



L'expert en reconnaissance de styles architecturaux grâce à l'intelligence artificielle
par SARTORIS Jean-Paul.S.I.

Sommaire

1. Introduction.

2. Présentation du project MyArchiStyle.

2.1 Cas d'utilisation et clientèle.

2.2 Type d'images et condition de prise.

2.3 Collection de styles architecturaux pris en charge.

3. Interface utilisateur et expérience utilisateur

3.1 Interface utilisateur

3.2 Expérience de l'utilisateur

4. Dataset

4.1 Constitution du dataset

4.2 Stockage du datasets.

4.3 Traitement et nettoyage des données

4.5 Data Augmentation pour les datasets d'images

5. Architecture globale & choix de technologies

6. SQL & archi BDD

Structure de la base de données

Relations entre les tables

Hachage des mots de passe

Choix de conception

Considérations techniques

7.API

Endpoints et Fonctionnalités

Intégration du Modèle

Gestion de la Base de Données

Sécurité et Performance

8. Monitoring

Graphique de Performance

Notifications d'Incidents

Logs du Serveur

9. Management

Gestion Proactive des Défis Infrastructurels

Suivi et Collaboration à l'aide d'Outils de Gestion de Projet

10. modèles

10.1 from Scratch

Architecture du Modèle:

Performances:

10.2 Utilisation de VGG16 un Modèles Pré-Entraînés

10.3 vgg16 transfer learning:

Intégration de VGG16 dans l'Architecture du Modèle

Définition de la Topologie Personnalisée:

Performances:

10.3 vgg16 Fine Tuning:

Structure du Modèle

Configuration d'Entraînement

Entraînement du Modèle

Conclusion et Performances

10.4 Sélection du modèle

Autres facteurs à prendre en compte:

Conclusion:

11. Axe d'amélioration

Détection de Contexte dans les Images

Réutilisation des Images Utilisateurs

Fonctionnalités de Collaboration et Partage de Groupes d'Images

Extension de la Collection d'Art Architectural

Création de Rapports Analytiques et Graphiques automatique

Expérimentation avec Divers Modèles Pré-Entraînés

1. Introduction.

Je suis [SARTORIS Jean-Paul](#) et je présente ce projet dans le cadre de ma formation de développeur en intelligence artificielle et Big Data. J'ai commencé ma formation le 10 octobre 2022 pour mon diplôme. Je dois réaliser un projet de fin de formation qui devra respecter les critères de la [grille d'évaluation](#) pour passer mon diplôme [RNCP37827](#) d'ici fin avril 2024. Ce document constitue le rapport de mon projet intitulé "MyArchiStyle" actuellement achevé à 20%.

2. Présentation du project MyArchiStyle.

2.1 Cas d'utilisation et clientèle.

MyArchiStyle est une application éducative et ludique conçue pour aider les utilisateurs à découvrir et à apprendre l'art architectural. Elle s'adresse à un large public, allant des touristes en quête de découvertes culturelles aux écoliers désireux d'enrichir leurs connaissances, en passant par les passionnés de tout âge qui souhaitent approfondir leur compréhension de l'architecture. Elle est accessible par une application web qui permet de télécharger une image pour en identifier les styles architecturaux dominants. Un commentaire détaillé est fourni en ligne avec l'image.

2.2 Type d'images et condition de prise.

Les images du Dataset et des utilisateurs devront respecter certains critères afin de permettre la détection du style du lieu. L'image doit comporter un bâtiment de type château, immeuble, religieux, théâtre, casino... avec des conditions d'environnement de prise de vue de jour en extérieur et sans jardin, ni fontaine, ni mobilier urbain.

2.3 Collection de styles architecturaux pris en charge.

Les arts sélectionnés pour la prédiction sont les suivant :

1. Gothique : (Gothique classique, Gothique primitif, Gothique flamboyant, Gothique Rayonnant, Gothique Tardif). Ne prend pas en compte le Gothique International.
2. Renaissance (Renaissance de tout pays européen)
3. Art Deco
4. Byzantin
5. Romanesque, roman
6. Baroque
7. Rococo
8. Brutalisme, modernisme , fonctionnalisme
9. Classique, grec antique (Dorique, Ionique, Corinthien)
10. Art Nouveau
11. Autre styles (Lorsque les modèles n'arrivent pas à classer avec précision)

Les styles Néo sont classés dans leurs arts initiaux.

Cette liste n'est pas fixe. À l'avenir, dans les versions à venir, de nouveaux styles seront ajoutés, mais aussi la prise en charge de la reconnaissance des intérieurs.

voici un exemple par style:

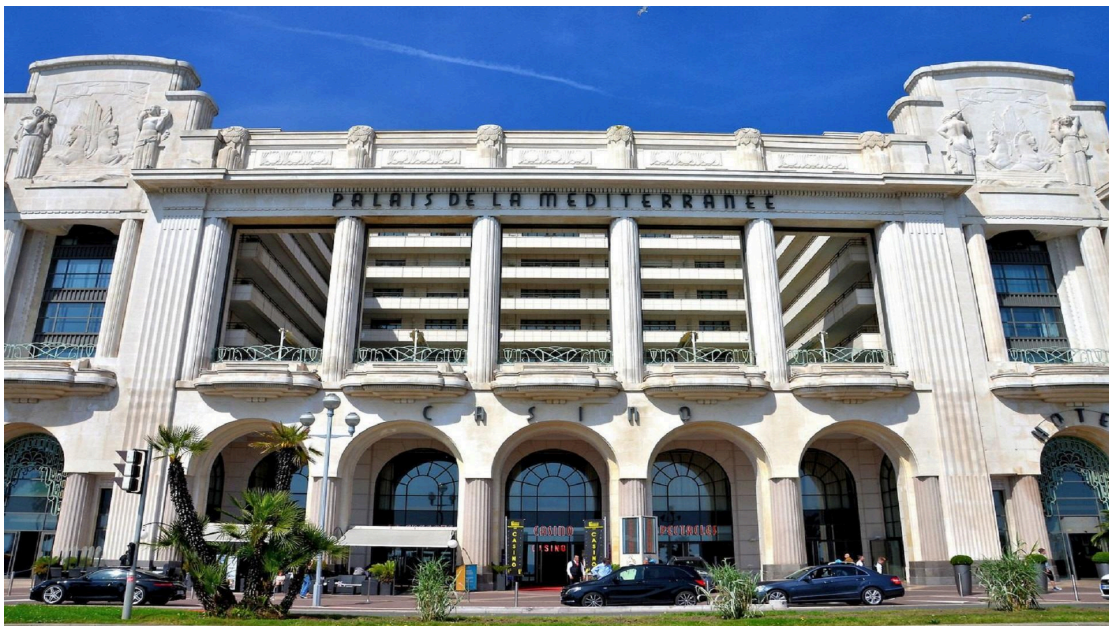
1.Gothique:



2.Renaissance:



3.Art Déco:



4.Byzantin:



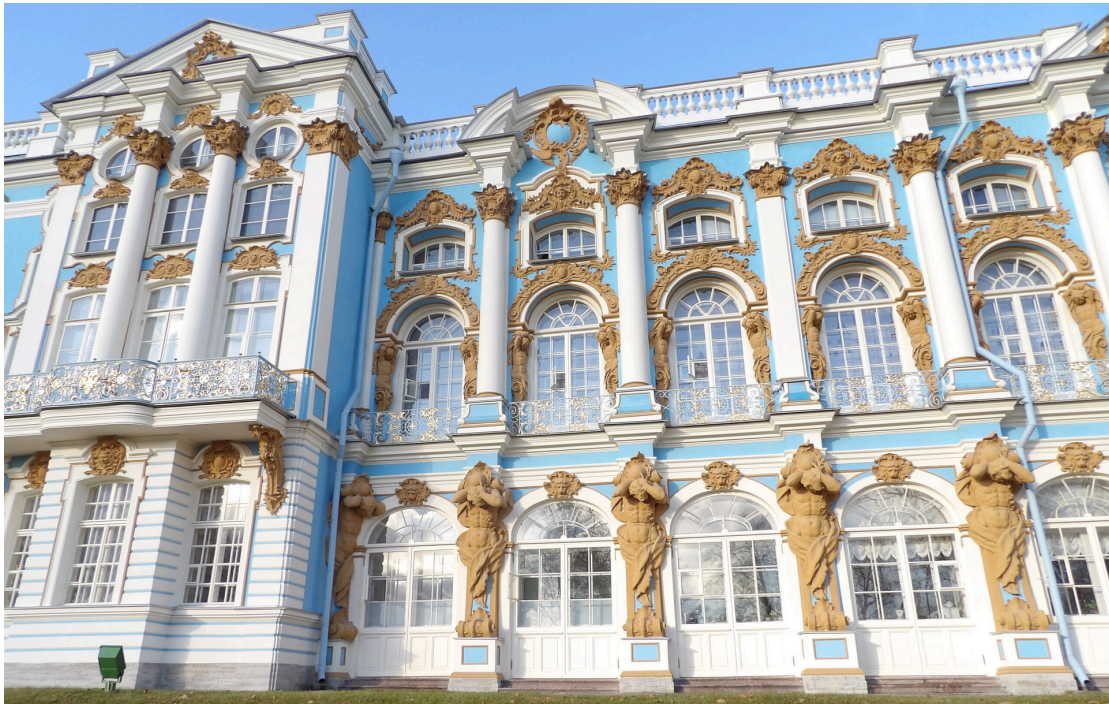
5.Romanesque:



