Slide 4

Avant de commencer, il est important de clarifier trois concepts clés que vous entendrez à plusieurs reprises au cours de cette présentation : la détection d’objets, la segmentation sémantique et la segmentation d’instances. Tout d’abord, la détection d’objets (Object Detection) consiste à localiser chaque objet dans une image à l’aide de boîtes englobantes cependant Ne précise pas la forme exacte de l’objet, juste sa position. quant à la Semantic Segmentation elle permet d'obtenir Une carte de segmentation où chaque pixel a une classe, mais Ne différencie pas les instances. Deux cellules collées apparaîtront comme une seule masse de pixels "cellule". Enfin, la segmentation d’instances combine les deux approches précédentes. Elle permet de segmenter chaque objet individuellement, pixel par pixel, en fournissant à la fois sa position précise et sa forme exacte, tout en le différenciant des autres objets même s’ils appartiennent à la même classe.

Slide 5

Pour mieux comprendre le fonctionnement d’un réseau de neurones, prenons l’exemple du cerveau humain, et plus précisément du cortex visuel. Lorsqu’une image est captée par la rétine, les signaux sont transmis au cerveau via le nerf optique. Le traitement débute au niveau du cortex visuel primaire, qui détecte des caractéristiques abstraites comme les bords, lignes, orientations et contrastes. Ensuite, V2 et V4 extraient des formes plus complexes, des couleurs et des contours fermés. Enfin, IT (cortex inféro-temporal) reconnaît des objets entiers, des visages ou des scènes. Ainsi, ce qui est important de retenir c’est que plus on progresse dans le cortex, plus les informations deviennent abstraites et structurées.

Slide 6

En parallèle, un réseau de neurones est formé de plusieurs couches, et dans chaque couche, il y a ce qu’on appelle des nœuds ou neurones. Un réseau de neurones repose majoritairement sur des opérations de convolution. Dans chaque neurone, un filtre est responsable d’extraire une information particulière de l’image. Comme vous pouvez le voir, lorsque l’information arrive au niveau d’un nœud dans une couche, le filtre balaie l’image. Chaque nœud possède un filtre qui extrait une information bien précise : certains filtres détectent les bordures extérieures, d’autres les bordures intérieures, ou encore des lignes horizontales, verticales ou obliques. Le résultat de chaque nœud s’appelle une carte de caractéristiques, et ces caractéristiques transitent vers la couche suivante, et ainsi de suite.

Slide 7

Tout comme le cortex visuel humain, vous pouvez voir que la première couche du réseau est équivalente à IT, où l’on peut reconnaître des objets complets, comme des visages par exemple. Plus on avance dans les couches du réseau, plus l’information devient abstraite. Les couches les plus profondes du réseau ressemblent à la zone V1 chez l’humain, où l’on retrouve des informations sur les bordures, les contrastes, les lignes, et d’autres caractéristiques abstraites.

Slide 8

L’objectif initial de mon travail était de développer un modèle capable de détecter les éléments de platine. Pour atteindre ce but, j’ai opté pour une approche de segmentation d’instance pour être capable de détecter individuellement chaque éléments de platine pour cela j’ai utilisé un modèle nommer Detectron2.

Slide 8

Après avoir entraîné le modèle sur notre base d’apprentissage, il faut le tester sur la base de test. Le résultat de l’inférence du modèle ressemble à ceci : les éléments de platine sont détectés avec un code couleur. Les objets en vert correspondent à ceux pour lesquels le modèle est sûr à plus de 70 % qu’il s’agit bien des objets qu’on souhaite détecter. En rouge, le modèle est moins sûr. Nous avons fixé un seuil et supprimé les détections en dessous de 30 %, en les considérant comme des faux positifs qu’il ne fallait pas garder.

Slide 9

Maintenant que les éléments de platine sont détectés, on peut calculer des statistiques, comme la fréquence des éléments de platine en fonction de la surface des particules détectées, ainsi que le diamètre moyen des nanoparticules, des clusters, ou encore des atomes isolés.

