset
- **Mapper** cette représentation sur les 5 classes d'intérêt (GOOD, Missing...)

De cette manière, on dispose d'un modèle qui :

- Permet d'**identifer et de classifier les différents types de défauts**

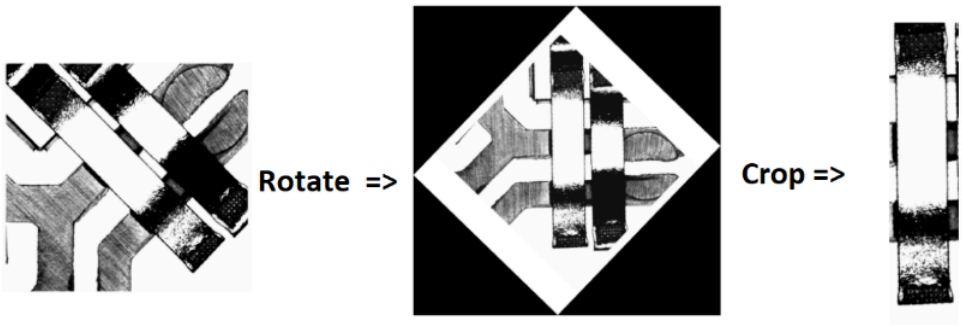
- Fait cela de manière **robuste**, c'est-à-dire qu'il écarte les images qui ne sont pas interprétables.

On rappelle par ailleurs que l'objectif de ce modèle est l'implémentation dans un processus de contrôle qualité. Par conséquent, il devra être suffisamment sensible aux défauts des pièces pour ne pas laisser passer de pièces non-conformes, sans pour autant détecter des défauts quand il n'y en a pas, sans quoi cela fait augmenter le volume de pièces à contrôler manuellement voire cela augmente le taux de rebut si jamais l'entreprise ne met pas d'opérateur humain pour contrôler les pièces détectées non-conformes par le modèle.

Preprocessing

Le prétraitement des images constitue une étape essentielle dans tout pipeline de traitement d'image, en particulier dans le contexte de l'apprentissage automatique ou de la vision par ordinateur. Cette étape vise à améliorer la qualité des données en vue d'optimiser les performances du modèle.

Les principales opérations de prétraitement utilisées dans ce projet sont les suivantes :



Correction de l'orientation (Rotation) :

Certaines images capturées peuvent présenter des orientations variables selon le composant représenté ou le dispositif de capture. Afin d'uniformiser leur position, une rotation est appliquée à chaque image selon un angle spécifique défini pour chaque composant. Cette rotation est réalisée avec l'option expand=True de la bibliothèque PIL, qui permet d'adapter dynamiquement la taille de l'image résultante afin d'éviter toute perte de contenu après transformation.

Par exemple, une image associée au composant "Die01" est tournée de 55° dans le sens antihoraire, tandis que celle de "Die02" est tournée de -44°, etc. Ces valeurs sont stockées dans

un dictionnaire dédié, ce qui permet une application systématique et personnalisée selon le type de composant.

```
rot_crop_data = {
        "Die01":[55,(340, 120, 500, 680)], # (left, upper, right, lower)
        "Die02":[-44, (480, 210, 640, 930)],
        "Die03":[134, (460, 200, 620, 920)],
        "Die04":[35, (310, 130, 470, 690)]
}
```

Recadrage (Cropping) de la zone d'intérêt :

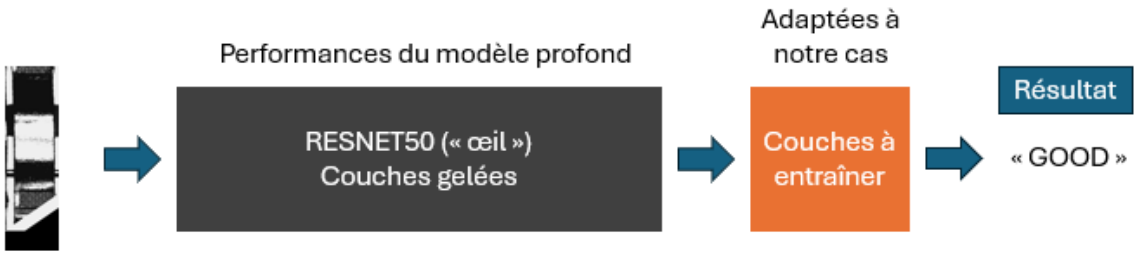
Après la rotation, l'image est recadrée pour isoler uniquement la zone pertinente, c'est-à-dire la région contenant les informations utiles à l'analyse (le die, composant, etc.). Le recadrage est défini à l'aide d'un rectangle de sélection (left, upper, right, lower) en pixels, également spécifique à chaque composant. Ce choix précis permet de retirer les marges inutiles introduites lors de la rotation et d'optimiser la qualité des données exploitées.

Du coup La fonction rotate_and_crop_image() gère ces deux étapes de manière automatisée :

- Elle charge l'image à partir de son chemin,
- Applique une rotation avec un angle donné,
- Puis effectue un recadrage selon un rectangle défini,
- Et enfin sauvegarde l'image traitée au format PNG.

Multi-class classification

Le problème qu'on nous demande de résoudre est un problème classique de classification multi-classe d'images en deep learning. La solution technologique de référence pour résoudre ce genre de problème est d'utiliser une technique qu'on appelle Transfer Learning. En bref, il s'agit d'utiliser les performances d'un modèle préentraîné sur une tâche précise mais générale, de capter sa sortie et de réentraîner un plus petit modèle par dessus ce modèle fixe. Pour les problèmes de classification d'image, le "gros modèle" va jouer le rôle de "l'oeil" si l'on veut se représenter la situation avec notre point de vue d'être humain. Un oeil est complexe, mais si on accès à ces informations, alors il suffit de construire les connextions neuronales pour associer le signal qu'il nous renvoie à une certaine classe (cette fille est blonde, cette fille est brune...). Ainsi, on peut obtenir des performances très élevées avec un minimum de calculs. Voici un schéma récapitulatif de la technique de Transfer Learning :