6.Baroque:



7.Rococo:



8.Brutaliste:



9.Classique:



10. Art Nouveau:




3. Interface utilisateur et expérience utilisateur


3.1 Interface utilisateur


L'interface de l'utilisateur se présentera comme une page web accessible sur PC, mobile et tablette. Elle sera constituée d'éléments simples et minimalistes avec une barre de menu et de profil sur la partie supérieure de l'écran. Elle est réaliser avec flask.

Les couleur seront sombres et neutres, pas de couleurs vives :

fond : #0a101e 

couleur des boîtes : #152436 

couleur des contours : #20354f 

couleur de secondaire : #f15662 

couleur du texte : #ffffff (blanc)

Le logo ne prend pas ces codes couleurs mais reste dans des tons sombres bleus et rouges.

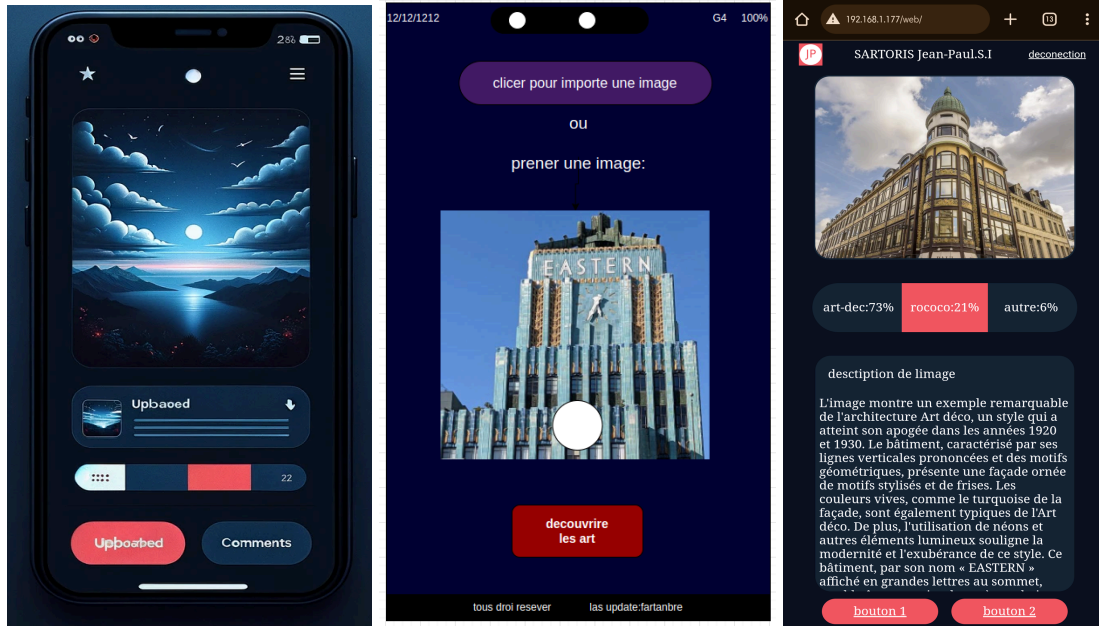
Évolution de l'interface :

Planche

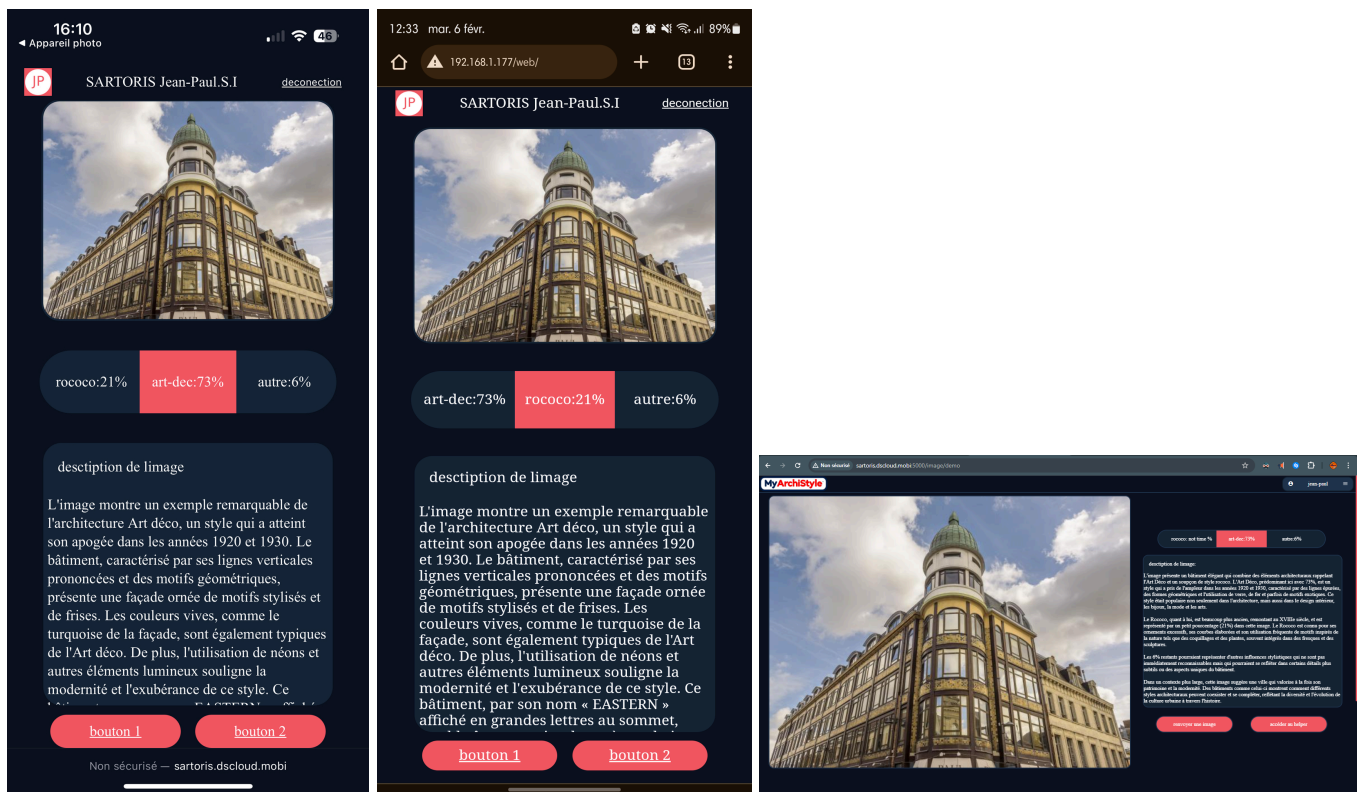
Croquis

Capture d'écran

MyArchiStyle



Exemple d'interfaces :

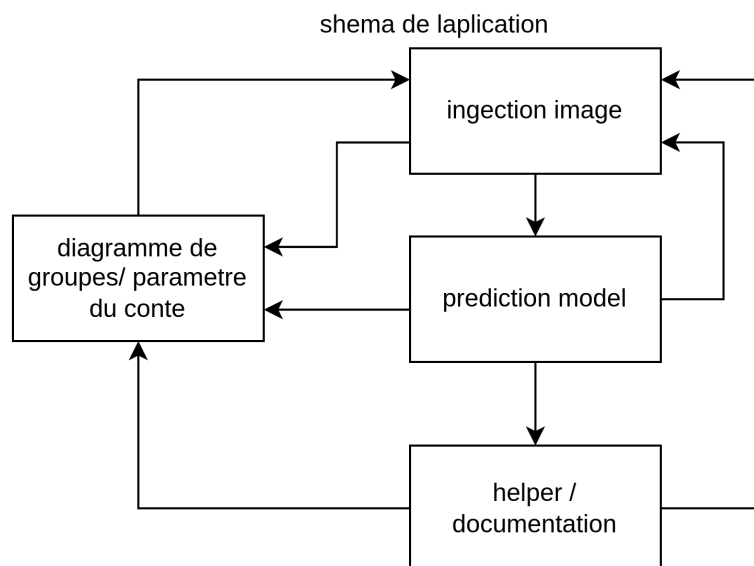


3.2 Expérience de l'utilisateur

L'architecture des pages de l'application MyArchiStyle reste minimaliste avec seulement 4 pages :

1. Ingestion de l'image. L'utilisateur prend ou sélectionne une image ; une fois celle-ci téléchargée, il est redirigé vers la page de prédiction.
2. Prédiction. L'utilisateur peut consulter le résultat de la prédiction avec un commentaire généré par GPT vision sur son image. L'utilisateur a ensuite la possibilité de revenir à l'ingestion d'image ou de consulter la page d'aide.
3. Helper. Cette page présente les arts pris en charge par MyArchiStyle.
4. Page du profil. Cette page affiche votre profil et vos groupes d'images suivies de leur analyse.