Slide 10

Au vu de la manière dont les annotations de nos images ont été réalisées, le modèle a bien appris à ne pas détecter d’objets dans les zones épaisses, qui apparaissent comme des régions très blanches sur l’image. Cette absence de détection s’explique par le fait que nous avons estimé que les atomes isolés (≤ 0,2 nm) ne peuvent pas être repérés dans ces zones, contrairement aux nanoparticules (> 0,5 nm) qui restent visibles, ce qui peut introduire un biais dans le comptage. Cependant, certains éléments situés aux bords de ces zones épaisses continuent d’être identifiés par le modèle. Pour cela, nous avons calculé la différence entre la valeur moyenne des pixels de l’objet et celle de ses voisins proches. Les objets présentant une différence proche de zéro sont difficilement distinguables du fond et peuvent être considérés comme des faux positifs.

Slide 11

Voici une représentation plus complète de la différence de valeur entre l’objet et le fond sur l’ensemble des images de test. On retrouve les différentes classes d’objets : les objets dont la valeur est inférieure à 0,2 correspondent aux atomes isolés, ceux compris entre 0,2 et 0,5 sont des clusters, et ceux dont la valeur dépasse 0,5 sont classés comme des nanoparticules. On peut même attribuer des sous-classes : par exemple, parmi les atomes isolés, ceux dont la différence d’intensité par rapport au fond est supérieure à 10 pourraient correspondre à des atomes isolés superposés, ce qui les rend plus brillants et plus facilement distinguables du fond, comparé à ceux dont la différence est inférieure à 10.

Slide 12

Passons maintenant au deuxième projet sur lequel j’ai travaillé. L’objectif était de développer un modèle capable de détecter automatiquement les feuillets déposés sur un support de catalyseur de type alumine. Le nombre de feuillets par paquet, ainsi que leur longueur, sont des caractéristiques importantes liées à l’activité catalytique. Le but était donc de pouvoir compter le nombre de feuillets par empilement et d’estimer la longueur de chaque empilement.

Slie 14

Cependant, il n’était pas possible d’utiliser le modèle Detectron2 ou tout autre modèle de segmentation classique pour ce projet, car plusieurs défis se présentaient. Tout d’abord, il y avait trop de bruit dans les images, et les feuillets pouvaient être coupés et considérés comme deux éléments distincts alors qu’ils devraient être comptabilisés comme un seul. De plus, les feuillets pouvaient apparaître sous plusieurs orientations, ce qui rendait les modèles de segmentation d’instance, commeDetectron2, inadaptés, car ils ne sont pas invariants à la rotation. Pour résoudre ces problèmes, j’ai opté pour l’utilisation du modèle mmrotate, qui propose des architectures invariantes à la rotation.

Slide 15

La principale différence entre MMRotate et les modèles de détection d’objets classiques réside dans sa capacité à prendre en compte l’orientation des objets. Contrairement aux approches traditionnelles qui prédisent des boîtes rectangulaires alignées aux axes, MMRotate permet de générer des boîtes englobantes orientées, mieux adaptées aux objets inclinés ou disposés dans des directions variées.

Slide 16

Après avoir effectué l'entraînement du modèle et réalisé des tests sur une image, on observe que le modèle génère majoritairement des boîtes englobantes pour chaque empilement de feuillets. Pour rappel, l’objectif est d’estimer la longueur des empilements ainsi que le nombre de feuillets par empilement. Concernant la longueur, nous avons choisi d’utiliser la diagonale de la boîte englobante comme approximation de la longueur du plus grand feuillet dans chaque empilement.

