Democratic and Popular Republic of Algeria
Ministry of Higher Education and Scientific Research
Aboubekr Belkaid University – Tlemcen –
Faculty of Science
Department of Computer Science

Projet de fin d'études

Pour l'obtention du diplôme de Master

Domaine : Informatique

Spécialité : Intelligence Artificielle

Sujet :

Réponse visuelle aux questions(VQA) à l'aide de réseaux neuronaux récurrents et de réseaux neuronaux convolutifs

Présenté par :
Mlle TAHRAOUI Nour El Houda

Encadré par :
Mr HADJILA Fethallah

Examinateurs :
Mr BELABED Amine (Président)
Mme AMGHAR Djazia (Examinatrice)

Année universitaire : 2024 / 2025

Remerciements

Avant toute chose, je rends grâce à Dieu Tout-Puissant, source de force, de persévérance et de sagesse, qui m'a guidée et soutenue tout au long de cette aventure académique.

Je tiens à exprimer ma profonde gratitude à mon encadrant, **Mr Hadjila Fethallah**, pour sa disponibilité, sa patience et la richesse de ses conseils. Son accompagnement rigoureux et bienveillant a été essentiel à la réussite de ce travail.

Je remercie chaleureusement les membres du jury pour l'honneur qu'ils m'ont fait en acceptant d'évaluer ce mémoire, et pour l'attention et l'intérêt qu'ils ont accordés à ce projet.

Ma reconnaissance s'adresse également à l'ensemble des enseignants du département d'informatique, pour la qualité de leur enseignement et leur implication tout au long de mon parcours universitaire.

Enfin, je remercie de tout cœur ma famille, mes amis, et toutes les personnes qui, de près ou de loin, ont apporté leur soutien moral ou matériel à la réalisation de ce mémoire. Leur présence et leurs encouragements ont été pour moi une source constante de motivation.

Dédicace

Je dédie ce modeste travail, avant toute chose,
à la mémoire de mon cher père, en priant Dieu de lui accorder Sa miséricorde
et de l'accueillir dans Son vaste paradis.

À ma chère mère, source d'amour, de patience et de force,
pour son soutien inconditionnel et ses prières sincères.

À toute ma famille,
pour leur encouragement constant tout au long de mon parcours.

À mes amies fidèles,
qui ont toujours été un soutien précieux et une source de motivation.

Je tiens également à exprimer ma profonde reconnaissance à
Mrs **Rafika Benledghem**,
pour son aide précieuse, son soutien moral
et les encouragements qu'elle m'a apportés tout au long de la réalisation de ce mémoire.

Résumé

Ce travail de fin d'études porte sur la conception et l'évaluation d'un système de *Visual Question Answering* (VQA), une tâche multimodale combinant vision par ordinateur et traitement du langage naturel. L'objectif est de permettre à un modèle d'apprentissage profond de répondre à des questions en langage naturel à partir d'images contenant des informations complexes. Pour cela, une architecture bimodale a été proposée, combinant un réseau de neurones convolutifs (CNN) pour le traitement des images, et un réseau LSTM pour l'analyse des questions textuelles.

Trois architectures CNN ont été testées : ResNet18, MobileNetV2, et EfficientNet-B0/B1, en utilisant le jeu de données CLEVR. Les performances ont été évaluées à l'aide de métriques telles que la précision, la fonction de perte, les courbes d'apprentissage et les matrices de confusion. Les résultats ont montré des performances comparables entre les modèles, avec un léger avantage pour ResNet18 en termes de précision.

Ce travail contribue au développement des systèmes VQA et ouvre la voie à l'utilisation de modèles plus avancés pour améliorer les performances dans les tâches multimodales.

Mots-clés : Visual Question Answering, apprentissage profond, réseaux convolutifs (CNN), LSTM, fusion multimodale, CLEVR.

ملخص

يهدف هذا المشروع إلى تطوير نظام للإجابة على الأسئلة البصرية ،(VQA) وهي مهمة متعددة الوسائط تجمع بين الرؤية الحاسوبية ومعالجة اللغة الطبيعية. الهدف هو تمكين نموذج من التعلم العميق من الإجابة على أسئلة بلغة طبيعية اعتمادًا على صور تحتوي على معلومات منطقية معقدة. لتحقيق ذلك، تم اقتراح بنية مزدوجة تتضمن شبكة عصبية التفافية (CNN) لمعالجة الصور، وشبكة LSTM لتحليل الأسئلة النصية.

تم استخدام ثلاث معماريات CNN هي ResNet18 وMobileNetV2 وEfficientNet-B0/B1، وذلك باستخدام مجموعة بيانات .CLEVR وتم تقييم الأداء باستخدام مؤشرات مثل الدقة، والخسارة، ومنحنيات التعلم، ومصفوفات الالتباس. أظهرت النتائج تقاربًا في الأداء بين النماذج، مع تفوق طفيف لـ ResNet18 من حيث الدقة.

يساهم هذا العمل في تطوير أنظمة VQA ويقترح إمكانية استخدام نماذج أكثر تقدمًا لتحقيق أداء أفضل في المهام متعددة الوسائط.

الكلمات المفتاحية : الإجابة على الأسئلة البصرية، التعلم العميق، الشبكات العصبية الالتفافية، ،LSTM الدمج متعدد الوسائط، .CLEVR

Abstract

This project focuses on the design and evaluation of a Visual Question Answering (VQA) system, a multimodal task that integrates computer vision and natural language processing. The goal is to enable a deep learning model to answer natural language questions based on complex images. We propose a bimodal architecture that combines a convolutional neural network (CNN) for image encoding and an LSTM-based encoder for textual questions.

We experimented with three CNN architectures : ResNet18, MobileNetV2, and EfficientNet-B0/B1, using the CLEVR dataset, which offers synthetic scenes with logical reasoning challenges. Performance was evaluated using accuracy, loss, learning curves, and confusion matrices. The results showed comparable performance across models, with ResNet18 achieving slightly higher accuracy.

This work contributes to the development of VQA systems by implementing a hybrid image-text approach and highlights the potential of more advanced models for future multimodal reasoning tasks.

Keywords : Visual Question Answering, deep learning, convolutional neural networks (CNN), LSTM, multimodal fusion, CLEVR.

Table des matières

Table des figures

Liste des abréviations

VQA Visual Question Answering

IA Intelligence Artificielle

DL Deep Learning

ML Machine Learning

CNN Convolutional Neural Network

LSTM Long Short-Term Memory

NLP Natural Language Processing

CV Computer Vision

GPU Graphics Processing Unit

ReLU Rectified Linear Unit

SGD Stochastic Gradient Descent

Adam Adaptive Moment Estimation

ViT Vision Transformer

FC Fully Connected (Layer)

CLEVR Compositional Language and Elementary Visual Reasoning (Dataset)

JSON JavaScript Object Notation

API Application Programming Interface

TPU Tensor Processing Unit

1 Introduction

1.1 Introduction

Le Visual Question Answering (VQA) est un domaine clé de l'IA multimodale combinant vision par ordinateur et traitement du langage naturel pour répondre à des questions sur des images. S'appuyant sur des modèles de deep learning comme les CNN [He et al., 2016] pour l'analyse visuelle et les transformers [Vaswani et al., 2017] pour le langage, il vise à reproduire un raisonnement contextuel humain [Antol et al., 2015]. Cependant, des défis persistent : biais de modalité (les modèles privilégient le texte plutôt que l'image,[Goyal et al., 2017] et nécessité d'inférences complexes. Malgré ces limites, le VQA offre des applications prometteuses en santé, éducation ou accessibilité.

1.2 Contexte

Le Deep Learning (DL) a révolutionné la vision par ordinateur grâce aux architectures comme les CNN [Krizhevsky et al., 2012], permettant la détection d'objets ou la segmentation de scènes. Des modèles avancés comme ResNe[He et al., 2016] ou YOLO dominent aujourd'hui les tâches comme la classification d'images ou la détection d'objets en temps réel.