Ce schéma décrit une architecture des pages simple et directe, ce qui devrait permettre une navigation facile pour les utilisateurs de l'application MyArchiStyle.



4. Dataset

4.1 Constitution du dataset

Il n'existe pas de dataset correspondant à mon projet, mais je recherche des datasets centrés sur un type de bâtiment et un style architectural donné. Les styles des datas trouvés ne correspondent pas au style architectural sélectionné. J'ai donc constitué dans une liste de bâtiments classés par catégorie, puis je me suis mis à constituer mon dataset. Ce dataset a été constitué avec des images provenant de [google image](#) et de [google maps](#). Les images ont été téléchargées au nombre de 100 (actuellement, mais il est prévu d'en ajouter jusqu'à 500) par catégorie. Avec 100 images par catégorie, le nombre total d'images est de 1000, et avec 500, il sera de 5000.

Les critères de sélection des images sont les mêmes que ceux des contributions des [images des utilisateurs](#) afin de réduire le DATA-DRIFT.

4.2 Stockage du dataset.

Le dataset d'images est hébergé dans l'infrastructure SartoCloud, ce qui me permet de rester propriétaire des données tout en pouvant réagir rapidement à d'éventuelles failles de sécurité et gérer les droits des utilisateurs de manière efficace.

Processus de requête et d'authentification

Lorsqu'une demande d'accès au dataset est initiée, le processus suivant est mis en œuvre :

1. **Authentification**: Chaque demande d'accès au dataset nécessite une authentification réussie de l'utilisateur demandeur. Cette mesure assure que l'accès aux données est strictement réservé aux utilisateurs autorisés.
2. **Création d'une archive instantanée**: Au moment où l'authentification est validée, SartoCloud crée une archive du dataset à l'instant T. Cette archive représente une copie exacte du dataset au moment de la demande, assurant ainsi l'intégrité des données lors de leur manipulation ou transfert.
3. **Transmission de l'archive**: Une fois l'archive créée, elle est envoyée au demandeur. Ce processus est sécurisé pour garantir que seules les personnes autorisées puissent accéder aux données.

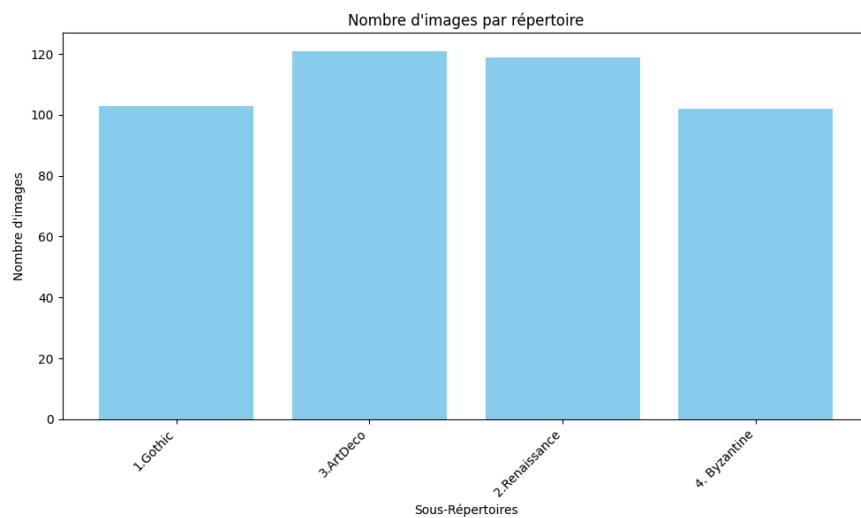
Sécurité et gestion des droits

SartoCloud intègre des fonctionnalités avancées pour protéger les données et contrôler l'accès :

1. **Surveillance continue**: L'infrastructure surveille en permanence l'accès aux datasets pour détecter et réagir rapidement à toute activité suspecte.
2. **Gestion des droits utilisateurs**: Les droits d'accès sont finement gérés pour s'assurer que chaque utilisateur ait uniquement accès aux données nécessaires à ses fonctions.

Ces mesures garantissent que le dataset d'images est non seulement bien protégé contre les intrusions et les fuites de données, mais aussi accessible de manière contrôlée et sécurisée.

exemple de visualisation du Dataset:



1. Gothic



3. ArtDeco



2. Renaissance



4. Byzantine

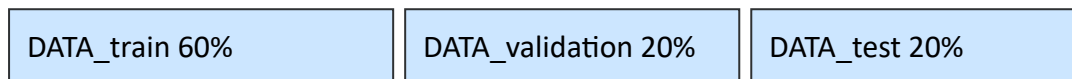


4.3 Traitement et nettoyage des données

Les images ont été renommées et l'uniformisation des images a été réalisée en utilisant des scripts Python avec les modules Os et OpenCV pour ce traitement.

Ensuite le dataset est divisé en trois : data_train (60%), data_validation (20%) et data_test (les 20% restants).

Voici un schéma de préparation du Dataset :



Le data_train, data_validation et data_test sont des partitions couramment utilisées dans le domaine du machine learning et de l'apprentissage automatique. Elles servent à diverses étapes du processus d'entraînement et d'évaluation d'un modèle. Voici une documentation pour expliquer ces concepts :

- data_train: La partition data_train, qui représente 60% des données disponibles, est employée pour entraîner un modèle de machine learning. Cette phase d'apprentissage permet au modèle d'ajuster ses paramètres internes en fonction des exemples présents dans cette partition, afin de découvrir des patterns et relations qui existent entre les variables d'entrée (features) et la variable cible (label).
- data_validation: La partition data_validation, qui représente 20% des données disponibles, est utilisée pour évaluer la performance du modèle au cours de la phase d'entraînement. Cette étape sert à vérifier que le modèle n'est pas en surapprentissage (overfitting), c'est-à-dire qu'il ne mémorise pas les données d'entraînement au détriment de sa capacité à généraliser les connaissances acquises à de nouvelles données. Le modèle est évalué sur cette partition pendant l'entraînement, et les métriques de performance obtenues sur data_validation permettent d'ajuster les hyperparamètres du modèle pour en améliorer les performances.
- data_test : La partition data_test, qui représente les 20% restants des données disponibles, est employée pour évaluer de manière finale et impartiale la performance du modèle une fois l'entraînement et l'optimisation des hyperparamètres terminés. Cette étape est cruciale pour estimer de manière réaliste les performances du modèle sur de nouvelles données inconnues. Il est important de noter que le modèle n'a pas accès aux données de la partition data_test pendant la phase d'entraînement et de validation, ce qui garantit une évaluation objective et indépendante de ses performances générales.

4.5 Data Augmentation.

La data augmentation est une technique cruciale pour améliorer la performance du modèle de deep learning, spécialement dans le domaine de la vision par ordinateur. En augmentant artificiellement mon dataset par des modifications légères des images.

Grâce à la data augmentation, je prévois une amélioration de la performance du modèle sur des données inédites et une meilleure capacité de généralisation, réduisant ainsi le surajustement et augmentant la robustesse du modèle face aux variations réelles des données.