Slide 17

Concernant la détection du nombre de feuillets par empilement, un autre type de modèle a été utilisé : un modèle de régression XGBoost. Ce modèle prend en entrée des caractéristiques sous forme de valeurs numériques et cherche à trouver des corrélations entre ces différentes données pour générer en sortie une prédiction sous forme d’un nombre entier, ce qui correspond bien à notre objectif. Pour cela, différentes caractéristiques ont été calculées pour chaque boîte englobante détectée, telles que la forme et les contours (HOG), les points clés distinctifs (ORB), la texture globale (GLCM), la texture locale (LBP), ainsi que la distribution des intensités (histogramme). Ces données ont été fournies au modèle de régression. Cependant, j’ai constaté que cette approche, bien que fonctionnelle, manquait de robustesse à cause du bruit important présent dans les images, ce qui compliquait la tâche. Pour y remédier, j’ai eu l’idée d’associer à ces caractéristiques classiques des caractéristiques extraites par un modèle CNN DenseNet, qui capture des relations plus détaillées et abstraites dans l’image. En combinant ces deux types de caractéristiques et en les fournissant au modèle de régression, nous avons obtenu un modèle beaucoup plus robuste. Comme vous pouvez le voir, pour une image d’entrée, le nombre de feuillets prédit est affiché dans le titre de l’image.

Slide 18

Tout ce processus réalisé nous permet de détecter la localisation des groupes de feuillets, d’estimer le nombre total de feuillets dans l’image, ainsi que la longueur des empilements et le nombre de feuillets par empilement.

Slide

Dans le but d’évaluer l’exactitude des résultats obtenus par notre modèle, ces résultats ont été obtenus par Mariana ; elle a pris le temps de les comparer avec ceux mesurés manuellement sur la même image. À travers cette comparaison, on peut s’apercevoir qu’en termes de longueur moyenne des empilements, on obtient plutôt des mesures très similaires. Cependant, concernant le nombre de feuillets détectés par empilement, notre modèle en détecte davantage. Néanmoins, comme l’empilement moyen est un entier (soit 1 soit 2), en arrondissant à l’entier inférieur, on tombe plutôt sur les mêmes valeurs.

Slide 19

Vincent, le coq du R52, a souhaité pouvoir identifier les pics présents dans les chromatogrammes et attribuer à chacun le nom du composant correspondant.

Slide 21

Concernant le modèle utilisé pour ce projet, on m’a proposé d’utiliser celui développé par Florent Haffner. Cependant, ce modèle est de type régression : il prend en entrée une série de points, mais ne prédit en sortie qu’un seul nombre. Pour répondre à notre besoin — obtenir une sortie de 1 000 points à partir de 1 000 points en entrée — il a donc fallu modifier l’architecture IPA. L’ajout d’une couche BiLSTM a permis de mieux capturer les relations temporelles sur des signaux longs, tout en produisant une sortie de même longueur que l’entrée. Chaque point prédit en sortie indique alors s’il correspond à un pic ou non.

Slide 22

En termes de métrique évaluation pour la localisation des pics on est plutôt pas mal, le modèle arrive bien à prédire le pic dans l’exemple là au bon endroit et on voit bien là les pics détectés pour tout un segment. À ce stade, les pics présents dans un signal chromatogramme sont détectés, ce qui permet d’avoir le temps de rétention correspondant à chaque pic. Le nom du composé est généralement attribué au pic selon son temps de rétention. La problématique est que, au cours de plusieurs analyses GC, le temps de rétention d’une substance donnée peut varier : on peut avoir dans un chromato une substance qui apparaît à un temps x1 et dans un autre chromato une autre substance qui apparaît au même temps x1 ou à un temps proche, ce qui crée des confusions.

Slide

Pour s’affranchir de cette problématique, on a constitué un Excel de référence comportant les composants qui peuvent apparaître dans un signal avec leur indice de rétention. Grâce à cette formule, connaissant l’indice de rétention et les temps de rétention des alcanes, on a pu calculer le temps de rétention de référence de chaque composant. Maintenant, pour un chromatogramme dont on a détecté les pics, on demande à l’utilisateur de spécifier le pic qui correspond au toluène, et à travers cette valeur on décale les valeurs de référence de notre Excel. Ces valeurs ajustées sont données à manger à un algorithme du plus proche voisin ; les temps de rétention des pics détectés par IPA sont ensuite donnés à l’algo qui attribue le nom du composé basé sur la base apprentissage. Voici un exemplaire là où vous pouvez voir que le modèle arrive à détecter les pics à leur bon temps de rétention et à attribuer le bon nom de composant aux pics.