Le DL a aussi transformé le Traitement Automatique du Langage Naturel (TALN), notamment via les transformers [Vaswani et al., 2017]. Des variantes comme BERT [Devlin et al., 2018] ou GPT permettent de générer et comprendre du texte de manière fluide, en tenant compte du contexte global.

1.3 Motivation

A. Synergie Multimodale

Le VQA tire profit de la synergie entre vision par ordinateur, TALN et DL. Cette combinaison permet un raisonnement sur des données visuelles et textuelles, en relevant les défis d'alignement et de fusion multimodale [Baltrušaitis et al., 2018] .

B. Applications Transversales

— **Médical :** assister au diagnostic via l'analyse conjointe d'images et de questions cliniques.

— **Éducation :** tutoriels interactifs pour expliquer des graphiques ou illustrations.

— **Robotique :** interaction contextuelle entre robots et environnement.

C. Défis Scientifiques

— **Fusion de modalités :** combiner efficacement les représentations visuelles et textuelles.

— **Robustesse :** éviter les biais de modalité, où le modèle se base trop sur le texte.

L'approche d'[Anderson et al. 2018] combine attention bottom-up et top-down pour améliorer cette fusion.

1.4 Problématique

Malgré ses avancées, le VQA reste limité face aux scènes ambiguës, aux relations implicites (causalité, intention) et aux raisonnements complexes. Les biais modaux persistent (Kafle & Kanan, 2017). Le défi est de concevoir des architectures capables d'intégrer équitablement les modalités visuelle et textuelle, avec une compréhension plus intuitive, pour des domaines comme la santé ou l'éducation personnalisée.

À cela s'ajoute le défi de l'occlusion des objets dans les images, qui complique davantage l'analyse visuelle et la correspondance avec la question posée. Une partie cruciale de l'information peut être cachée, rendant la tâche de raisonnement encore plus difficile, en particulier lorsque le modèle doit inférer la présence ou la fonction d'un objet partiellement ou totalement invisible.

1.5 Contributions

Pour traiter la problématique du Visual Question Answering (VQA), nous avons exploré trois approches combinant traitement d'images et compréhension du langage naturel.

Approche 1 : CNN from scratch + LSTM

Dans cette approche, nous avons conçu un réseau CNN simple à partir de zéro pour extraire les caractéristiques des images. Les questions sont encodées via un LSTM, puis fusionnées avec les données visuelles pour prédire la réponse.

Approche 2 : MobileNetV2 + LSTM

Nous utilisons ici MobileNetV2, un modèle léger pré-entraîné, comme extracteur de caractéristiques. Le LSTM traite les questions, et les deux sorties sont combinées pour générer la réponse. Cette approche vise un bon compromis entre performance et efficacité.

Approche 3 : EfficientNet-B0 + LSTM

Cette dernière approche exploite EfficientNet-B0, connu pour son excellent rapport précision/complexité. Couplé à un LSTM pour la compréhension des questions, il permet d'atteindre des performances élevées sur des tâches VQA complexes.

1.6 Plan du manuscrit

Chapitre 2 : Présentation des concepts théoriques liés au Visual Question Answering (VQA) et des principales architectures utilisées.

Chapitre 3 : Description de l'implémentation expérimentale du système VQA, des modèles testés et de l'analyse des résultats.

Conclusion : Synthèse des travaux menés et ouverture sur des perspectives futures.

2 Chapitre 2 : Deep Learning

2.1 Introduction à l'apprentissage profond

L'apprentissage profond (*deep learning*) est une sous-branche de l'apprentissage automatique (*machine learning*), lui-même inclus dans le champ plus large de l'intelligence artificielle (IA) [Russell et Norvig, 2020]. Le *machine learning* permet aux systèmes d'apprendre à partir de données pour résoudre des tâches spécifiques, tandis que le *deep learning* repose sur des réseaux de neurones profonds capables d'extraire automatiquement des représentations hiérarchiques et complexes à partir de données brutes [Goodfellow et al., 2016 ; LeCun et al., 2015].

Contrairement aux approches classiques nécessitant une ingénierie manuelle des caractéristiques, le *deep learning* apprend directement à partir des données, ce qui le rend particulièrement efficace dans des domaines tels que la vision par ordinateur, le traitement du langage naturel et la reconnaissance vocale [LeCun et al., 2015 ; Vaswani et al., 2017].

L'apprentissage profond englobe plusieurs types d'apprentissage :

- **Apprentissage supervisé** : le modèle est entraîné sur des données étiquetées (exemples d'entrée avec sortie attendue). Cette approche est utilisée dans la classification d'images (par exemple : détection de maladies en imagerie médicale avec des CNN [Litjens et al., 2017]) ou la prédiction de séquences (ex. : traduction automatique avec des RNN [Sutskever et al., 2014]).
- **Apprentissage non supervisé** : le modèle analyse des données non étiquetées pour en extraire des structures cachées. Les auto-encodeurs [Hinton et Salakhutdinov, 2006] et les GAN (*réseaux antagonistes génératifs*) [Goodfellow et al., 2014] en sont des exemples, utilisés pour la compression, la génération d'images ou la réduction de dimensions.
- **Apprentissage par renforcement** : un agent interagit avec un environnement et apprend à maximiser une récompense à travers l'expérience. Cette méthode est efficace pour des tâches de décision complexes, comme le jeu de Go (*AlphaGo* [Silver et al., 2016]) ou la navigation autonome de robots [Kober et al., 2013].

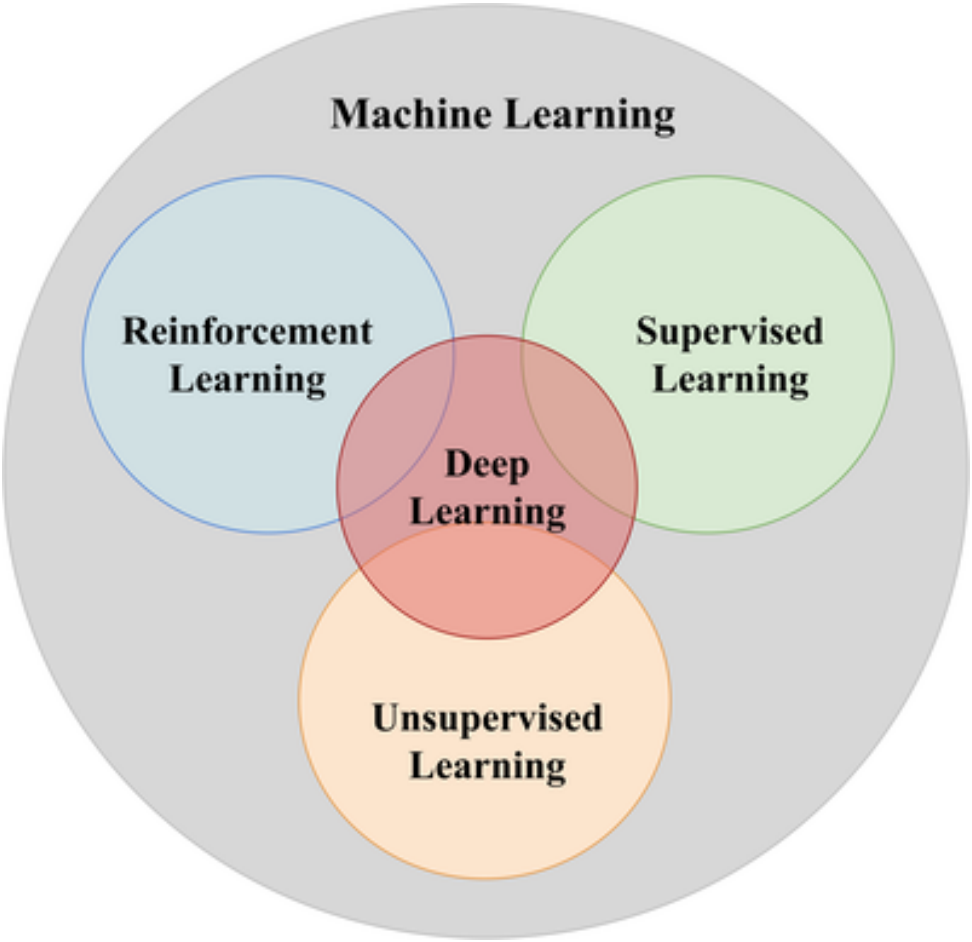


FIG. 2.1 – The main types of machine learning .