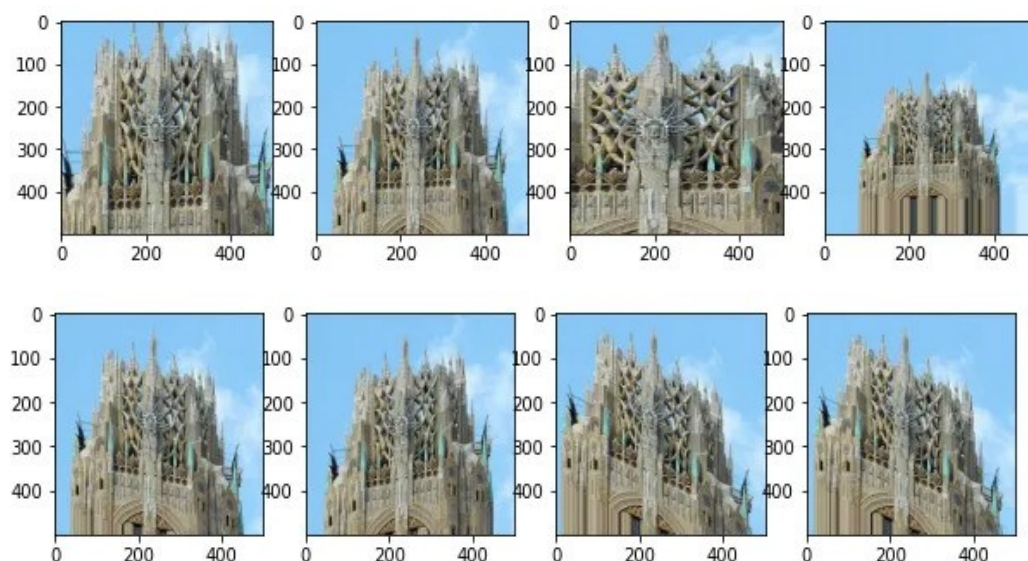
Méthodes d'Augmentation Utilisées

- **Rotation** : Les images sont légèrement tournées dans les deux sens (gauche et droite) pour simuler des angles de prise de vue différents.
- **Translation** : Les images sont décalées verticalement et horizontalement, imitant le mouvement et la position variables de l'objet observé.
- **Zoom** : Un léger zoom avant et arrière est appliqué aux images pour représenter différentes échelles.
- **Renversement horizontal** : Les images sont retournées horizontalement pour simuler une symétrie et diversifier la perspective.
- **Changement de luminosité** : La luminosité des images est ajustée pour modéliser différentes conditions d'éclairage.
- **Bruit aléatoire** : Du bruit est ajouté aux images de façon aléatoire pour simuler des perturbations visuelles et entraîner le modèle à les ignorer.

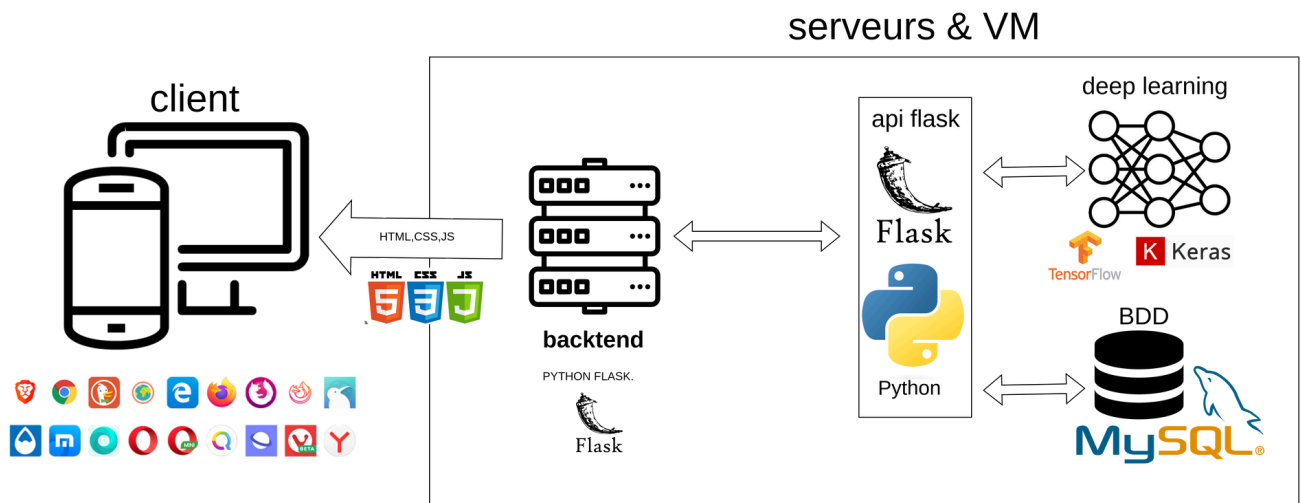
Implémentation avec les images fournies

En utilisant une image d'un édifice du dataset, j'ai appliqué les méthodes d'augmentation de données mentionnées pour générer de nouvelles instances.

en voici un exemple:



5. Architecture globale & choix de technologies



Le choix des technologies a été effectué sur la base des connaissances acquises lors de ma formation et en autonomie, j'ai préféré utiliser les technologies dans lesquelles je suis compétent.

- J'ai choisi TensorFlow et Keras pour leur facilité d'utilisation et leur puissance en matière de machine learning et de deep learning. Ils offrent une grande flexibilité pour créer et entraîner des modèles, ainsi qu'une large communauté de développeurs.
- J'ai choisi MySQL pour sa fiabilité et ses fonctionnalités avancées en matière de gestion de bases de données. Il est largement utilisé dans l'industrie et offre une excellente performance pour le stockage et la récupération de données.
- J'ai choisi Flask et Python pour créer une API (Application Programming Interface) simple et efficace. Flask est un micro-framework léger et flexible pour le développement web, tandis que Python est un langage de programmation puissant et facile à apprendre.
- J'ai choisi HTML, CSS, JS et Django pour créer une application web complète et dynamique. HTML, CSS et JS sont les technologies de base pour la création de sites web, tandis que Django est un framework Python puissant pour le développement web.

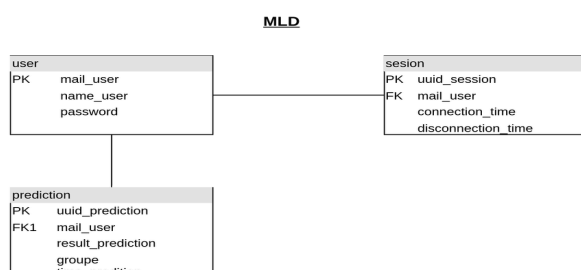
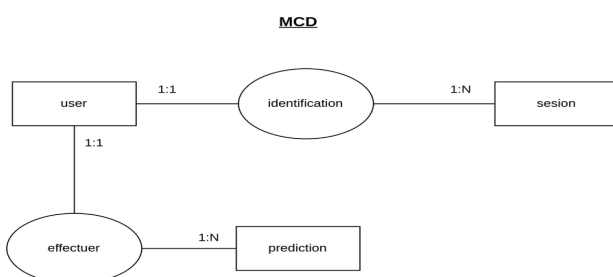
6. SQL & archi BDD

La base de données utilisée pour stocker les données de l'application MyArchiStyle a été conçue en utilisant le langage de programmation SQL. Cette section détaille la structure de la base de données, les tables et les relations entre elles, ainsi que les choix de conception et les considérations techniques.

Structure de la base de données

La base de données MyArchiStyle comprend trois tables :

- **users** : cette table stocke les informations de base des utilisateurs, y compris leur adresse e-mail, leur nom et leur mot de passe. La clé primaire est l'adresse e-mail.
- **sessions** : cette table stocke les informations sur les sessions utilisateurs, y compris l'identifiant unique de la session, l'adresse e-mail de l'utilisateur, l'heure de connexion et l'heure de déconnexion. La clé primaire est l'identifiant unique de la session et la clé étrangère est l'adresse e-mail de l'utilisateur.
- **predictions** : cette table stocke les informations sur les prédictions effectuées par les utilisateurs, y compris l'identifiant unique de la prédiction, l'adresse e-mail de l'utilisateur, le résultat de la prédiction et le groupe. La clé primaire est l'identifiant unique de la prédiction et la clé étrangère est l'adresse e-mail de l'utilisateur.



MPD			
N°	Name	user type	commentary
1	mail_user	VARCHAR(255) NOT NULL	user's email
2	name_user	VARCHAR(255) NOT NULL	user's display name
3	password	VARCHAR(255) NOT NULL	user's password hash

session			
N°	Name	type	commentary
1	uuid_session	VARCHAR(36) NOT NULL	session UUID
2	mail_user	VARCHAR(255) NOT NULL	user's email
3	connection_time	DATETIME NOT NULL	connection time and date
4	disconnection_time	DATETIME	disconnection time and date
5	time_prediction	DATETIME	time of pretiction

session			
N°	Name	type	commentary
1	uuid_prediction	VARCHAR(36) NOT NULL	prediction UUID
2	mail_user	VARCHAR(255) NOT NULL	user's email
3	result_prediction	VARCHAR(255) NOT NULL	result of the prediction
4	groupe	VARCHAR(255) NOT NULL	group of the prediction

Relations entre les tables

La table "users" est liée aux tables "sessions" et "predictions" par une relation one-to-many. Une adresse e-mail dans la table "users" peut avoir plusieurs sessions associées dans la table "sessions" et plusieurs prédictions associées dans la table "predictions". Cette relation permet de relier les données de l'utilisateur aux données de session et de prédiction associées.