Le modèle qui est notre "oeil" est appelé backbone (colonne vertébrale). Cette backbone reste fixe, et on entraîne seulement les couches dites superficielles. Nous employons depuis tout à l'heure le terme de couches car on se situe ici, si ce n'est pas déjà clair, dans un cadre de Deep Learning, autrement dit les modèles de type réseaux de neurones (qui sont simplement des combinaisons linéaires de fonctions composées dites d'activation telles que max(0, x) encore appelée RELU). On parle de modèle "profond" (origine du terme "deep" dans Deep Learning) quand le nombre de couches de neurones est important : c'est souvent corrélé avec des meilleures performances du modèle car ce dernier peut capter des motifs plus complexes dans les données et les retranscrire dans ses prédictions, moyennant qu'il n'y a pas d'overfitting ou "surentraînement" en français, autrement dit que le modèle ne se contente pas d'apprendre bêtement un grand nombre de cas particuliers, mais qu'il soit capable d'inférer des prédictions générales à partir d'un nombre raisonnable de cas auxquels il aura été exposé durant son entraînement. Bien sûr tout ce qui est écrit ici est une vulgarisation de notre travail, on pourra consulter les notebooks d'entraînement pour l'implémentation mathématique concrête de ces

derniers (repose essentiellement sur l'algèbre linéaire sur les tenseurs, origine du nom de la librairie TensorFlow, développée par Google, qui a été utilisée pour notre projet, et qui est actuellement la référence industrielle pour la conception de modèles de Deep Learning, même si son lointain cousin PyTorch (développée par Meta AI) prend de plus en plus de place en raison de sa flexibilité et de son omniprésence dans le monde académique).

Un mot tout de même sur l'implémentation pratique car cela a été un challenge durant cette phase : l'utilisation de TensorFlow implique une accéélération matérielle possible moyennant l'utilisation d'un GPU du constructeur américain NVIDIA, qui a développé un firmware nommé CUDA qui facilite énormément l'exploitation de la puissance de parallélisation des calculs des GPU, mais qui réseve ce firmware à ses propres composants hardware (donc pas pour les GPU AMD qui sont le principal concurrent pour les GPU en B2C). Or aucun membre de notre équipe PJT ne possédait de GPU NVIDIA. Par conséquent, il fallait trouver une manière d'accéder à un tel GPU sans quoi l'entraînement n'était pas réalisable (surtout pour des modèles de cette taille, qui bien qu'on utilise le transfer learning pour alléger le volume de paramètres d'entraînement, demandent de traiter des données de type image qui sont extrêmement volumineuses par rapport à d'autres types de données comme le texte ou l'audio...). La solution que nous avons retenue est le cloud computing. En un mot, il s'agit de faire tourner notre script d'entraînement sur un serveur qui possède des GPU NVIDIA, et de récupérer seulement les résultats de l'entraînement sous la forme d'un fichier .h5 de modèle. On pourrait penser qu'il faut être un client de la taille d'une entreprise pour accéder à ce genre de service, mais pas du tout. L'ère AWS est révolue (même si AWS est toujours la référence en B2B et parfois même en B2C), et l'on peut utiliser d'autres plateformes pour entraîner nos modèles, dont certaines sont très connues (et pour lesquelles on ne se doute pas forcément qu'il s'agit de plateformes de Cloud Computing). Les deux principales sont :

- Kaggle
- Google Colab

En raison de notre familiarité avec les produits de Google, notamment de Google Drive pour le stockage des données, nous avons opté pour la 2e option. De cette manière toute l'équipe data avait accès à la même donnée (pas de duplicatas, ce qui est une bonne chose pour l'impact écologique), et aux mêmes scirpts, car Google Colab a ceci de génial qu'il est aussi une plateforme de collaboration sur des jupyter notebooks.

PADIM

Pour rappel, un des critères de validation du modèle était la bonne détection des images non traitables (problème sur la caméra, mauvaise pièce sur la ligne...). Comment peut on réaliser ce genre de détection ? Une technique récente et spécialement conçue pour les applications industrielles comme pour notre cas se nomme PaDiM. C'est le diminutif pour Patch Distribution Modelling.

Les problèmes de détection d'outlier sont classiques en machine learning. On a par exemple la détection de fraudes bancaires qui est un cas d'étude extrêmement fréquent pour toute personne se formant au machine learning. Il existe des modèles dédiés que nous ne détaillerons pas ici, mais qui permettent de traiter avec un minimum de lignes de code et en un minimum de temps de calcul ce genre de problèmes.

Toutefois, ces derniers ne fonctionne pas bien dans des données multidimensionnelles, la raison principale : un phénomène qu'on appelle la curse of dimensionality. C'est la traduction concrête du fait qu'il y a infiniment plus de points dans un plan que dans une ligne, et infiniment plus de points dans un volume que dans un plan. Cela se traduit concrêtement par une augmentation exponentielle du volume de données nécessaires à l'entraînement des modèles avec la dimension d.

$$Volume\ de\ données \sim e^{d}$$

Par conséquent, quand on travaille sur des données fortement multidimensionnelles telles que des images (224x224 = 50176 dimensions numériques), il n'est pas réaliste d'utiliser les modèles traditionnels lorsque le volume de notre dataset est "seulement" de 10000 images.

Le PaDiM a pour objectif de lever cette difficulté. Pour cela, il se sert de techniques issues de l'IA générative, en particulier de l'IA générative d'images (exemples de modèles : MidJourney, DALL-E d'OpenAI...). En effet, ces modèles arrivent à créer des images, ce qui signifie que durant leur entraînement, ils ont réussi à "apprendre" les patterns de ces dernières malgré la dimension élevée des données. Autrement il s'agit de modèles qui sont capables de très bien inférer une règle générale à partir de cas particuliers.

Et cette capacité d'inférence, ils la doivent à des hypothèses plus fortes sur la structures des données images. Là où avant on supposait que même après transformations dans les couches du réseau de neurone, la distribution des données de notre dataset avait une forme quelconque, on suppose désormais une distribution gaussienne. C'est une hypothtèse extrêmement forte, qui est à la base des modèles nommés Variational Auto-Encoders, qui ont été un certain temps la

référence en terme d'IA générative d'images, avant que les technique de Stable Diffusion ne viennent leur voler la vedette.