2.2 Fondamentaux de l'apprentissage profond

L'apprentissage profond repose sur des réseaux de neurones artificiels organisés en couches et entraînés à l'aide de processus spécifiques. Ces éléments permettent de modéliser des relations complexes et d'extraire des représentations hiérarchiques à partir de grandes quantités de données [Goodfellow et al., 2016].

2.2.1 Réseaux de neurones

A. Neurones artificiels

Un neurone artificiel (ou perceptron) reçoit des entrées $x_1, x_2, \ldots, x_n$, les multiplie par des poids $w_1, w_2, \ldots, w_n$, les additionne avec un biais b, puis applique une fonction d'activation f :

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right)$$

Ce mécanisme simule le comportement des neurones biologiques et constitue l'unité de base des réseaux de neurones [McCulloch et Pitts, 1943 ; Goodfellow et al., 2016].

B. Couches

Les réseaux de neurones sont composés de plusieurs couches :

— **Couche d'entrée** : reçoit les données brutes (ex : image, texte, audio).
— **Couches cachées** : apprennent des représentations intermédiaires et abstraites.
— **Couche de sortie** : produit la prédiction finale.

Plus le nombre de couches cachées est grand, plus le réseau est dit "profond"[LeCun et al., 2015].

C. Fonctions d'activation

Les fonctions d'activation introduisent de la non-linéarité, permettant au réseau de capturer des relations complexes. Les plus courantes sont :

— **ReLU (Rectified Linear Unit)** :

$$f(x) = \max(0, x)$$

Rapide à calculer, elle évite le problème du gradient qui disparaît.

— **Sigmoïde** :

$$f(x) = \frac{1}{1 + e^{-x}}$$

Utilisée en sortie pour des tâches de classification binaire.

— **Tanh (Tangente hyperbolique)** :

$$f(x) = \tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

La sortie est centrée autour de zéro, utile dans les réseaux récurrents.

— **Softmax** :

$$f(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

Utilisée en sortie pour la classification multiclasse.

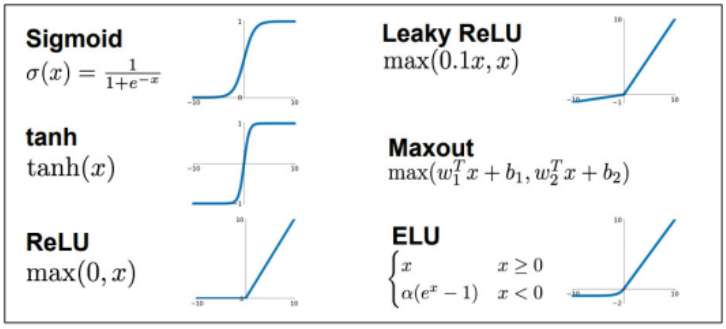


FIG. 2.2 – Activation functions curves .

2.2.2 Processus d'entrainement

A. Propagation avant

Les données passent couche par couche. Chaque neurone calcule sa sortie à partir des entrées reçues, appliquant les poids, biais et la fonction d'activation [Goodfellow et al., 2016].

B. Fonction de perte

La fonction de perte mesure l'écart entre la sortie prédite et la sortie attendue. Exemples :

— **Erreur quadratique moyenne (MSE)** :

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

— **Entropie croissée (Cross-Entropy)** :

$$\mathcal{L} = -\sum_{i} y_i \log(\hat{y}_i)$$

C. Rétropropagation

La rétropropagation ajuste les poids via le gradient de la fonction de perte, calculé à l'aide de la règle de la chaîne. Les algorithmes courants sont :

— SGD (Stochastic Gradient Descent)

— Adam (Adaptive Moment Estimation)

Ces techniques permettent au réseau de converger vers un ensemble de poids qui minimise l'erreur [Rumelhart et al., 1986 ; Kingma et Ba, 2014]

2.3 L'apprentissage profond en vision par ordinateur

Ces études suggèrent que l'apprentissage profond en vision par ordinateur permet d'extraire automatiquement des informations visuelles complexes à partir d'images. Combiné au traitement du langage naturel, il rend possible le *Visual Question Answering* (VQA), où un système répond à des questions sur des images en intégrant et en analysant conjointement les données visuelles et textuelles [Antol et al., 2015].

2.3.1 Réseaux de neurones convolutifs (CNN)

Un réseau de neurones convolutif (CNN) est un type de réseau de neurones profond conçu pour traiter des données structurées en grille, comme les images. Il apprend automatiquement des hiérarchies de caractéristiques à partir des images grâce à des couches spécialisées. Il est devenu la méthode dominante pour de nombreuses tâches de vision par ordinateur, telles que la classification et la détection d'objets [Krizhevsky et al., 2012; LeCun et al., 2015; He et al., 2016].

A. Couches de convolution

Les couches de convolution sont responsables de l'extraction automatique des caractéristiques locales de l'image en appliquant des filtres (ou noyaux) sur l'entrée. Elles produisent des cartes de caractéristiques (*feature maps*) qui capturent des motifs comme les bords, les textures ou des formes plus complexes à mesure que l'on progresse dans les couches [Zeiler et al., 2014].

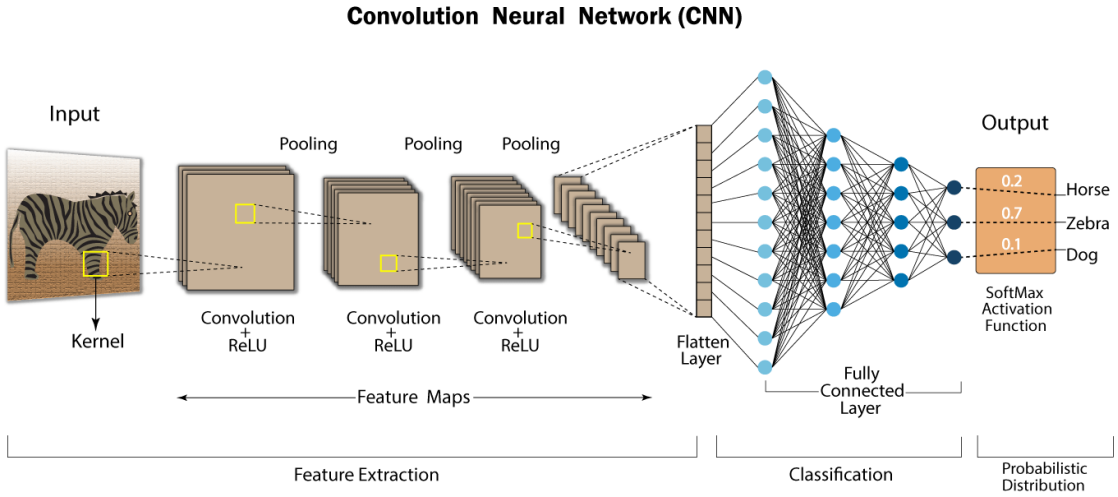


FIG. 2.3 – Architecture typique d'un Réseau de Neurones Convolutif (CNN) pour la classification d'images.

B. Couches de regroupement (pooling)

Les couches de pooling réduisent la dimensionnalité des cartes de caractéristiques en résumant les informations locales (par exemple, via le *max pooling* ou l'*average pooling*). Cela permet de diminuer le nombre de paramètres, de contrôler le surapprentissage et de rendre les représentations plus invariantes aux translations et déformations [Boureau et al., 2010; LeCun et al., 2015].

— **Le pooling max (*max pooling*)** : sélectionne la valeur maximale dans chaque région d'une carte de caractéristiques, mettant en évidence les éléments les plus saillants de l'image. Cette opération est efficace pour préserver les détails critiques, mais elle peut également entraîner une perte d'informations subtiles [Zeiler et al., 2014; Graham, 2014].