Hachage des mots de passe

Pour assurer la sécurité des mots de passe stockés dans la base de données, une fonction de hachage personnalisée a été créée en utilisant l'algorithme bcrypt. Cette fonction est utilisée pour hasher les mots de passe avant de les stocker dans la table "users". L'utilisation d'un algorithme de hachage sécurisé tel que bcrypt permet de protéger les mots de passe contre les attaques par force brute et les attaques par dictionnaire.

Choix de conception

La conception de la base de données MyArchiStyle a été réalisée en utilisant les meilleures pratiques de conception de base de données. Les tables ont été conçues pour stocker les données de manière logique et efficace, en évitant la redondance et en minimisant les dépendances entre les tables. Les clés primaires et étrangères ont été utilisées pour relier les données entre les tables et garantir l'intégrité des données.

Considérations techniques

La base de données MyArchiStyle a été conçue pour être utilisée avec l'application web MyArchiStyle. Les choix de conception ont été faits en tenant compte des exigences de l'application et des contraintes techniques. Par exemple, la table "sessions" a été conçue pour stocker les informations de session des utilisateurs, ce qui permet de suivre les sessions utilisateur et de fournir une expérience utilisateur personnalisée. La table "predictions" a été conçue pour stocker les résultats des prédictions effectuées par les utilisateurs, ce qui permet de suivre les performances de l'application et d'améliorer la précision des prédictions.

La conception de la base de données MyArchiStyle a été réalisée en utilisant les meilleures pratiques de conception de base de données. Les tables nécessaires ont été créées et les relations entre elles ont été définies. De plus, une fonction de hachage personnalisée a été ajoutée pour assurer la sécurité des mots de passe stockés dans la base de données. Cette conception permettra de stocker et de récupérer les données de manière efficace et sécurisée, tout en répondant aux exigences de l'application web MyArchiStyle.

7.API

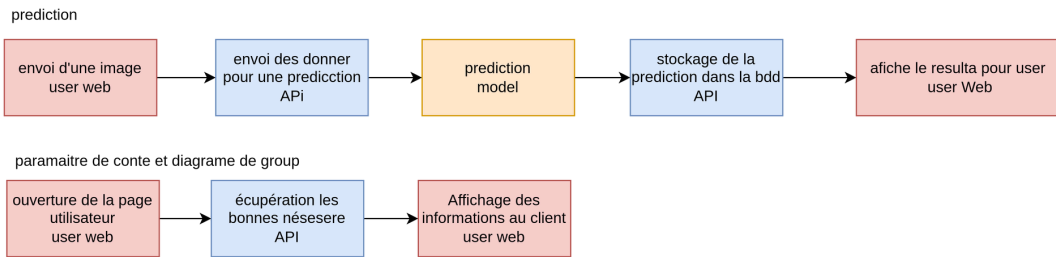
Mon application système utilise une API REST construite avec le micro-framework Flask pour orchestrer les interactions entre l'interface utilisateur web et les opérations côté serveur. Flask est choisi pour sa simplicité, sa légèreté et sa flexibilité, permettant de déployer rapidement des endpoints robustes pour la réception des données, l'exécution des prédictions du modèle et la gestion des transactions avec la base de données.



Endpoints et Fonctionnalités

L'API fournit les endpoints suivants :

- **Envoi d'Image (`/upload`)** : Accepte les images transmises par les utilisateurs via le site web. L'endpoint décode les images et les prépare pour la prédiction.
- **Prédiction (`/predict`)** : Reçoit l'image traitée de l'endpoint `/upload`, l'envoie au modèle de deep learning pour la prédiction, et renvoie les résultats de prédiction.
- **Stockage des Prédictions (`/store`)** : Enregistre les détails de la prédiction, y compris l'identifiant de la session et le résultat, dans la base de données pour une récupération et une analyse ultérieures.
- **Récupération des Données (`/results`)** : Fournit une interface pour récupérer les résultats de prédiction stockés, permettant leur affichage à l'utilisateur ou leur analyse statistique.
- **Connexion (`/login`)** : Permet aux utilisateurs de s'authentifier et d'initier une session sécurisée.
- **Déconnexion (`/logout`)** : Permet aux utilisateurs de terminer leur session sécurisée et de se déconnecter de l'application.



Intégration du Modèle

L'intégration du modèle de deep learning se fait via une interface qui charge le modèle entraîné en mémoire lors du démarrage de l'API. Les appels au modèle sont gérés de manière synchrone pour maintenir l'intégrité des prédictions.

Gestion de la Base de Données

La base de données est accédée via SQLAlchemy, un ORM (Object Relational Mapper) qui simplifie les interactions avec la base de données. Cela permet un code plus propre, une meilleure maintenance et une migration aisée vers différentes bases de données si nécessaire.

Sécurité et Performance

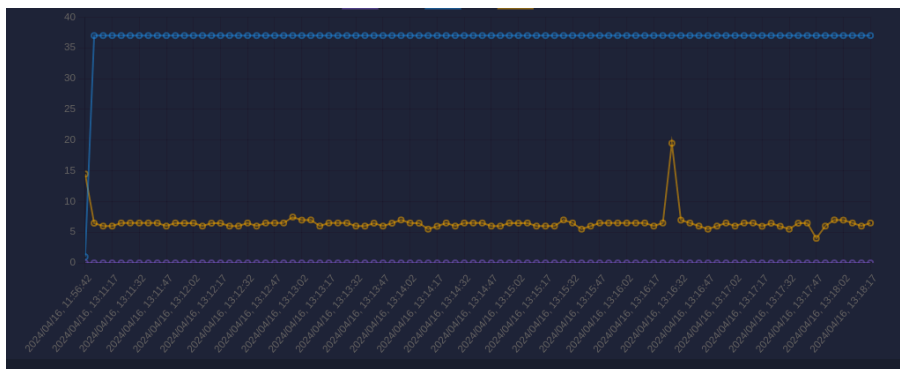
Des mesures telles que la validation des entrées, la gestion des sessions utilisateur et l'authentification sont mises en place pour sécuriser les endpoints contre les utilisations abusives. Les performances sont surveillées et optimisées grâce à des techniques de caching et une gestion efficace de la charge.

8. Monitoring

Le monitoring des performances du serveur est essentiel pour maintenir la stabilité et l'efficacité des opérations informatiques. Non outils de monitoring permettent une surveillance en temps réel et une réponse rapide aux incidents.

Graphique de Performance

Le premier élément de notre monitoring est un graphique de performance qui montre la consommation de la RAM (en jaune) et de l'utilisation du processeur (en bleu) sur une période donnée. Ce graphique révèle les tendances et les pics d'utilisation qui pourraient indiquer des problèmes sous-jacents ou des besoins de mise à niveau de ressources.



Notifications d'incidents

Une notification d'incident est générée automatiquement par le serveur en cas de défaillance critique ou une atteinte à la sécurité des utilisateurs et des données. L'email de notification identifie les détails tels que le nom de la machine virtuelle, le type d'incident, la date et l'heure de l'incident, ainsi que les actions entreprises en réponse à cet incident.



Logs du Serveur

Les logs du serveur sont essentiels pour le diagnostic des problèmes. Ils fournissent un enregistrement détaillé de tous les événements du serveur, ce qui est crucial pour l'analyse post-incident.

9. Management

Gestion Proactive des Défis Infrastructurels

Au cours du développement de mon projet, une gestion efficace a été cruciale pour maintenir l'avancement et la qualité. Un exemple pertinent de cette gestion a été l'approche proactive face à un défi majeur concernant notre infrastructure cloud.

Mon manager a identifié un problème avec notre abonnement actuel à Azure, où les coûts dépassaient le budget prévu et des dysfonctionnements impactaient ma productivité. Une communication ouverte a été établie avec l'équipe, sollicitant des retours et suggestions. En réponse, une proposition détaillée a été faite pour migrer nos services vers MySQL, une solution plus rentable et stable.

L'échange entre le manager et moi-même (voir capture d'écran ci-jointe) illustre comment une discussion constructive a conduit à la validation de cette nouvelle stratégie, marquant un tournant dans notre gestion des ressources.