Le PaDiM, c'est l'application directe de cette hypothèse de gaussiannité à nos données industrielles. On suppose que pour une classe de défauts données (qu'on peut connaître durant notre entraînement puisqu'on est ici dans un cadre d'apprentissage supervisé), on a une distribution gaussienne des donnéees dans les couches centrales du réseau "oeil". En effet, ce réseau (le RESNET50) possèdre l'architecture suivante :

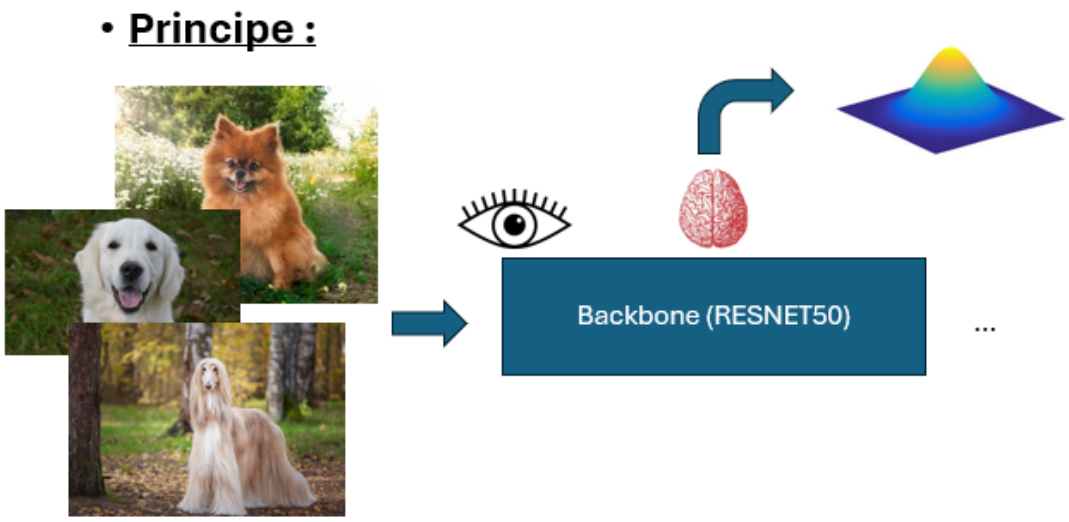
- Première phase de convergence, avec des couches convolutives, dont l'objectif est de synthétiser l'information présente dans l'image en réduisant la dimension totale du tenseur de données
- 2e phase de divergent avec des couches feed forward dnt l'objectif est d'orthogonaliser les features des images.

Par conséquent, si l'on cherche à :

- minimiser la taille du modèle PaDiM
- travailler sur les données les plus abstraites

On extrait l'output des couches entre la fin de la phase de convergence et le début de la phase de divergence. C'est dans cette région que l'hypothèse de données gaussiennes est la plus pertinente.

Voici un schéman récapitulatif :



En outre, un modèle PaDiM c'est une distribution gaussienne multivariée inférée à partir des données. Cette dernière est donc paramétrée par :

- une matrice de covariance : taille N^2
- un vecteur espérance ou barycentre : taille N

Par conséquent, le nombre de paramètres du modèle est de N(N+1), avec N le nombre de neurones de la couche intermédaire de laquelle on extrait les données.

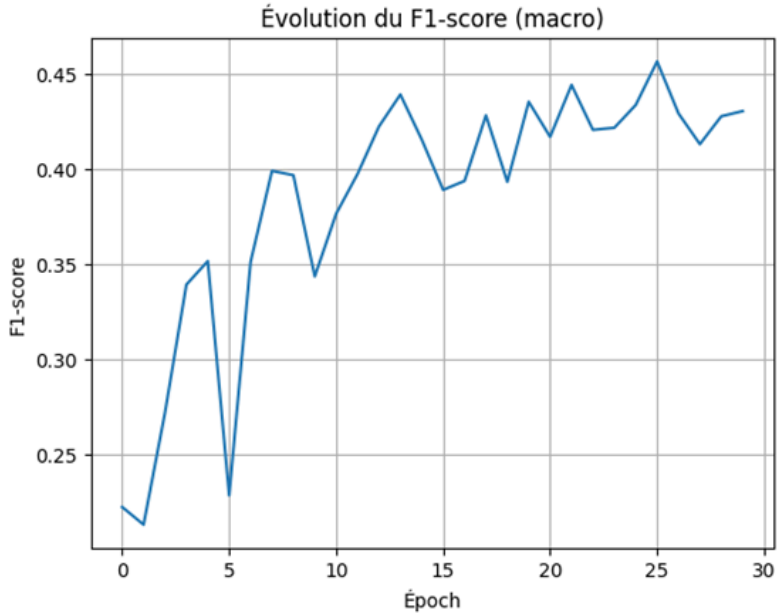
On notera ici qu'on sort a priori du cadre deep learning, on est davantage dans un cadre statistique avec de l'optimsiation type MLE. Par conséquent, on ne peut plus bénéfier cu framework de TensorFlow pour la conception du modèle, on est obligé de le construire "from scratch", via des techniques de programmation orientée objet. Enfin, il est probablement possible d'utiliser l'API d'une librairie comme scikit-learn pour créer un modèle au sens général du terme et ainsi de bénéficier par le jeu de l'héritage de classes des fonctions utiles pour le test, la validation, le tracé de performances, mais cela impose un certain nombre de contraintes au niveau de la syntaxe et de la nature des données que nous avons jugé superflu, surtout pour la conception du MVP où notre objectif était d'aller "straight to the point" comme disent nos collègues américains. Parlons maintenant de l'entraînement du modèle (méthode fit) qui consiste en l'ajustement de la matrice de covariance et du barycentre de la distribution.

Pour le barycentre, on peut utiliser l'estimateur classique de Stein, qu'on pourra éventuellement biaiser au besoin (technique de Shrinkage pour améliorer la vitesse de convergence).