— **Le pooling moyenne (*average pooling*)** : calcule la moyenne des valeurs dans chaque région, ce qui permet une représentation plus lissée et globale des caractéristiques, tout en réduisant la perte d'information. Toutefois, cette méthode peut diluer les motifs discriminants essentiels [Scherer et al., 2010; Springenberg et al., 2014].

Des recherches suggèrent que la combinaison de *max* et *average pooling*, ou leur utilisation adaptative selon le contexte, peut améliorer la robustesse et la performance globale des modèles de vision par ordinateur, en capturant des informations complémentaires issues des deux approches[Yu et al., 2014; Lee et al., 2016].

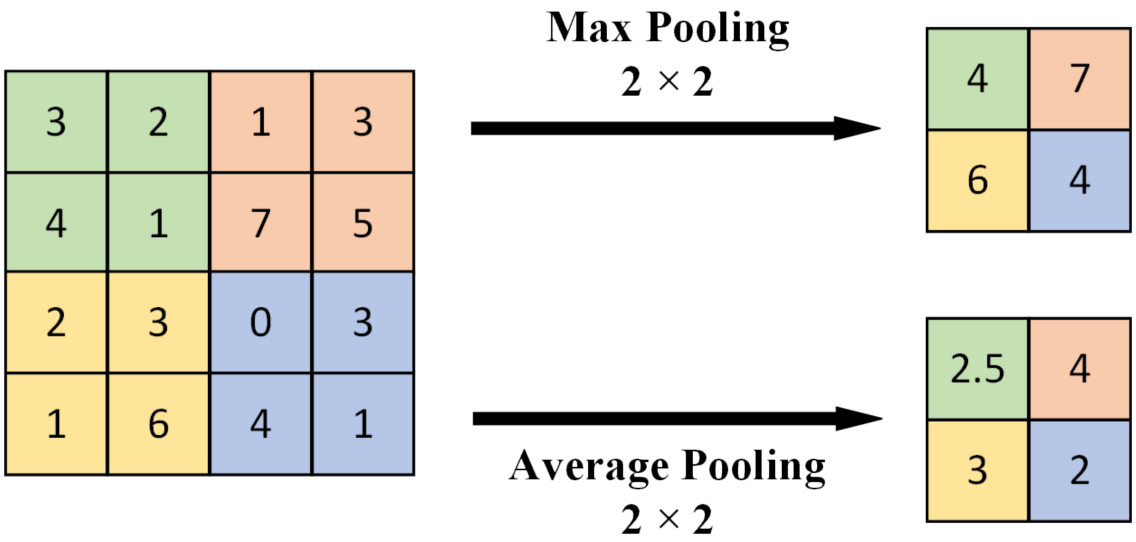


FIG. 2.4 – pooling.

C. Couches entièrement connectées

Les **couches entièrement connectées** (*fully connected layers*) relient chaque neurone d'une couche à tous les neurones de la couche suivante. Elles servent à combiner les caractéristiques extraites par les couches précédentes pour effectuer la classification ou la régression finale [Goodfellow et al., 2016 ; LeCun et al., 2015].

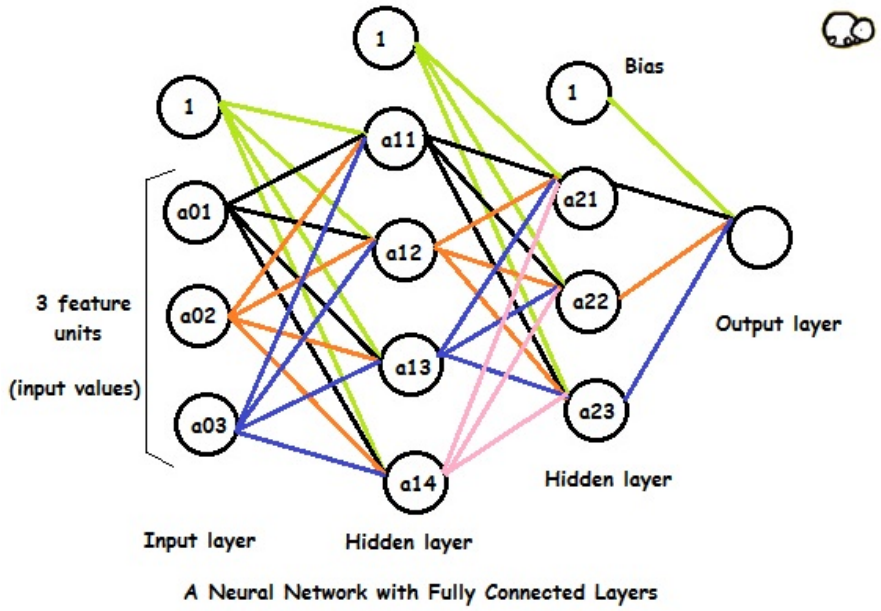


FIG. 2.5 – Couches entièrement connectées (*Fully Connected Layers*).

2.3.2 Architecture CNN

Ces études [LeCun et al., 1998; Krizhevsky et al., 2012] suggèrent qu'une architecture CNN (réseau de neurones convolutifs) est une structure composée de couches spécialisées, principalement des couches de convolution, de regroupement (*pooling*) et entièrement connectées, permettant d'extraire automatiquement des caractéristiques hiérarchiques pertinentes à partir de données comme des images [Zeiler et al., 2014].

La recherche récente [Zoph et al., 2017; Tan et al., 2019] démontre que la conception de ces architectures peut être soit manuelle soit automatisée (via le *Neural Architecture Search* – NAS), pour optimiser conjointement la précision, l'efficacité computationnelle et l'adaptation à des tâches spécifiques [He et al., 2020].

A. Architecture ResNet

L'architecture de ResNet (*Residual Network*)[He et al., 2016] est une avancée majeure dans le domaine des réseaux de neurones profonds, conçue pour faciliter l'entraînement de réseaux très profonds en résolvant le problème du gradient qui disparaît [Hochreiter et Schmidhuber, 1997].

L'idée centrale repose sur l'utilisation de "blocs résiduels" : au lieu d'apprendre directement une fonction complexe, chaque bloc apprend la différence (ou *résidu*) entre l'entrée et la sortie attendue, grâce à un *chemin de raccourci* (*skip connection*) qui ajoute l'entrée du bloc à sa sortie [He et al., 2016].

Cette structure permet au réseau de transmettre plus facilement l'information et le gradient à travers de nombreuses couches, rendant possible l'entraînement de réseaux avec des dizaines, voire des centaines de couches, comme ResNet-50 ou ResNet-152 [He et al., 2016] . Les blocs résiduels sont composés de couches de convolution, de normalisation et d'activation, mais c'est la connexion directe entre l'entrée et la sortie qui distingue ResNet des architectures classiques [He et al., 2016].

Cette approche améliore la stabilité de l'apprentissage, la généralisation et la performance sur des tâches complexes comme la classification d'images [Veit et al., 2016]. Des variantes et améliorations de ResNet, comme Res2Net [Gao et al., 2019] ou ResNeSt [Zhang et al., 2020], ont été développées pour renforcer la représentation multi-échelle et l'attention dans le réseau.

En résumé, ResNet se distingue par sa capacité à entraîner des réseaux très profonds grâce à l'introduction des connexions résiduelles, ce qui a révolutionné l'apprentissage profond pour la vision par ordinateur [He et al., 2016; Veit et al., 2016].

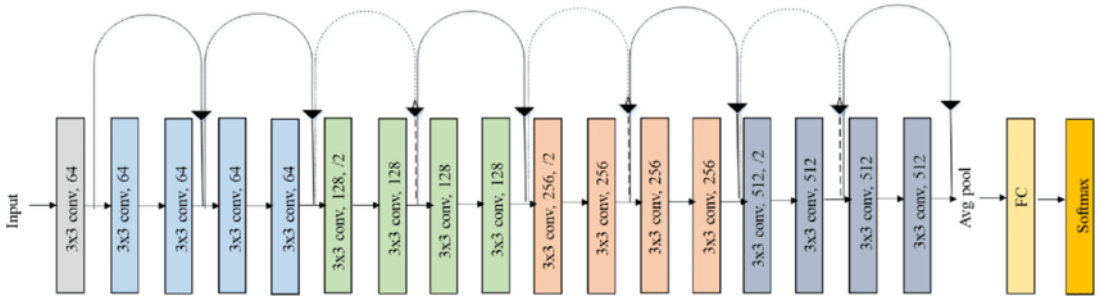


FIG. 2.6 – Architecture générale du réseau ResNet avec blocs résiduels.