Cher Jean-Paul,

J'espère que ce message te trouve en pleine forme. Je souhaitais t'informer d'une situation concernant notre abonnement à Azure et j'aimerais avoir ton avis éclairé sur la question.

Tout d'abord, le coût de l'abonnement à Azure s'avère plus élevé que prévu. Cela commence à peser sur notre budget et il serait judicieux de trouver une solution pour réduire ces dépenses.

De plus, nous avons rencontré plusieurs bugs sur la plateforme Azure ces derniers temps. Ces dysfonctionnements nous ont causé quelques désagréments et ont ralenti notre productivité.

Je t'écris donc pour te demander si tu as des idées ou des suggestions pour résoudre ces problèmes. Peut-être connais-tu des alternatives à Azure qui seraient moins coûteuses et plus stables ? Ou peut-être as-tu des conseils pour optimiser notre utilisation d'Azure de manière à réduire les coûts et à éviter les bugs ?

J'attends avec impatience tes propositions et je te remercie par avance pour ton aide.

Bien cordialement.

Cher manager,

Je te remercie pour ton message et pour m'avoir fait part de tes préoccupations concernant notre utilisation d'Azure. J'ai effectivement constaté les problèmes que tu as mentionnés et je suis d'accord avec toi sur le fait que nous devons trouver une solution.

Après avoir étudié la question, je te suggère de migrer vers MySQL, une base de données open source qui offre de nombreux avantages par rapport à Azure.

Tout d'abord, MySQL est gratuit, ce qui signifie que nous n'aurons plus à payer de coûts d'abonnement mensuels. De plus, MySQL est très stable et fiable, ce qui devrait réduire considérablement les bugs et les temps d'arrêt que nous avons connus avec Azure.

En outre, MySQL est très évolutif et peut gérer de grands volumes de données, ce qui en fait une solution idéale pour notre entreprise en croissance. MySQL est également compatible avec de nombreux langages de programmation, ce qui facilitera l'intégration avec nos autres outils et applications.

Enfin, MySQL dispose d'une grande communauté d'utilisateurs et de développeurs, ce qui signifie que nous pourrions trouver facilement de l'aide et des ressources en ligne en cas de besoin.

En résumé, je pense que la migration vers MySQL est une solution judicieuse pour réduire les coûts, améliorer la stabilité et la fiabilité de notre infrastructure de données, et faciliter notre croissance future.

J'espère que cette suggestion te sera utile. N'hésite pas à me faire part de tes réflexions et à me poser des questions si tu en as.

Bien cordialement,

Jean-Paul

Cher Jean-Paul,

Merci pour ta suggestion concernant la migration vers MySQL. Après avoir pris connaissance des avantages que cette solution offre, je suis convaincu que c'est la meilleure option pour nous.

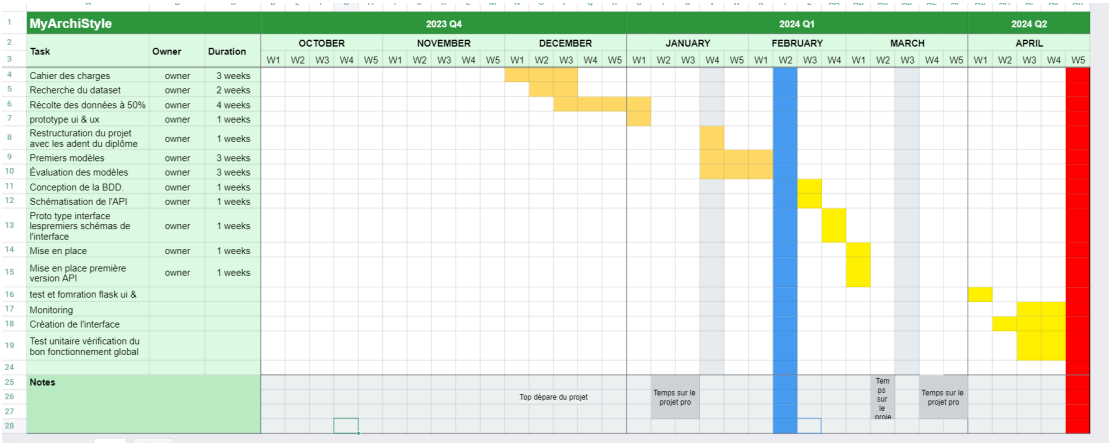
Je te donne donc ma validation pour utiliser MySQL en remplacement d'Azure.

Merci encore pour ton aide précieuse dans cette affaire.

Bien cordialement,

Suivi et Collaboration à l'aide d'Outils de Gestion de Projet

En complément à ces échanges, un diagramme de Gantt a été mis en place pour suivre le progrès des différentes tâches du projet, allant de la recherche du dataset à l'évaluation des modèles (voir capture d'écran du diagramme de Gantt ci-jointe). Ce diagramme a permis de visualiser clairement le planning, les délais et les responsabilités, garantissant que je reste aligné et concentré sur les objectifs.



Ma capacité à collaborer et à ajuster rapidement le plan de projet a démontré l'importance d'une communication efficace et d'une gestion de projet agile, qui sont des éléments clés du succès dans un environnement technologique en constante évolution.

AVRIL W3		
a faire	en cours	terminer
Vérifier les tests unitaires sur la sauvegarde l'image	X	
Léger décalage CSS portable entre le texte et la prédictio	X	X
Refaire la connexion déconnexion Code mauvais		
Transformer les images en .webp pour l'inteface	X	
Corriger les fautes d footeur	X	X
Bug de transparence mobile connexion overlay	X	

10. modèles

10.1 from Scratch

Le modèle initial conçu from scratch est structuré comme une architecture séquentielle, utilisant la bibliothèque TensorFlow Keras, visant à classer des images. Le modèle se compose de couches conçues pour extraire les caractéristiques puis pour effectuer la classification.

Architecture du Modèle:

L'architecture comporte plusieurs couches, organisées comme suit :

- **Couche de Convolution 2D:** Avec 64 filtres de dimension (3x3), la fonction d'activation 'ReLU' et un 'padding' identique pour conserver la dimension spatiale des images. La couche d'entrée accepte des images de taille (28, 28, 1), correspondant à des images en niveaux de gris de 28x28 pixels.
- **Couche de Max Pooling 2D:** Cette couche réduit la dimension spatiale de moitié (réduction 2x2) tout en conservant les caractéristiques les plus saillantes.
- **Deuxième Couche de Convolution 2D:** Également avec 64 filtres et une taille de noyau (3x3) utilisant 'ReLU' comme fonction d'activation.
- **Deuxième Couche de Max Pooling 2D:** Réduit à nouveau les dimensions spatiales.
- **Couche d'Aplatissement (Flatten):** Cette couche transforme la matrice 2D des caractéristiques extraites en un vecteur 1D pour permettre la classification.
- **Couche Dense:** Avec 128 unités et 'ReLU' comme fonction d'activation pour interpréter les caractéristiques.
- **Couche de Sortie Dense:** La dernière couche, avec 10 unités correspondant aux classes à prédire, utilise la fonction d'activation 'softmax' pour obtenir des probabilités de classe.

Performances:

Le modèle entraîné a abouti à une précision (accuracy) de classification de 0.67. Cette performance est considérée comme sous-optimale pour des tâches de classification d'images, indiquant potentiellement un besoin d'amélioration dans la conception du modèle, le prétraitement des données ou les techniques d'augmentation de données pour améliorer les capacités d'apprentissage du réseau.