Pour la matrice de covariance, on utilise souvent une technique de shrinkage comme celle de Ledoit Wolf (issue de la finance quantititative : l'objectif était d'estimer les matrices de covariance associée à la volatilité et covolatilité des actions composant les portefeuilles des organismes bancaires et des fonds d'investissement : la technique a ensuite été étendue à toutes les statistiques, raison pour laquelle on la retrouve dans les bibliothèques de Machine Learning telles que scikit-learn, qui est d'ailleurs l'implémentation que nous avons utilisé pour notre projet). La raison la même que celle citée pour le barycentre : c'est une application classique du trade-off biais-variance.

Bilan des performances du modèle et décisions pour la suite du projet

Voici sans plus attendre les performances de notre modèle. Déjà durant l'entraînement, voici l'évolution de F1-score sur le split de validation :



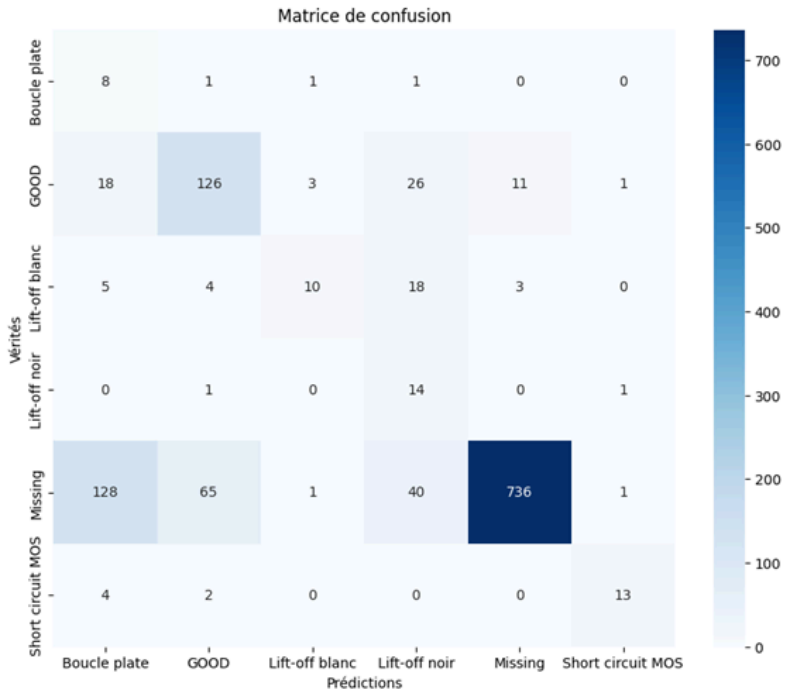
On observe une amélioration progressive des performances, ce qui confirme que le modèle apprend bien. Petite précision sur ce qu'est le score F1 :

$$F1\ score\ =\ Harm(Précision,\ Rappel)$$

Le F1 score est la moyenne harmonique entre la précision et le rappel. Par conséquent, il s'agi t d'un score plutôt punitif (moyenne harmonique) donc très adapté à un contexte zéro défaut, qui prend à la fois en compte la précision qui récompense les bonnes prédictions et le rappel qui pénalise les faux positifs. Il s'agit donc de la métrique idéale pour un problème de classification multi-classe (Attention toutefois, ce n'est pas notre loss function, qui est l'entropie croisée adaptée à un encoding de type one hot).

	precision	recall	f1-score	support
Boucle plate	0.05	0.73	0.09	11
GOOD	0.63	0.68	0.66	185
Lift-off blanc	0.67	0.25	0.36	40
Lift-off noir	0.14	0.88	0.24	16
Missing	0.98	0.76	0.86	971
Short circuit MOS	0.81	0.68	0.74	19
accuracy			0.73	1242
macro avg	0.55	0.66	0.49	1242
weighted avg	0.90	0.73	0.79	1242

Une fois l'entraînement terminé, on peut afficher les métriques de performance du modèle. Ici, on s'intéresse au F1-score macro, qui est de 49%. Cela signifie que le modèle apprend effectivement des patterns, mais qu'il y a un grand nombre de faux positifs, c'est en particulier montré par la matrice de confusion :



Par conséquent, on va chercher à améliorer les performances de ce modèle. Pour cela, différentes stratégies :

- la première est d'augmenter la profondeur du réseau superficiel (nombre de couches)

- la seconde d’augmenter sa largeur (nombre de neurones par couches)

Dans tous les cas il faut augmenter la taille du modèle. La solution que nous allons proposer ensuite est une augmentation en largeur, en particulier en ne compressant plus les données de la dernière couche RESNET50 avec un couche de pooling, mais en utilisant une couche de flattenning qui préserve la dimension des données.

Phase 3 - amélioration des performances

- **Deadline :** 01/04/2025
- **Livrables :**
    - Modèles PADIM et classifier optimisés (F1-score > 90%)

Choix de conception

Comme dit précédemment, le changement d'architecture principal entre le nouveau modèle et l'ancien est une augmentation de la largeur de ce dernier. En particulier, cela se fait en remblaçant une couche d'average pooling 2D par une couche de flattenning pour supprimer complètement la perte de dimensionnalité. Nous ne nous sommes pas tournés directement vers l'option flattenning car cette dernière est souvent peu adaptée au travail avec des données très fortement multi-dimensionnelles telles que les images, car on vient mapper sur un vecteur toutes les composantes de tenseurs de dimensions 3 ou 4, par conséquent le nombre de paramètres de ce modèle est exponentiel, et le modèle devient rapidement ingérable computationnellement.