B. Architecture MobileNet

L'architecture **MobileNet**est une famille de réseaux de neurones convolutifs conçue spécifiquement pour les applications de vision par ordinateur sur des appareils mobiles et embarqués, où les ressources en mémoire et en calcul sont limitées. Sa principale innovation réside dans l'utilisation de *convolutions séparables en profondeur* (*depthwise separable convolutions*) [Howard et al., 2017; Sandler et al., 2018], qui décomposent la convolution standard en deux opérations :

(1) une convolution spatiale par canal (depthwise),

(2) une convolution *pointwise* (1×1).

Cette structure réduit considérablement le nombre de paramètres et la charge de calcul (jusqu'à 8 à 9 fois moins que les CNN standards), tout en maintenant de bonnes performances [Howard et al., 2019].

MobileNet introduit également des *hyperparamètres globaux*, tels que le **width multiplier** α et le **resolution multiplier** ρ, permettant d'ajuster dynamiquement le compromis entre précision et rapidité selon les besoins de l'application [Howard et al., 2017; Tan et al., 2019].

Cette architecture légère est largement utilisée pour la classification d'images, la détection d'objets et la reconnaissance faciale [Zhang et al., 2020]. Elle se décline en plusieurs versions :

— **MobileNetV1** [Howard et al., 2017] : introduction des convolutions séparables en profondeur ;

— **MobileNetV2** [Sandler et al., 2018] : ajout des blocs à inversion de résidu (*inverted residuals*) et goulots d'étranglement linéaires ;

— **MobileNetV3**[Tan et al., 2019] : optimisation via la recherche d'architecture (NAS) et ajout de l'attention *h-swish*.

MobileNet est particulièrement adaptée aux dispositifs à faible consommation d'énergie (≤100 mW), comme les smartphones ou microcontrôleurs [Howard et al., 2019; Warden, 2019], tout en offrant des résultats compétitifs par rapport à des modèles plus volumineux (ex. : 70,6% de précision sur ImageNet avec seulement 4,2M de paramètres) [Howard et al., 2017; Tan et al., 2019].

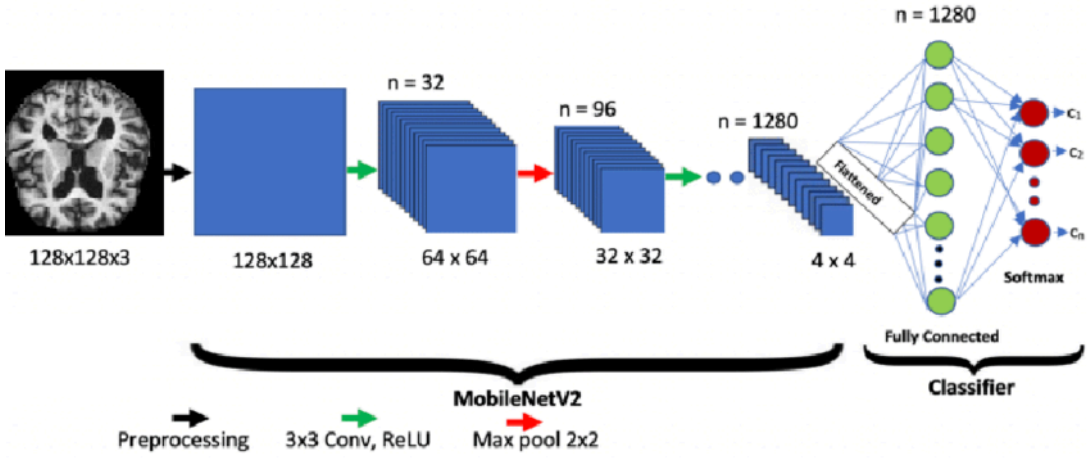


FIG. 2.7 – Proposed MobileNetV2 model basic architecture.

C. Architecture EfficientNet

EfficientNet [**tan2019efficientnet**] est une famille d'architectures de réseaux de neurones convolutifs conçue pour optimiser à la fois la précision et l'efficacité computationnelle. Son innovation principale réside dans le *compound scaling*[**Tan et al., 2019**], une méthode qui ajuste simultanément la profondeur (nombre de couches), la largeur (nombre de canaux par couche) et la résolution d'entrée selon des ratios équilibrés contrôlés par un facteur ϕ. Cette approche permet d'obtenir de meilleures performances avec moins de paramètres et de calculs que les architectures traditionnelles [Radosavovic et al., 2020].

EfficientNet repose sur :

— des blocs *inverted bottleneck* issus de **MobileNetV2** [Sandler et al., 2018],

— combinés à des modules *squeeze-and-excitation* (SE) [Sandler et al., 2018],

qui permettent au réseau de renforcer la représentation des canaux les plus importants, améliorant ainsi l'expressivité du modèle.

Le modèle de base, **EfficientNet-B0** [Tan et al., 2019], constitue la fondation d'une série de variantes (B1 à B7) qui augmentent en capacité en suivant la méthode de scaling ($\phi = 1.0$ à 1.8). Cette architecture a montré des performances de pointe sur des tâches de classification d'images [Deng et al., 2009], tout en étant 8.4× plus légère et 6.1× plus rapide que d'autres modèles conventionnels [Tan et al., 2019], ce qui la rend idéale pour des applications en temps réel ou embarquées.

Des variantes optimisées comme **EfficientNet-Lite** [Google, 2020] et **EfficientNet-EdgeTPU**[Google, 2021] ont été développées pour des contraintes strictes de ressources (vitesse <10 ms, faible mémoire), tout en conservant une précision élevée (>75% sur ImageNet) [Wu et al., 2021].

En résumé, EfficientNet se distingue par son équilibre entre efficacité et précision, grâce à une conception modulaire innovante et à une stratégie d'agrandissement bien calibrée [Tan et Le, 2021].

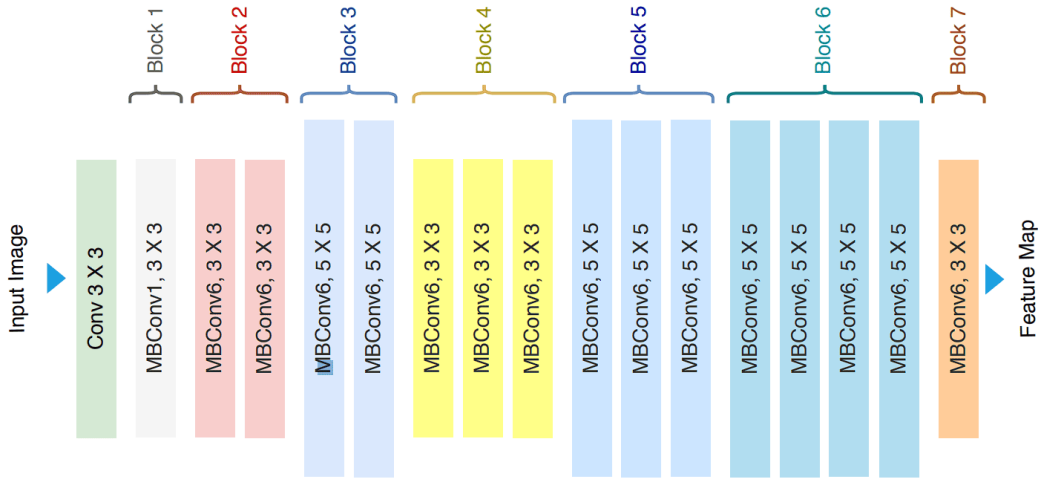


FIG. 2.8 – Architecture EfficientNet.