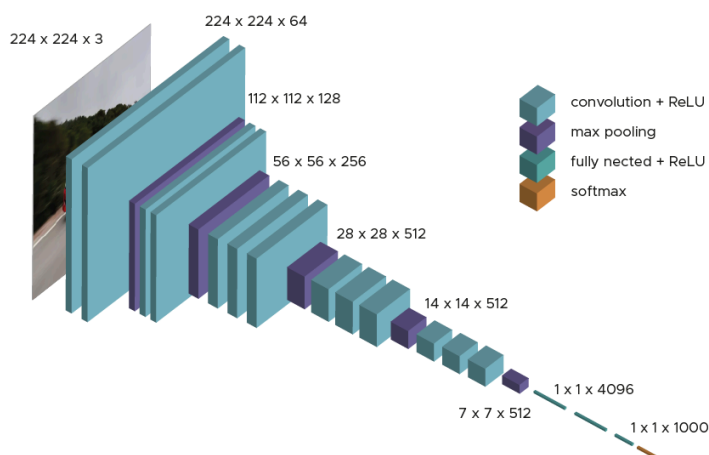
Suite à ce premier modèle, les points suivants ont été améliorés ou ajoutés. Pour augmenter la précision du modèle, j'ai envisagé les stratégies suivantes :

- **Augmentation de Données:** Appliquer des transformations aléatoires aux images d'entraînement pour rendre le modèle plus robuste.
- **Réglage des Hyperparamètres:** Expérimenter avec différents nombres de filtres, tailles de noyau, et taux d'apprentissage.
- **Architectures Plus Profondes:** Ajouter plus de couches de convolution et envisager des blocs résiduels pour permettre au modèle d'apprendre des caractéristiques plus complexes.
- **Utilisation de Modèles Pré-Entraînés :** Incorporer une architecture de modèle pré-entraînée comme Resnet50, InceptionResnetV2 ou VGG-19 qui a été largement validée pour des tâches de classification d'images. Utiliser VGG16 comme point de départ, avec ses poids entraînés sur la base de données ImageNet, peut non seulement accélérer la convergence mais aussi améliorer la précision du modèle grâce à la transférabilité des caractéristiques apprises. Il est possible de personnaliser les couches supérieures de VGG16 pour les adapter au nombre spécifique de classes dans notre tâche de classification.

10.2 Utilisation de VGG16 un Modèles Pré-Entraînés

VGG16 est un modèle de réseau de neurones convolutifs (CNN) qui a été introduit en 2014 par Karen Simonyan et Andrew Zisserman de l'Université d'Oxford dans le cadre du challenge ImageNet. Le nom "VGG" vient de Visual Geometry Group, le laboratoire où il a été développé, et le "16" fait référence au nombre de couches avec des poids dans le réseau, ce qui le rend assez profond pour son époque.

Le modèle se caractérise par son architecture uniforme, utilisant principalement des couches convolutives de petite taille (3x3) avec des strides de 1 pixel et en utilisant le même padding pour conserver les dimensions spatiales des images à travers les couches. Après certaines couches convolutives, le modèle utilise des couches de pooling max pour réduire la dimensionnalité. Cette stratégie permet au réseau d'apprendre des caractéristiques complexes des images en augmentant la profondeur tout en conservant une certaine contrainte sur le nombre de paramètres grâce à l'utilisation de filtres de petite taille.



VGG16 a été l'un des premiers modèles à démontrer que la profondeur du réseau est un facteur clé pour améliorer la performance dans les tâches de reconnaissance visuelle. Il a obtenu de très bons résultats sur plusieurs benchmarks, y compris le challenge ImageNet, où il a été utilisé pour classer des images dans 1000 catégories différentes.

Bien que VGG16 soit considéré comme un modèle assez simple et direct par les standards actuels, son architecture a eu une grande influence sur le développement de modèles de réseaux de neurones convolutifs ultérieurs. Toutefois, en raison de sa profondeur et de la taille de son architecture, il est assez gourmand en ressources, ce qui peut poser des problèmes pour l'entraînement et l'inférence sur des appareils avec des capacités de calcul limitées. Malgré cela, il reste une pierre angulaire dans le domaine de la vision par ordinateur et un excellent point de départ pour comprendre les CNNs.

source:

<https://arxiv.org/abs/1409.1556>

<https://datascientest.com/quest-ce-que-le-modele-vgg>

10.3 vgg16 transfer learning:

Le transfer learning est une méthode puissante en apprentissage profond qui implique l'utilisation d'un modèle pré-entraîné sur un grand ensemble de données et son adaptation à un problème similaire ou connexe. Pour notre projet, nous avons sélectionné le réseau VGG16, bien connu pour sa simplicité et sa robustesse, pré-entraîné sur la vaste base de données ImageNet qui contient plus d'un million d'images réparties en 1000 classes.

Intégration de VGG16 dans l'Architecture du Modèle

Le modèle VGG16 a été initialisé avec ses poids entraînés sur ImageNet et toutes ses couches ont été marquées comme non entraînables (``trainable=False``). Cela permet de fixer les caractéristiques extraites par le modèle et de les utiliser comme points de départ pour notre tâche spécifique.

Définition de la Topologie Personnalisée:

- **Entrée** : une couche d'entrée est définie pour accepter les images de dimension (224, 224, 3), ce qui est nécessaire pour correspondre à la première couche de convolution de VGG16.
- **Application de VGG16**: les entrées sont passées à travers le modèle VGG16 configuré préalablement, en maintenant le paramètre ``training`` sur ``False`` pour éviter des changements dans les poids lors de l'inférence.
- **Flatten Layer**: Les caractéristiques extraites sont ensuite aplaties en un vecteur pour permettre le traitement par des couches entièrement connectées.
- **Couches Denses**: Après aplatissage, une couche dense de 128 neurones avec l'activation 'ReLU' est ajoutée pour introduire la capacité d'apprendre des relations non linéaires et complexifier l'interprétation des caractéristiques. La sortie de cette couche est ensuite passée à une couche dense avec une seule unité et une activation 'sigmoid', ce qui est typique pour la classification binaire.

Techniques d'Optimisation et Callbacks

- **Optimisation:** L'optimiseur RMSprop est utilisé avec un taux d'apprentissage fixé à 0.001, un choix stratégique pour une convergence stable étant donné le faible taux d'apprentissage et l'efficacité prouvée de RMSprop dans le traitement des gradients.
- **-Fonction de Perte et Métriques:** La fonction de perte 'binary_crossentropy' est utilisée conjointement avec la métrique 'accuracy' pour la compilation du modèle. Cette configuration est standard pour les tâches de classification binaire où l'objectif est de minimiser la probabilité d'erreur entre les prédictions et les véritables étiquettes.
- **Callbacks pour la Préservation et l'Optimisation du Modèle:**
 - **ModelCheckpoint:** Un callback 'ModelCheckpoint' est instauré pour enregistrer automatiquement les poids du modèle dans un fichier 'aug_model.h5' chaque fois qu'une amélioration de la précision de validation est observée. Cela garantit que nous conservons la meilleure version du modèle tout au long du processus d'entraînement.
 - **EarlyStopping:** Pour éviter le surajustement, un callback 'EarlyStopping' est mis en place pour interrompre l'entraînement si aucune augmentation de la précision de validation n'est observée pendant 6 époques consécutives. De plus, le callback est configuré pour restaurer les poids du meilleur modèle en cas d'arrêt précoce, s'assurant ainsi que nous utilisons la version la plus optimisée du modèle pour les prédictions.

Maintenant que nous avons vue l'architecture de notre modèle, passons en revue sa mise en œuvre réelle. Voici le code qui définit notre modèle de transfer learning utilisant VGG16 :

```
# Définition de l'entrée du modèle avec une forme de (224, 224, 3), qui correspond à une image de
224x224 pixels avec 3 canaux de couleur (RGB)
inputs = tf.keras.Input(shape=(224, 224, 3))

# Utilisation du modèle pré-entraîné sur les entrées, avec le paramètre training défini sur False pour
utiliser le modèle en mode inférence
x = pretrained_model(inputs, training=False)

# Aplatissement de la sortie du modèle pré-entraîné en un vecteur unidimensionnel pour pouvoir
l'utiliser dans les couches denses suivantes
# La forme de la sortie est (7, 7, 512), donc après aplatissement, la forme devient (7 * 7 * 512,)
x = flatten_layer(x) # 7*7*512

# Ajout d'une couche dense avec 128 unités et une fonction d'activation ReLU pour introduire de la
non-linéarité
x = tf.keras.layers.Dense(128, activation='relu')(x)

# Ajout d'une couche dense de sortie avec une unité et une fonction d'activation 'softmax' pour obtenir
une probabilité de 10 classe
outputs = tf.keras.layers.Dense(10, activation='softmax')(x)
```

```
# Définition du modèle complet avec les entrées et les sorties définies précédemment  
model = tf.keras.Model(inputs, outputs)
```

Ce code définit le modèle complet, de l'entrée des images à la sortie binaire. Le modèle utilise le modèle VGG16 pré-entraîné pour extraire des caractéristiques des images, puis passe ces caractéristiques à travers une couche dense avec une activation ReLU pour introduire de la non-linéarité, et enfin à travers une couche dense de sortie avec une activation sigmoïde pour obtenir une probabilité de classe binaire.

Performances:

En utilisant cette méthode de transfer learning avec VGG16, le modèle a atteint des résultats prometteurs, révélant une perte de test de 0.2165 et une précision de 0.9160. Ces indicateurs mettent en évidence l'efficacité de l'approche adoptée, où les couches de VGG16 pré-entraînées servent de base solide pour la classification d'images, tandis que les couches personnalisées ajoutées par dessus permettent un ajustement fin et une spécialisation pour notre tâche spécifique.