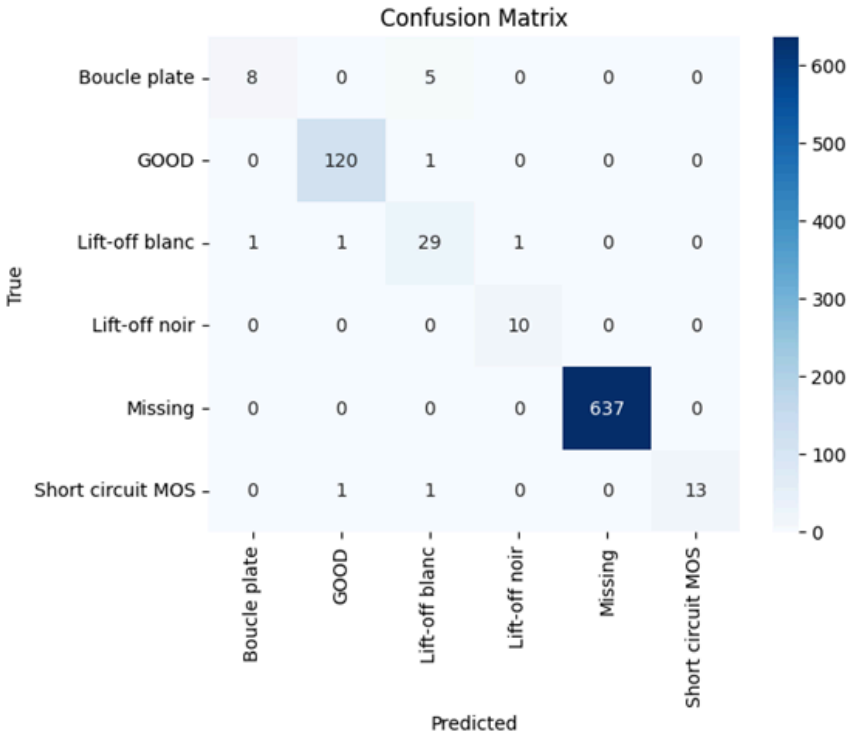
Toutefois, ici, nous pouvons nous permettre, au regard des capacités matérielles permises par le cloud computing, d'utiliser cette approche car la mémoire du GPU est suffisante pour paralléliser l'optimisation de toutes ces couches, sans impact significatif sur le temps d'entraînement.

Implémentation et évaluation des nouvelles performances

Voici les nouvelles performances que nous obtenons grâce à ce modèle.

	precision	recall	f1-score	support
Boucle plate	0.89	0.62	0.73	13
GOOD	0.98	0.99	0.99	121
Lift-off blanc	0.81	0.91	0.85	32
Lift-off noir	0.91	1.00	0.95	10
Missing	1.00	1.00	1.00	637
Short circuit MOS	1.00	0.87	0.93	15
accuracy			0.99	828
macro avg	0.93	0.90	0.91	828
weighted avg	0.99	0.99	0.99	828

Déjà on peut constater que le score F1 est désormais de 91%, c'est 2 fois mieux que pour le précédent modèle. Par ailleurs, on peut analyser les faux positifs au moyen de la matrice de confusion :



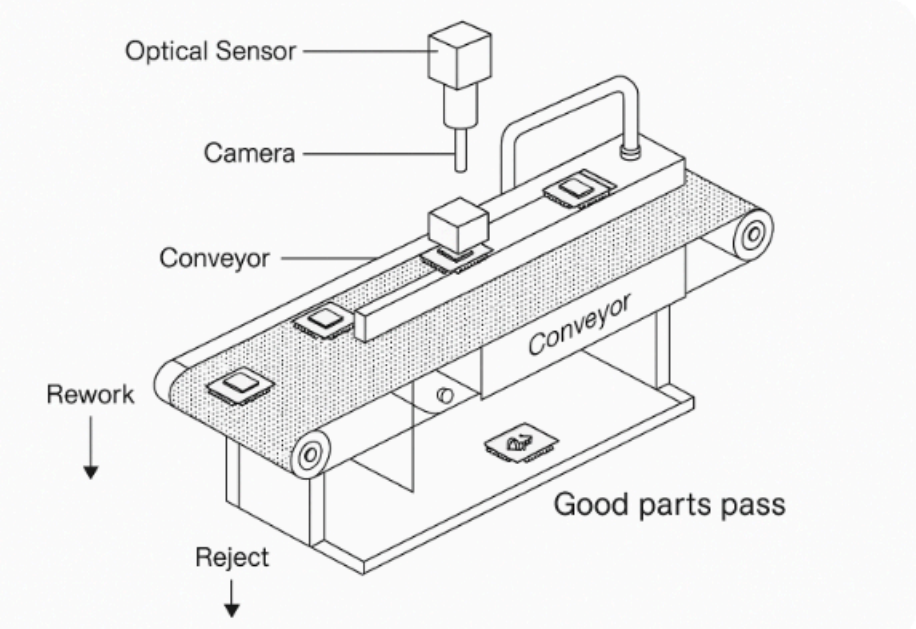
Les objectifs de performance sont atteints pour le modèle. On peut passer à la phase d'industrialisation.

Phase 4 - industrialisation

Intégration dans la chaîne de production

Objectif : Intégrer notre solution IA directement dans la chaîne de production de VALEO, en fin de ligne, afin d'inspecter automatiquement chaque microcontrôleur pour garantir leur conformité sans intervention humaine.

l'image présente un exemple concret d'intégration d'un système de vision dans une ligne de production. On y observe clairement l'emplacement optimal de la caméra au-dessus du convoyeur ainsi que la fixation rigoureuse des pièces, permettant à l'intelligence artificielle d'analyser chaque composant et de prendre une décision fiable quant à sa conformité (bon, à retoucher ou à rejeter).



Dans cette optique, nous avons identifié trois leviers essentiels à la mise en œuvre efficace de l'inspection par vision assistée :

1. Emplacement optimal des caméras dans la chaîne de production

L'intégration des caméras industrielles haute résolution doit se faire à la sortie immédiate du convoyeur, juste avant l'emballage final. Cette position garantit que chaque microcontrôleur inspecté est définitivement terminé, sans risque de modification après analyse. Les caméras sont montées à la verticale au-dessus de la pièce, sur un bras fixe équipé d'un système anti-vibration pour garantir la netteté de chaque cliché.

Élément	Description technique
Type de caméra	Caméra industrielle CMOS, 5MP, 60fps
Distance focale	25 cm, fixe, adaptée à la taille du micro-chip
Champ de vision	1 composant par image (pas de chevauchement)
Support	Bras articulé fixé au châssis du convoyeur

Cette configuration garantit une capture d'image fiable, stable et répétable, condition essentielle pour un bon fonctionnement du modèle IA.