2.3.3 Vision Transformer (ViT)

Le *Vision Transformer* (ViT) est une architecture de Deep Learning appliquée à la vision par ordinateur. Contrairement aux réseaux convolutifs classiques (CNN), le ViT exploite le mécanisme d'attention des Transformers pour traiter des images sans utiliser de convolutions. L'image est divisée en petits patchs, chacun étant projeté dans un espace vectoriel, puis traité comme une séquence par un encodeur Transformer. Ce modèle permet de capturer des dépendances globales entre les différentes régions de l'image, ce qui s'avère particulièrement efficace lorsque le modèle est entraîné sur de grandes quantités de données. ViT représente une étape importante vers des architectures plus générales et puissantes dans le domaine de la vision par ordinateur [Dosovitskiy et al., 2020].

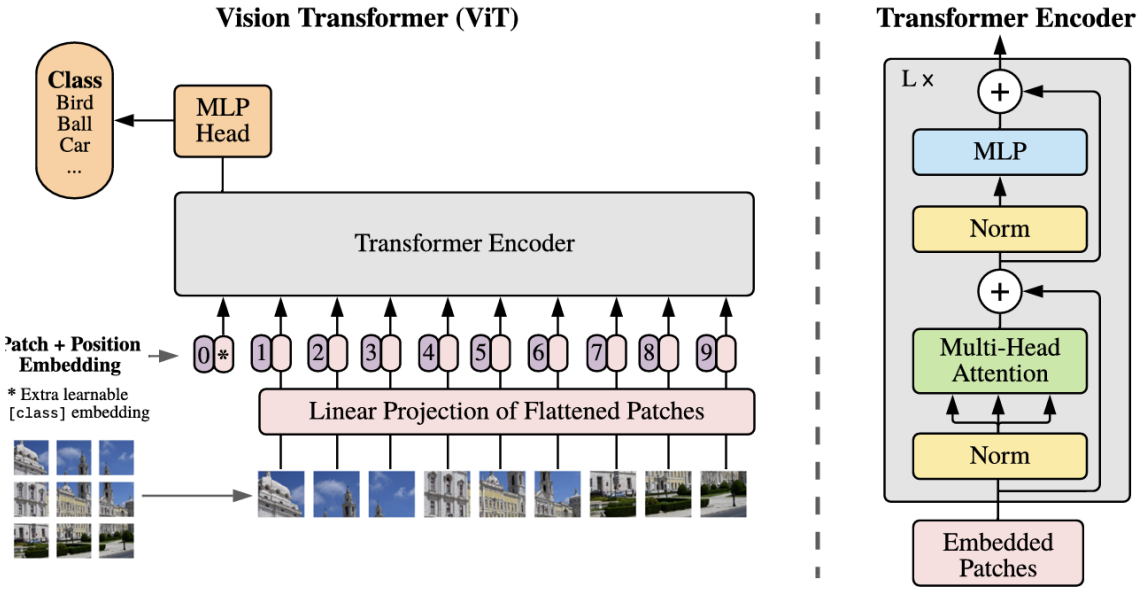


FIG. 2.9 – Architecture du Vision Transformer (ViT).

Le modèle Vision Transformer (**ViT**) se compose de trois blocs principaux : [**Dosovitskiy et al., 2020**] :

1. **Patch Embedding** :
   L'image d'entrée est divisée en petits patchs non superposés de taille fixe (par exemple 16×16 pixels). Chaque patch est transformé en un vecteur, projeté dans un espace de dimension D à l'aide d'une couche linéaire.

   — Une image de taille $H \times W \times C$ est découpée en N patchs de taille $P \times P$.
   — Chaque patch devient un vecteur de taille $P \times P \times C$.
   — Une couche linéaire transforme ces vecteurs en des embeddings de dimension D.
   — Un token spécial [CLS] (utilisé pour la classification) est ajouté au début de la séquence.
   — Un encodage positionnel est ajouté pour indiquer la position spatiale de chaque patch.

   Cette étape transforme l'image en une séquence de vecteurs — tout comme un texte est transformé en une séquence de mots dans le NLP.

2. **Encodeur Transformer** :
   C'est le cœur du modèle ViT. Il est composé de plusieurs blocs identiques contenant :

   — des mécanismes d'attention multi-têtes (Multi-Head Self-Attention),
   — des couches feed-forward (MLP),
   — des normalisations de couches (LayerNorm),
   — et des connexions résiduelles (Residual Connections).

   L'objectif est de permettre à chaque patch d'accéder au contexte global de l'image — contrairement aux CNN qui traitent localement.

3. **Tête de classification (MLP Head)** :
   Après le passage dans l'encodeur, seul le vecteur correspondant au token [CLS] est utilisé pour la sortie finale :

   — Le vecteur [CLS] contient un résumé global de l'image.
   — Il passe à travers une ou plusieurs couches entièrement connectées (MLP) avec activation (ReLU ou GELU).
   — La dernière couche donne les prédictions finales (par exemple, une probabilité pour chaque classe).

   Ce composant permet d'utiliser ViT pour des tâches telles que la classification, la régression ou d'autres tâches supervisées.

2.4 Apprentissage profond dans le traitement automatique du langage naturel (TALN)

L'**apprentissage profond** (*deep learning*), sous-domaine de l'apprentissage automatique, modélise des données linguistiques via des architectures neuronales multicouches

combinant des transformations non linéaires successives [Goodfellow et al., 2016]. Cette approche permet de construire des *représentations hiérarchiques du langage* : des concepts abstraits (syntaxe, sémantique, pragmatique) émergent progressivement de motifs élémentaires (morphèmes, mots), organisés en couches de traitement [Goldberg, 2017].

La « profondeur » désigne ici le nombre de couches transformant les entrées brutes en représentations de haut niveau [Schmidhuber, 2015].

2.4.1 Mémoire à long terme et court terme (LSTM)

Le **LSTM (Long Short-Term Memory)** est un type avancé de réseau de neurones récurrents (**RNN**) conçu pour traiter, apprendre et mémoriser des séquences temporelles sur de longues périodes. Il a été introduit par [**Hochreiter et Schmidhuber**, **1997**] pour résoudre les problèmes des RNN classiques, notamment la disparition ou l'explosion du gradient lors de l'apprentissage de dépendances à long terme.

A. Fonctionnement de la mémoire à court et long terme dans LSTM

L'unité LSTM repose sur une architecture composée de trois portes principales et d'une cellule mémoire centrale permettant de contrôler le flux d'information à travers le temps.

B. Architecture du LSTM pour le traitement des questions

Dans notre système de Visual Question Answering (VQA), la composante textuelle — à savoir les questions en langage naturel — est traitée à l'aide d'un réseau LSTM (Long Short-Term Memory). Ce type de réseau récurrent est particulièrement bien adapté pour modéliser les séquences, car il est capable de capturer les dépendances à long terme entre les mots, tout en limitant le problème du gradient qui disparaît.

Chaque question est d'abord convertie en une séquence d'indices entiers correspondant aux mots du vocabulaire, puis encodée par une couche d'**embedding**. Cette dernière transforme chaque mot en un vecteur dense de dimension fixe (par exemple, 300 dimensions), permettant au modèle d'apprendre des représentations sémantiques plus riches.

La séquence de vecteurs denses est ensuite transmise à une couche **LSTM** unidirectionnelle. Cette couche est constituée de plusieurs cellules LSTM qui traitent la séquence mot par mot. À chaque pas de temps, le LSTM met à jour un état caché qui résume l'information textuelle jusqu'au mot actuel. Nous retenons généralement le dernier état caché de la séquence, car il encode l'information globale de la question.

Formellement, le traitement de la question peut être représenté par :

$$\text{Question}_{\text{encoded}} = \text{LSTM}(\text{Embedding}(Q))$$

où Q est la séquence d'indices de la question, et $\text{Question}_{\text{encoded}}$ est le vecteur de représentation final, de dimension fixe (par exemple 512), utilisé dans la concaténation multimodale avec l'image.