10.3 vgg16 Fine Tuning:

Le Fine Tuning est une technique de transfer learning avancée qui consiste à ajuster les poids d'un modèle pré-entraîné pour les adapter à une tâche spécifique. Cette section détaille le processus de Fine Tuning appliqué au modèle VGG16 dans le but d'améliorer davantage la classification d'images spécifique à notre ensemble de données.

Structure du Modèle

Le modèle VGG16 utilisé pour le Fine Tuning se présente avec la configuration suivante :

- **Couches de Convolution (blocks 1 à 5)** : Comprenant plusieurs couches Conv2D et MaxPooling2D. Chaque "bloc" a pour but d'extraire des caractéristiques à différents niveaux de granularité, augmentant la profondeur du modèle pour capter une variété de motifs et de textures dans les images.

- **Nombre Total de Paramètres** : Le modèle comprend 14,714,688 paramètres, tous non-entraînaibles dans cette phase initiale pour préserver les caractéristiques extraites de la base de données ImageNet.

Configuration d'Entraînement

- **Callbacks** :
 - **Model Checkpoint** : Sauvegarde le meilleur modèle en termes de précision de validation (*val_accuracy*) dans le fichier *aug_model.h5*.
 - **EarlyStopping** : Surveille la précision de validation et interrompt l'entraînement si aucune amélioration n'est constatée pendant 16 époques consécutives, tout en restaurant les meilleurs poids obtenus.

Compilation :

- **Optimiseur** : RMSprop avec un taux d'apprentissage de 0.001

Cette configuration est optimale pour un ajustement fin sur les nouvelles données tout en maintenant la stabilité du processus d'apprentissage.

- **Fonction de Perte** : *'binary_crossentropy'*, adaptée aux problèmes de classification binaire où notre objectif est de prédire deux classes distinctes.

- **Métriques:** '*accuracy*' pour mesurer la proportion des prédictions correctes par rapport aux étiquettes réelles.

Entraînement du Modèle

L'entraînement a été réalisé avec les paramètres suivants :

- **Données:** Utilisation de générateurs ('*train_generator*' et '*test_generator*') qui fournissent les images de manière itérative pour l'entraînement et la validation, permettant une gestion efficace de la mémoire et l'introduction de l'augmentation de données si nécessaire.
- **Nombre d'Époques:** Le modèle a été entraîné pour 50 époques, avec un arrêt anticipé en fonction de l'amélioration de la précision de validation.
- **Callbacks:** Utilisation des callbacks '*ModelCheckpoint*' pour enregistrer le meilleur modèle et '*EarlyStopping*' pour prévenir le surajustement.

Conclusion et Performances

Après l'application du Fine Tuning au modèle VGG16, les performances ont montré une perte (loss) de 0.3286 et une précision (accuracy) de 0.8862. Ces résultats indiquent que le modèle ajusté est bien adapté à la tâche spécifique de classification, démontrant la valeur ajoutée du Fine Tuning sur les modèles pré-entraînés dans des contextes spécifiques.

10.4 Sélection du modèle

Modèle	Profondeur	Exactitude
from Scratch	3	67%
<u>transfer learning</u>	<u>18</u>	<u>91%</u>
Transfer Learning (no data augmentation)	18	85%
Fine Tuning	18	88%
Fine Tuning (no data augmentation)	18	81%

Le choix du modèle dépend de plusieurs facteurs:

Précision: Le modèle de Transfer Learning a la meilleure précision (91%). Le Fine Tuning offre une légère amélioration (88%) mais peut être plus complexe à mettre en œuvre.

Complexité: Le modèle from Scratch est le plus simple, mais sa précision est moindre. Le Transfer Learning est plus complexe mais offre une meilleure précision. Le Fine Tuning est le plus complexe et peut ne pas justifier l'amélioration de la précision obtenue.

Ressources disponibles: Le modèle from Scratch nécessite moins de ressources que les autres. Le Transfer Learning et le Fine Tuning nécessitent plus de puissance de calcul et de mémoire.

Objectifs du projet: Si la précision est la priorité absolue, le Transfer Learning est le meilleur choix. Si la simplicité est le facteur le plus important, le modèle from Scratch est suffisant. Le Fine Tuning peut être un bon compromis entre précision et complexité.

Autres facteurs à prendre en compte:

Taille du jeu de données: Un modèle plus complexe peut être nécessaire pour un jeu de données volumineux.

Disponibilité de données prétraitées: Le Transfer Learning et le Fine Tuning peuvent s'appuyer sur des données prétraitées.

Expérience et compétences: Le choix du modèle dépend également de l'expérience et des compétences du développeur.

Conclusion:

Le choix du modèle est une décision importante qui doit être prise en fonction des besoins spécifiques du projet. Il est important de comparer les différentes options et de choisir celle qui offre le meilleur compromis entre précision, complexité et ressources disponibles.

En plus des facteurs mentionnés ci-dessus, il est important de prendre en compte la possibilité d'erreurs systématiques dans les données. Si les données sont biaisées, le modèle risque de l'être également. Il est donc important de tester le modèle sur un ensemble de données indépendant pour s'assurer qu'il est généralisable.

Enfin, il est important de noter que le choix du modèle n'est pas la seule étape importante dans le processus de développement d'un modèle d'apprentissage automatique. Il est également important de prétraiter les données, de choisir les bonnes métriques d'évaluation et d'optimiser les hyperparamètres du modèle.

11. Axe d'amélioration

Afin de continuer à innover et à fournir une expérience utilisateur exceptionnelle, plusieurs axes d'amélioration ont été identifiés. Ces améliorations visent à augmenter la précision analytique de notre système, enrichir les fonctionnalités de notre plateforme et maximiser la pertinence des contenus pour nos utilisateurs. Voici les domaines clés d'intervention

Détection de Contexte dans les Images

Un axe crucial d'amélioration réside dans la mise au point d'un système sophistiqué de reconnaissance de contexte. Cet outil analysera les éléments visuels et conceptuels des images soumises, en utilisant des techniques avancées de vision par ordinateur et d'intelligence artificielle pour évaluer leur adéquation avec les normes d'utilisation de l'application. En s'assurant que les images contribuent de manière significative à la base de données, nous maintenons un haut niveau de qualité et de pertinence des données. Ceci est essentiel pour la précision du modèle et pour offrir une expérience utilisateur cohérente.

Réutilisation des Images Utilisateurs

L'implémentation d'une plateforme de stockage sécurisée et éthique pour les images utilisateurs est primordiale. Cette initiative soutiendra le principe du continuous training, où le modèle s'auto-perfectionne en apprenant de nouvelles données. Cette réutilisation des images nécessite une attention particulière à la vie privée des utilisateurs et au consentement pour l'utilisation des données, conformément aux réglementations telles que le RGPD.

Fonctionnalités de Collaboration et Partage de Groupes d'Images

Pour enrichir l'expérience communautaire, nous envisageons de développer des fonctionnalités collaboratives permettant aux utilisateurs de constituer et partager des collections d'images avec d'autres membres de la plateforme. Cela pourrait inclure des outils de commentaires, de notation et de recommandations qui non seulement stimuleront l'engagement des utilisateurs mais fourniront également des données précieuses pour affiner notre modèle.

Extension de la Collection d'Art Architectural

Nous projetons d'élargir notre répertoire d'œuvres architecturales au sein de l'application. Cela impliquera une collaboration étroite avec des historiens de l'art, des architectes et des photographes pour collecter et cataloguer un éventail plus large de styles architecturaux. L'objectif est de diversifier les prédictions du modèle et d'offrir une ressource éducative plus riche aux utilisateurs.

Création de Rapports Analytiques et Graphiques automatique

La capacité pour les utilisateurs de générer des rapports personnalisés est une autre amélioration envisagée. Cette fonctionnalité intégrera des outils d'analyse visuelle pour créer des graphiques et des résumés à partir de leurs ensembles d'images. Les utilisateurs pourront ainsi obtenir des insights significatifs, des tendances et des résumés qui pourraient être utilisés à des fins de présentation ou d'étude.

Expérimentation avec Divers Modèles Pré-Entraînés

Enfin, une exploration méthodique de modèles pré-entraînés alternatifs tels que ResNet50 ou InceptionResNetV2 sera entreprise pour évaluer leur performance par rapport à notre ensemble de données spécifique. Des benchmarks rigoureux seront effectués pour comparer les taux de précision, les temps de réponse et la consommation de ressources. Le choix final du modèle sera basé sur une combinaison de performance technique et de facilité d'intégration dans notre infrastructure existante.