2. Éclairage maîtrisé avec LED annulaire pour homogénéité visuelle

L'éclairage dans l'environnement de prise d'image est essentiel pour éviter les reflets ou ombres qui pourraient gêner la détection des défauts. L'implantation d'un système LED annulaire à lumière froide autour de l'objectif permet de produire une lumière uniforme, orientée, sans ombre portée, avec intensité contrôlable.

Composant	Détail
Type de LED	LED blanche à température de couleur 5000K
Contrôle	Variateur automatique selon la lumière ambiante

Objectif	Éviter la surexposition et renforcer les contrastes
Position	Autour de la lentille de la caméra, à 90°

Combiné à la bonne position de la caméra, ce système d'éclairage améliore significativement la fiabilité de la détection de défauts.

3. Gabarit de positionnement pour le cadrage constant des pièces

Pour garantir une analyse cohérente, chaque pièce doit être positionnée exactement de la même manière sous la caméra. On propose un système de gabarit mécanique (type emboîtement) sur le tapis roulant. Ce système, combiné à un capteur de présence déclenchant la prise de photo au bon moment, permet un alignement automatique.

Élément	Solution proposée
Gambari	Plateau usiné aux dimensions des pièces
Système de centrage	Ergots latéraux + butée arrière
Capteur	Capteur optique déclenchant la capture
Logiciel	Rotation + recadrage automatique via script

Ce dispositif garantit la reproductibilité du cadrage, facilitant le traitement des images par le modèle IA et réduisant les erreurs d'interprétation.

Pour conclure alors En combinant un positionnement optimal de la caméra, un éclairage maîtrisé et un cadrage précis via gabarit, nous assurons une acquisition d'images standardisées, indispensable à la fiabilité du modèle d'intelligence artificielle. Cette base solide permet un déploiement industriel robuste et scalable dans l'usine VALEO.

Structure physique du système de tri

L'objectif principal de notre système est de trier automatiquement les microcontrôleurs vers l'un des cinq compartiments possibles : bon, défaut 1, défaut 2, défaut 3, ou défaut inconnu. Ce tri repose entièrement sur la décision prise par notre modèle d'intelligence artificielle, après analyse visuelle de chaque composant. La solution repose sur une chaîne de traitement automatisée, combinant vision industrielle, IA, et actionneurs physiques.

Le processus débute par une caméra industrielle, placée en amont du convoyeur. Cette caméra capture une image haute résolution de chaque pièce en mouvement. Elle est positionnée de

manière fixe et dispose d'un éclairage contrôlé pour assurer une inspection fiable quelles que soient les conditions d'éclairage ambiantes. Les images sont ensuite transmises à notre système d'analyse IA pour classification.

Les microcontrôleurs sont transportés le long d'un **convoyeur à bande plane**, conçu pour assurer un déplacement fluide et stable des composants. Ce convoyeur est motorisé et fonctionne à vitesse constante, ce qui permet de synchroniser avec précision les actions des bras pousseurs plus loin sur la ligne.

Une fois la pièce classifiée par l'IA, elle continue sa trajectoire sur le tapis jusqu'à atteindre la **zone de tri physique**. Cette zone est équipée de **bras pousseurs latéraux**, disposés en série. Chaque bras est associé à un compartiment de tri (bon, défaut 1, etc.). Lorsque la pièce arrive devant le bras qui correspond à sa classe, ce dernier s'active brièvement pour dévier la pièce latéralement dans le bac de réception correspondant. Ce mécanisme permet un tri rapide et sans interruption du flux.

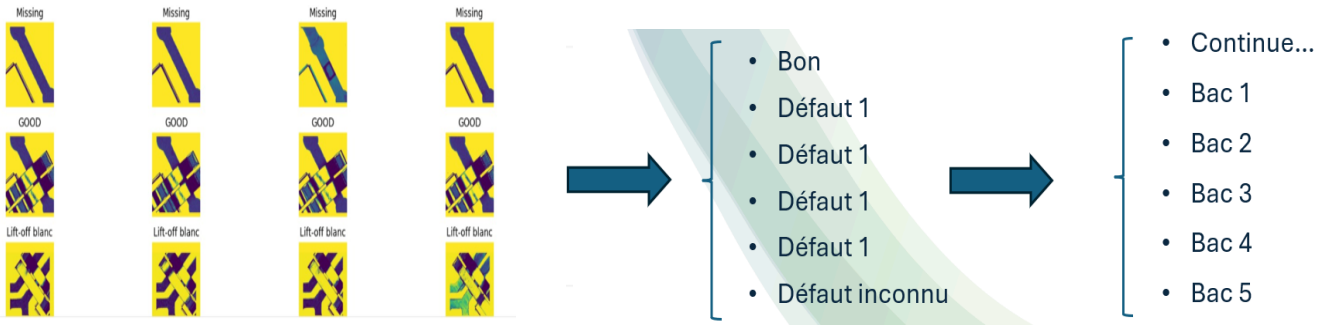
L'ensemble du système est orchestré par un **automate programmable industriel (API)**, en l'occurrence un **Siemens S7-1200**. Cet automate reçoit la classe prédite via une **API Python** (voir slide suivante) et détermine précisément le **moment d'activation du bras** en fonction de la vitesse du convoyeur et de la position de la pièce.

Enfin, les **bacs de tri** sont disposés latéralement à la ligne, en face de chaque bras. Chaque bac est conçu pour recevoir uniquement les pièces d'une classe donnée. Pour garantir la sécurité des composants électroniques, ces bacs sont fabriqués en **plastique antistatique** (ESD), afin de protéger les microcontrôleurs contre les décharges électrostatiques lors de leur réception.

Communication entre IA, API Python et Automate

L'objectif de cette étape est d'assurer une **liaison fluide et fiable entre le modèle d'intelligence artificielle et les actionneurs physiques** du système, notamment les bras pousseurs chargés du tri. Pour cela, nous avons mis en place une architecture logicielle simple et modulaire, qui repose sur une communication entre l'IA, une API Python intermédiaire, et un automate programmable industriel (API).