Cette représentation vectorielle de la question est ensuite combinée avec la sortie du CNN appliqué à l'image, permettant à la couche suivante (un classifieur entièrement connecté) d'apprendre à prédire la réponse en fonction des deux modalités.

L'utilisation du LSTM permet ainsi de capter la structure syntaxique et sémantique de la question, tout en fournissant une vectorisation compacte compatible avec les traitements multimodaux.

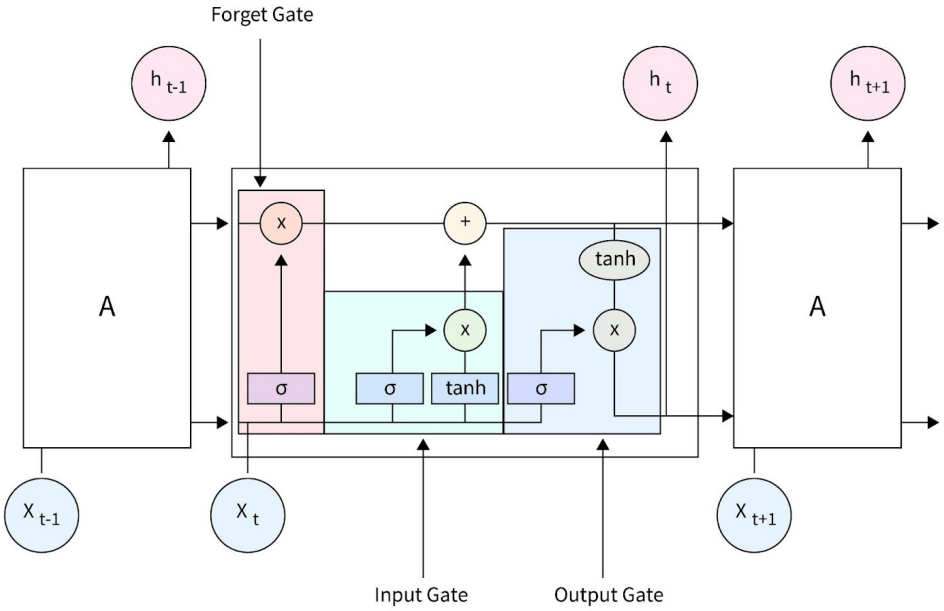


FIG. 2.10 – Architecture LSTM

C. Structure de la cellule LSTM

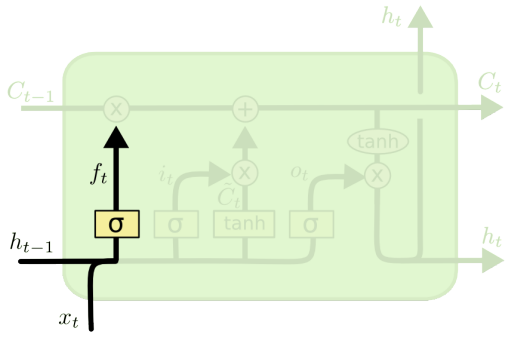
— **Cellule mémoire C_t :** conserve l'information pertinente sur plusieurs pas de temps, jouant le rôle de mémoire à long terme.

— **État caché h_t :** représente la mémoire à court terme, utilisée comme sortie de l'unité LSTM à chaque étape temporelle t.

D. Mécanismes de portes

Chaque porte est contrôlée par une fonction sigmoïde σ, et les informations sont combinées à l'aide de produits de Hadamard (*multiplication élément par élément*).

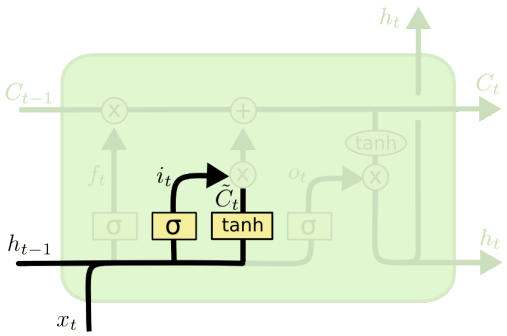
— **Porte d'oubli :** Elle décide des informations à oublier dans la cellule mémoire C_{t-1}.



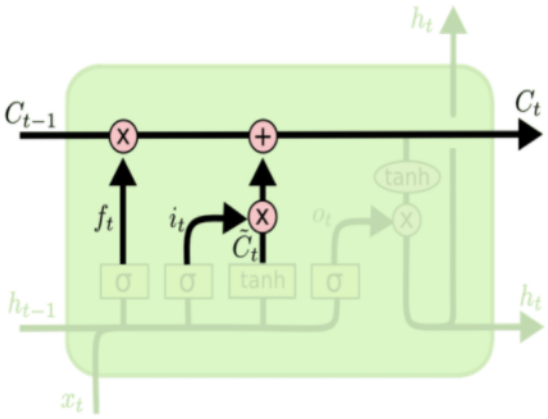
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

FIG. 2.11 – La porte d'oubli d'une cellule LSTM

— **Porte d'entrée :** Elle choisit les nouvelles informations à ajouter à la cellule mémoire.



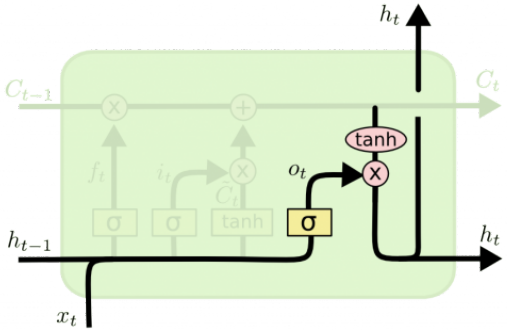
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

FIG. 2.12 – La porte d'entrée d'une cellule LSTM

— **Porte de sortie :**
Elle détermine la sortie actuelle du LSTM à chaque étape.



$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

FIG. 2.13 – La porte d'oubli d'une cellule LSTM

2.5 Visual Question Answering (VQA)

Le Visual Question Answering (VQA) est une tâche multimodale qui consiste à répondre automatiquement à une question formulée en langage naturel à propos d'une image. Par exemple, à partir d'une image, un système peut répondre à la question :

« Combien de parts de pizza y a-t-il ? ». Pour cela, le modèle doit analyser l'image, comprendre le texte de la question, et modéliser les interactions entre ces deux modalités [Geman et al., 2015].

Cette tâche, souvent considérée comme un "Visual Turing Test", requiert une compréhension avancée du langage naturel et de la perception visuelle. Elle constitue un excellent banc d'essai pour évaluer conjointement les capacités de raisonnement, d'analyse d'image et de traitement du langage d'un modèle d'apprentissage profond [Gurari et al., 2018]. Elle trouve également des applications concrètes dans des domaines variés tels que l'assistance aux personnes malvoyantes, l'interaction homme-robot, ou encore la recherche d'information visuelle [Malinowski et Fritz, 2014].

Les premières approches du VQA reposaient sur des jeux de données restreints comme DAQUAR[Malinowski , 2014] ou le Visual Turing Test [Geman et al., 2015]. Cependant, la petite taille et la limitation du vocabulaire de ces jeux de données ne permettaient pas de tirer pleinement parti de l'apprentissage profond. Des datasets plus larges comme VQA v2.0 ou CLEVR ont alors été développés, facilitant l'émergence de modèles plus puissants capables de combiner efficacement texte et image [Johnson et al., 2017].

Des approches variées ont été proposées pour fusionner les deux modalités, comme le *Multimodal Compact Bilinear pooling* (MCB) [Fukui et al., 2016] ou les *Relational Networks* (RN) [Santoro et al., 2017], qui permettent au modèle de raisonner sur les relations entre objets visuels en combinant efficacement les représentations textuelles et visuelles.