Slide 23

Une image peut être considérée comme un ensemble de valeurs de pixels, où chaque pixel représente une intensité. On peut donc la voir comme un nuage de points formé par ces valeurs. Le fonctionnement de K-Means repose sur ce principe : si l'on souhaite diviser nos données en 5 groupes (ou clusters), l’algorithme commence par choisir aléatoirement 5 points appelés centroïdes initiaux, qui représentent les centres temporaires des clusters. Ensuite, chaque point est affecté au centroïde le plus proche. Une fois cette assignation faite, on recalcule les centroïdes en prenant la moyenne des points de chaque groupe. Ces deux dernières étapes (assignation et recalcul) sont répétées jusqu’à convergence, c’est-à-dire lorsque les centroïdes ne changent plus ou très peu.

Slide 24

Pour ce type de grandissement, je n'avais à ma disposition que 12 images. Entraîner un modèle supervisé sur seulement 12 images, c’est, on peut le dire, perdu d’avance. Pour pallier ce problème, chaque image a été divisée en 4 parties, ce qui nous donne un total de 48 images. Pour augmenter encore la quantité de données, des transformations ont été appliquées : des changements de luminosité ainsi que des rotations à 90 et 180 degrés. Cela nous a permis d’obtenir un ensemble de 240 images pour entraîner notre modèle.

Slide 25

Le modèle choisi pour ce projet est le même que celui utilisé pour les éléments de platine, à savoir Detectron2. On peut dire que le modèle parvient globalement à bien détecter les différentes zones denses et creuses dans les images.

Slide

La problématique ici est que pour générer les microstructures, il fallait spécifier la taille des trois demi-axes x, y, z pour chaque phase. Pour cela, il fallait trouver une technique permettant d’inférer des propriétés 3D à partir de mesures 2D. Le but est d’estimer les tenseurs volumiques (T\_k) (pour k = 0, 1, 2) d’une particule 3D (X). Ces tenseurs capturent la taille (volume), la position (centre de gravité) et la forme/orientation (via l’ellipsoïde de Miles). Dans ce genre de situation, on est obligé de poser une hypothèse : on suppose que la distribution des particules est invariante sous rotation autour de l’axe vertical. On prend un plan vertical (L), contenant l’axe vertical (VA) et passant par le point de référence (O). On superpose à cette section 2D un ensemble de demi-droites parallèles et perpendiculaires à l’axe vertical. On identifie les points où ces demi-droites croisent le contour de la particule, avec un signe alterné (+ ou -), en commençant par un "+" pour le point le plus éloigné de l’axe. Ce signage permet de transformer une mesure de longueur en volume. On obtient alors (g\_0) (volume), (g\_1) (vecteur position (T\_1(X))) qui mene au centre de gravite, et le tenseur de forme (T\_2(X)) (répartition du volume autour du centre de gravité). A partir des valeur propre de g2 on calcule les 3 demi axes. Enfin, on fait la moyenne de ces estimations en pondérant par le volume des particules.

Slide

Ensuite, il fallait trouver quelle taille pour notre cube. Pour cela, la taille est calculée en se basant sur le rapport entre la somme des volumes de chaque phase et la somme des fractions volumiques des phases, puis on multiplie la valeur obtenue par 4 ou 5 pour modéliser la matrice avec des agrégats faisant au moins 3 pixels. À partir des détections obtenues sur les images à grandissement 250x, on calcule la fraction volumique des phases et les trois demi-axes. À partir des détections sur les images 15kx, on calcule le rayon moyen des cristaux. Et à partir du masque non supervisé, on obtient la fraction volumique des agrégats de la phase intermédiaire , et on initialiser la fraction volumique de la matrice avec cette valeur.