Le processus commence avec notre **modèle d'IA**, déployé sur un serveur Python. Ce modèle reçoit les images capturées par la caméra industrielle, les analyse et **prédit la classe** de chaque microcontrôleur : bon, défaut 1, défaut 2, défaut 3 ou défaut inconnu. Cette prédiction prend la forme d'un message structuré (ex. JSON) contenant la classe attribuée à la pièce.



Une **API Python REST**, développée avec Flask, assure ensuite le relais entre l'IA et l'automate. Cette API reçoit la prédiction depuis le modèle et la **transmet à l'automate industriel** (Siemens S7-1200) via le protocole Modbus TCP. Ce découplage entre les modules IA et automate permet une meilleure organisation du code, une maintenance facilitée et la possibilité de modifier indépendamment les modules.

L'**automate programmable** reçoit donc l'information de classe associée à chaque pièce. En tenant compte de la **vitesse du convoyeur**, il calcule le **moment précis où la pièce arrivera** devant le bras pousseur correspondant à sa classe. Il synchronise alors l'activation du moteur du bras pour pousser la pièce dans le compartiment prévu.

Enfin, le **bras pousseur** s'active brièvement pour **dévier la pièce latéralement** dans le bac de tri associé. Cette action se déroule en temps réel, de manière fluide, sans interrompre le flux des pièces sur le convoyeur.

Ce système de communication IA ➜ API Python ➜ automate ➜ actionneur permet une **intégration fluide du logiciel et du matériel**, garantissant un tri automatisé rapide, précis et adaptable.

Technologies et composants utilisés

Dans le cadre de notre système de tri automatisé basé sur l'intelligence artificielle, plusieurs composants matériels et logiciels ont été sélectionnés pour leur fiabilité, leur intégration industrielle et leur compatibilité avec des architectures modulaires. Ci-dessous, nous présentons les équipements principaux utilisés dans la solution, accompagnés de références concrètes.

Automate programmable

Le contrôle de l'ensemble des actionneurs (notamment les bras pousseurs) est assuré par un **automate programmable industriel (API)** de type **Siemens S7-1200**. Ce contrôleur est largement utilisé dans l'industrie pour sa robustesse, sa flexibilité et sa compatibilité avec des protocoles standards comme **Modbus TCP** ou **Profinet**. Il reçoit les instructions de tri via l'API Python, puis synchronise les actions physiques en fonction de la vitesse du convoyeur.

Bacs de tri

Les pièces triées sont récupérées dans des **bacs en plastique antistatique**, pour éviter tout risque de décharge électrostatique pouvant endommager les microcontrôleurs. Nous recommandons l'utilisation de modèles fournis par **RS Pro** ou **Conrad**, qui proposent des formats industriels empilables et adaptés à la logistique de composants électroniques.

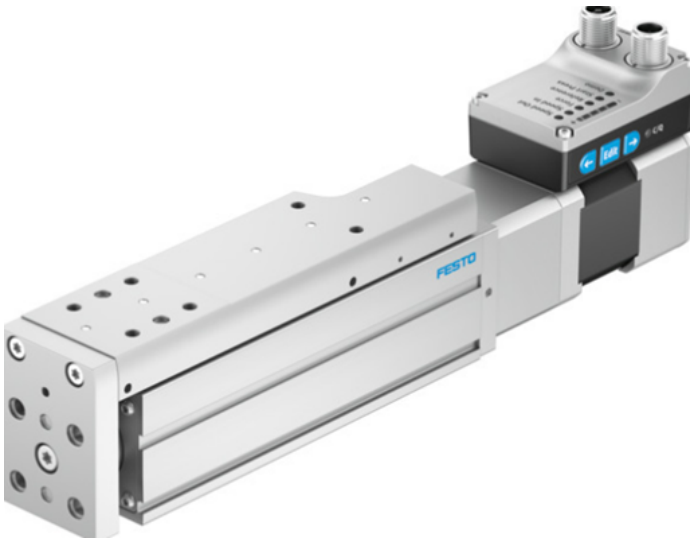
Convoyeur

Le transport des microcontrôleurs est assuré par un **convoyeur à bande plane**, permettant une stabilité optimale lors de la capture d'image et du tri. Nous avons identifié deux fournisseurs compatibles avec notre architecture : **MSET** et **Interroll**. Le choix s'est porté sur des convoyeurs à bande lisse, motorisés, avec des guides latéraux en option pour éviter tout glissement latéral des pièces.

Bras pousseurs

Le mécanisme de tri physique repose sur des **bras pousseurs latéraux motorisés**, capables de dévier les pièces vers les bacs de tri. Pour cette tâche, nous recommandons l'utilisation de bras **électriques linéaires** tels que les **Festo DGEA**, les modèles **SMC** ou encore les actionneurs compacts **LinMot**. Ces équipements offrent une bonne précision, une intégration facile avec les

automates Siemens, et une maintenance simple. Chaque bras est associé à un bac, et sa



commande est synchronisée via le S7-1200.

Objectifs de l'industrialisation

L'objectif de notre projet ne se limitait pas au développement d'un modèle performant sur des données annotées. L'enjeu était d'intégrer ce modèle dans un environnement de production réel, en tenant compte des contraintes industrielles et opérationnelles. Trois axes principaux ont guidé notre démarche :

- **Fiabiliser l'analyse IA en conditions réelles**, en nous assurant que le modèle reste performant même en dehors de son environnement d'entraînement.
- **Mettre en place une traçabilité numérique complète**, assurant un suivi rigoureux de chaque pièce inspectée, avec une conservation fiable des résultats d'inspection.
- **Superviser en temps réel les performances du modèle**, pour anticiper les dérives et organiser une amélioration continue grâce à des outils de pilotage automatisés.

Ces trois axes ont permis d'orienter la phase d'industrialisation vers des solutions concrètes, efficaces et directement applicables sur une ligne de production.

Mise en place de la traçabilité

La traçabilité est un enjeu majeur dans tout système industriel. Elle permet non seulement de suivre chaque produit, mais également de relier les résultats de contrôle qualité à une entité

physique précise. Dans notre projet, chaque pièce entrant sur la ligne de contrôle est dotée d'un identifiant unique sous forme de QR code. Ce code est scanné automatiquement avant que la pièce soit analysée par le modèle IA.

Cette étape garantit une liaison directe entre les données d'inspection (image, prédiction, type de défaut) et la pièce concernée. Cela est essentiel pour pouvoir réagir rapidement en cas de non-conformité ou de réclamation client.

Les résultats sont enregistrés dans une **base de données centralisée** comprenant :

- L'image d'inspection (brute ou annotée)
- La prédiction du modèle (label, score de confiance, métriques associées)
- Le type de défaut détecté (s'il y a lieu)
- L'identifiant de la pièce (QR code), l'horodatage, la version du modèle utilisé

Grâce à cette architecture, les opérateurs qualité et les responsables production peuvent retrouver l'historique complet d'une pièce en quelques secondes. Cette base peut aussi être exploitée pour des analyses statistiques avancées ou des rapports d'audit.

Supervision continue via tableau de bord

Dans une logique de production industrielle, il est essentiel de pouvoir surveiller les performances d'un système en temps réel. Nous avons donc développé un **dashboard qualité** accessible et automatisé, qui synthétise plusieurs indicateurs de performance.

Indicateur	Utilité principale
Taux de défauts	Suivi global de la qualité
Types de défauts	Identification des défauts fréquents
Score F1 / Précision	Évaluation de la fiabilité du modèle
Taux de drift (PADIM)	Détection des dérives
Validation manuelle	Suivi des cas à double vérification

Ce tableau de bord est mis à jour automatiquement à partir de la base de données d'inspection. Il permet aux équipes de production d'avoir une vue synthétique du niveau de conformité global, tout en alertant sur les éventuelles anomalies ou dérives du modèle IA.

Il constitue un véritable outil de pilotage pour assurer la stabilité du système.

Maintenance préventive adaptée à notre environnement

Le bon fonctionnement du modèle dépend directement de la qualité et de la stabilité des images d'entrée. Dans notre cas, nous avons identifié plusieurs facteurs pouvant impacter cette qualité : variation de l'éclairage, déréglage de la caméra, perte de netteté ou déplacement des pièces dans le champ de vision.

Pour y répondre, nous avons mis en place une **maintenance préventive hebdomadaire**, structurée autour des points suivants :

- Calibration optique des caméras : mise au point et centrage
- Vérification de l'éclairage : homogénéité, température de couleur, intensité
- Utilisation de pièces de référence extraites du dataset fourni par Valeo, permettant de vérifier visuellement la stabilité des images

Cette maintenance est effectuée systématiquement en début de semaine, avant toute session d'inspection. Elle permet de s'assurer que le modèle travaille dans des conditions constantes, ce qui est une garantie de performance et de fiabilité.

Maintenance évolutive et détection de dérive

Même avec des conditions de prise de vue stabilisées, un modèle IA peut voir ses performances se dégrader avec le temps. Cela peut être dû à des variations dans les pièces produites, à de nouveaux types de défauts, ou à un vieillissement du matériel.

Pour cela, nous avons mis en place une **maintenance évolutive** qui repose sur deux grands leviers :

- Le **réentraînement régulier du modèle**, à partir des images récemment mal classées, afin de corriger les erreurs les plus fréquentes et d'actualiser le comportement du modèle.
- La **détection automatique des dérives**, via l'analyse de l'incertitude des prédictions (calculs d'entropie softmax) et la comparaison hebdomadaire avec les validations manuelles effectuées par les opérateurs.

Dès qu'une dérive est détectée, une alerte est remontée, et un cycle de vérification est lancé. Si la dégradation est confirmée, un nouveau modèle est entraîné et soumis à validation avant déploiement.

Grâce à ce système, nous avons pu garantir une stabilité dans les prédictions, tout en permettant au modèle d'évoluer avec les conditions réelles de production.

Bilan du projet / REX

Voici un rétroplanning du projet :

Phase	Jalon	Objectifs	Date de fin effective
Exploration du dataset	01/03/2025	Produire une série de graphiques permettant de mieux interpréter les données mises à disposition (quantité, nature...) afin d'aiguiller l'entraînement du modèle.	01/03/2025
Réalisation d'un MVP	01/04/2025	Entraîner un premier modèle respectant le CDC du challenge, avec des performances encore à améliorer	15/04/2025
Optimisation des performances	01/05/2025	Fine tuner les paramètres du modèle / revoir l'architecture du NN pour améliorer le F1-score.	15/04/2025
Industrialisation	Jalon final	Rédaction d'un plan d'action permettant de déployer ce modèle dans un process industriel	22/05/2025

Comme on peut le constater, du retard a été accumulé à partir de la phase 2. La raison principale est le problème hardware qui nous a forcé à passer sur des solutions cloud computing.

Dans l'ensemble, les délais ont été respectés, et les objectifs du projet atteints.

Les points clés d'amélioration si c'était à refaire seraient les suivants :

- favorsier davantage de communication entre les équipes data et industrialisation
- faire l'acquisition d'un GPU NVIDIA car Google Colab, bien que gratuit, ne propose qu'un temps de calcul limité, ce qui nous a posé problème pour réentraîner les modèles.
- se former sur la librairie PyTorch, qui est la 2e librairie de référence pour le Deep Learning, juste après TensorFlow (bien que dans le monde académique ce soit PyTorch qui domine).
- normaliser la structure du dépôt Google Drive, et utiliser des conventions de nommage. En effet, il y a eu des problèmes de mélanges de fichiers en raisons de conventions de nommages non-établies, et même de pertes de données (des fichiers de modèles qui se

sont retrouvés écrasés, ce qui nous forcé à réentraîner le premier modèle, qui heureusement était petit : qu'est-ce que cela aurait été si c'était un modèle plus grand ?)

Le message de l'ensemble de l'équipe est le suivant :

Ce projet nous a permis de collaborer sur des problématiques modernes d'industrie digitale, avec un ancrage fort dans le concrêt notamment grâce au travail sur des données réelles fournies par le constructeur Valeo, mais aussi en proposant une implémentation réelle du modèle dans le cadre d'un processus de production. Nous pensons que ce projet a été bénéfique pour nos parcours respectifs, car il recrée un contexte réel de collaboration entre data scientists et ingénieurs.