Slide

Comme dans la bille, les zones creuses et denses ont été divisées en deux sous-classes. On a d’abord généré les deux microstructures pour les zones creuses, puis fait leur union et pris le complément pour obtenir la microstructure creuse. Pareil pour la dense : on générait les deux, puis on faisait l’union. Ensuite, on générait celle correspondant à l’intermédiaire, puis celle de la matrice, et on faisait l’union entre la dense, l’intermédiaire et la matrice. On faisait ensuite l’intersection avec le complément de la creuse pour obtenir la microstructure finale. Concernant la matrice, historiquement les images à 15kx sont prises dans des zones intermédiaires où les cristaux sont bien individualisés. On s’est servi de ces images pour initialiser la matrice, sauf que cela ne reflète pas la compacité réelle. Pour corriger, on calcule la fraction volumique de l’espace poreux obtenue par le résultat final, on fait la différence avec le volume injecté de poromercure, puis on régénère une deuxième fois la matrice corrigée, et on tombe sur la fraction volumique de l’espace poreux attendue à la fin.

Slide

En plus d’avoir une fraction volumique finale très proche de la fraction volumique attendue, et dans le but de vérifier si on conserve bien la continuité matrice et porosité inter-agrégat, on a lancé le module PNP de Plugim! sur la microstructure finale. Ce module permet de calculer la distribution en taille de pore, qu’on compare ensuite au poromercure. En raison de limitations de calcul, on est limité en résolution et on descend jusqu’à 200 nm de diamètre. Dans la courbe bleue, on n’a pas ce pic car il faut passer par le petit pour accéder au grand : quelque part, les volumes des grands pores apparaissent dans les petits. Dans la microstructure qu’on génère à 200 nm, en additionnant avec la valeur du pic, on tombe sur le même volume cumulé à 200 nm que dans les données Excel.

Conclusion

Pour conclure, concernant la détection des éléments de platine dans les images STEM-HAADF et les feuillets, comme pour la plupart des projets sur lesquels j’ai travaillé, nous ne disposions pas d’annotations très fiables. Il fallait donc passer par des vérifications visuelles, des étapes de post-traitement et des comparaisons avec les statistiques manuelles, et on peut dire que les performances des modèles sont plutôt bonnes, logiques et cohérentes. Concernant Eluxyl, on a fait face à des limitations en termes de résolution et de temps de calcul. Même en ayant pu générer une microstructure avec une résolution inférieure à 100 nm, le module PNP de Plugim! buguait et n’arrivait pas à traiter les grandes tailles de microstructure. Pour les perspectives futures, une interface web est en cours de développement pour intégrer tous les modèles développés afin qu’ils soient facilement utilisables. Pour Eluxyl, les modèles booléens n’offrent pas la possibilité d’avoir une distribution de taille des zones denses et creuses : on n’a pas de distribution des diamètres des objets, seulement des tailles uniques. Ce serait intéressant d’avoir une distribution des diamètres pour les différentes zones (dense, creuse, intermédiaire), et aussi d’héberger les modèles Eluxyl sur des serveurs pour viser plus haut et essayer d’atteindre des résolutions inférieures à 100 nm.

Remerciements

Pour finir, je souhaite remercier chaleureusement mes encadrants. Je remercie Virgile pour son suivi et ses précieux conseils tout au long de mon alternance. Je tiens également à remercier le chef de département Tivadar, qui rend le cadre de travail agréable, ainsi que les ingénieurs et techniciens du R05. Les modèles IA sont essentiels, mais sans eux, il n’y aurait pas de données ni d’annotations pour les entraîner. Je n’oublie pas non plus mes collègues du bureau, qui ont grandement facilité mon intégration à IFPEN et m’ont apporté leur soutien. Merci à tous.