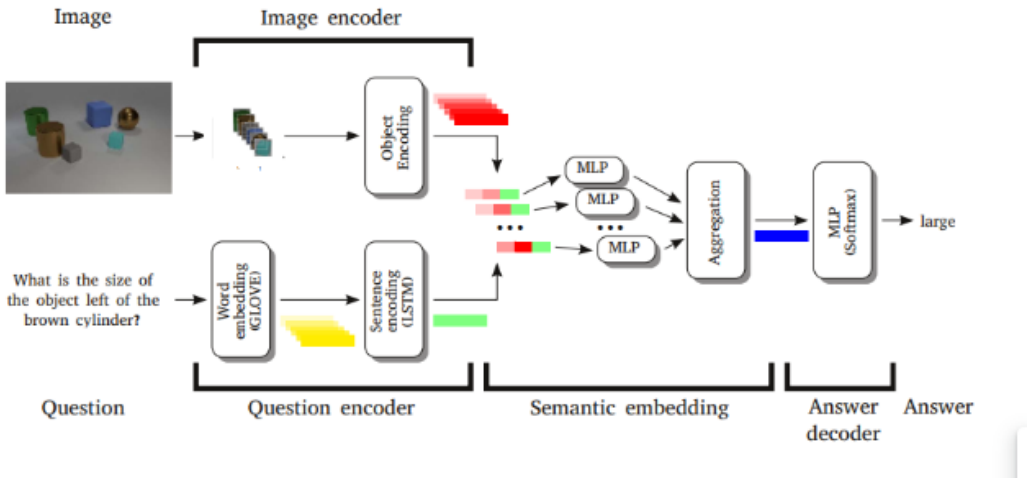


FIG. 2.14 – Architecture générale d'un système VQA basé sur l'encodage multimodal

2.5.1 Méthodes multi-modales

Les méthodes multi-modales ont pour objectif de traiter simultanément des données issues de modalités différentes — ici, les **images** et le **texte**. Dans le cas du VQA, le modèle doit extraire des caractéristiques visuelles via des réseaux convolutifs (CNN), encoder la question avec des architectures séquentielles comme les LSTM ou des Transformers, et apprendre à combiner les deux types de représentation dans un espace latent commun. Cette intégration permet au système de raisonner sur le contenu de l'image en lien avec la question posée.

2.5.2 Exemples de méthodes

Plusieurs approches ont été proposées pour résoudre le problème du VQA :

— **MCB (Multimodal Compact Bilinear Pooling)** : permet la fusion efficace de représentations visuelles et textuelles via une opération bilinéaire compressée.

— **Relational Networks (RN)** : concatènent les caractéristiques visuelles de paires de régions avec la représentation de la question afin de raisonner sur les relations entre objets.

— **Modèles Transformer multi-modaux** comme *LXMERT*, *UNITER* ou *ViLBERT* : utilisent une attention croisée entre les mots et les régions d'images, permettant une représentation contextuelle riche pour chaque modalité.

Ces approches renforcent la capacité des systèmes à répondre à des questions nécessitant du raisonnement spatial, logique ou sémantique.

2.5.3 Applications

Le VQA trouve de nombreuses applications concrètes :

— **Aide aux personnes malvoyantes** : un système peut répondre à des questions visuelles sur l'environnement immédiat.

— **Interaction homme-machine** : pour les robots ou assistants intelligents dotés de vision artificielle.

— **Recherche d'images intelligente** : permettre des requêtes textuelles complexes sur des bases de données d'images.

— **Applications éducatives ou ludiques** : intégration de VQA dans des outils pédagogiques ou de divertissement interactifs.

Le VQA représente ainsi une avancée vers des systèmes d'intelligence artificielle capables de compréhension multi-modale et d'interaction naturelle.

2.6 Conclusion

Dans ce chapitre, nous avons exploré les concepts fondamentaux de l'apprentissage profond, notamment les réseaux de neurones convolutifs (CNN) et récurrents (RNN), avec un accent particulier sur les architectures pertinentes pour les données visuelles et séquentielles. Nous avons étudié comment les CNN extraient automatiquement des caractéristiques hiérarchiques à partir des images, et comment les RNN, en particulier les LSTM, sont capables de modéliser des dépendances à long terme dans des séquences de donnée(text, audio, vidéo...).

Nous avons également présenté le fonctionnement des couches de convolution, de regroupement et entièrement connectées, ainsi que les mécanismes de mémoire des LSTM, qui jouent un rôle central dans la compréhension du langage naturel. Cette base théorique est directement exploitée dans notre projet, où une combinaison d'extraction de caractéristiques visuelles par CNN et de traitement séquentiel par LSTM est utilisée pour répondre à des questions en langage naturel à partir d'images (Visual Question Answering - VQA).

Cette approche conjointe image/texte illustre la puissance du Deep Learning à modéliser et traiter des données multimodales complexes. Le chapitre suivant sera dédié

à la présentation de l'architecture de notre système, aux choix méthodologiques et aux résultats obtenus.

3 Chapitre 3 : Étude expérimentale

3.1 Introduction

Le Visual Question Answering (VQA) est un domaine interdisciplinaire qui combine l'analyse d'images et la compréhension du langage naturel. L'objectif est de permettre aux systèmes intelligents de répondre automatiquement à des questions textuelles portant sur le contenu d'une image. Afin d'évaluer avec précision les capacités de raisonnement de ces modèles, le dataset CLEVR a été conçu. Il s'agit d'un ensemble de données synthétiques contenant des scènes 3D soigneusement générées, composées de formes géométriques variées possédant différentes propriétés (couleur, taille, position, etc.). Chaque scène est accompagnée de questions complexes nécessitant des formes de raisonnement logique avancées, telles que le comptage, la comparaison ou l'inférence causale.

Ce qui distingue CLEVR des autres jeux de données, c'est son absence de biais visuels ou linguistiques, ce qui en fait un outil de référence rigoureux pour tester la compréhension multimodale. Il est largement utilisé pour développer et évaluer des modèles VQA visant une compréhension visuelle et linguistique profonde et intégrée.

3.2 Préparation des données

3.2.1 Dataset CLEVR

Le dataset CLEVR (Compositional Language and Elementary Visual Reasoning) est un jeu de données synthétique conçu pour évaluer les capacités de raisonnement visuel et linguistique des modèles d'intelligence artificielle, en particulier dans le cadre de la tâche de Visual Question Answering (VQA).

Les images de CLEVR représentent des scènes complexes en 3D, générées à l'aide du moteur de rendu Blender. Chaque scène se compose d'un plan gris sur lequel sont disposés entre 3 et 10 objets géométriques (sphères, cubes, cylindres). Ces objets varient selon quatre attributs principaux : la forme, la couleur, la matière (caoutchouc mat ou métal brillant) et la taille (petit ou grand). La position des objets ainsi que celle des sources lumineuses est aléatoire afin de garantir une grande diversité visuelle.

Le jeu de données contient environ 70,000 images distinctes dans l'ensemble d'entraînement et 15,000 images différentes dans l'ensemble de validation. Chaque image est associée à un ensemble de questions textuelles générées automatiquement à partir d'un modèle de génération. On dénombre au total 699,989 questions pour l'ensemble d'entraînement et 149,991 pour la validation.

Les questions posées couvrent plusieurs catégories :

— des questions existentielles (par exemple : « Y a-t-il une sphère rouge ? »),

— des questions de comptage (ex. : « Combien de cubes verts sont présents ? »),

— des questions d'identification d'attributs (ex. : « Quelle est la couleur de l'objet situé à droite du cylindre ? »),

— des questions de comparaison (ex. : « Y a-t-il plus de cubes que de sphères ? »).

Certaines questions sont composées et nécessitent plusieurs étapes de raisonnement pour être résolues, impliquant ainsi des compétences cognitives avancées.

Malgré sa richesse en termes de structure et de diversité, CLEVR reste un environnement contrôlé et artificiel. Le langage utilisé est basé sur des modèles de phrases générés automatiquement à partir de templates, ce qui le distingue fortement du langage naturel auquel sont exposés les humains. Par exemple, les enfants acquièrent le langage à partir d'entrées bien plus riches, incluant l'intonation, les gestes, les contextes sociaux et des structures syntaxiques variées.

Cependant, cette simplicité constitue également une force : elle permet d'isoler les compétences de raisonnement des modèles, de contrôler précisément les variables, et de poser un cadre expérimental rigoureux pour l'étude de l'apprentissage multimodal.

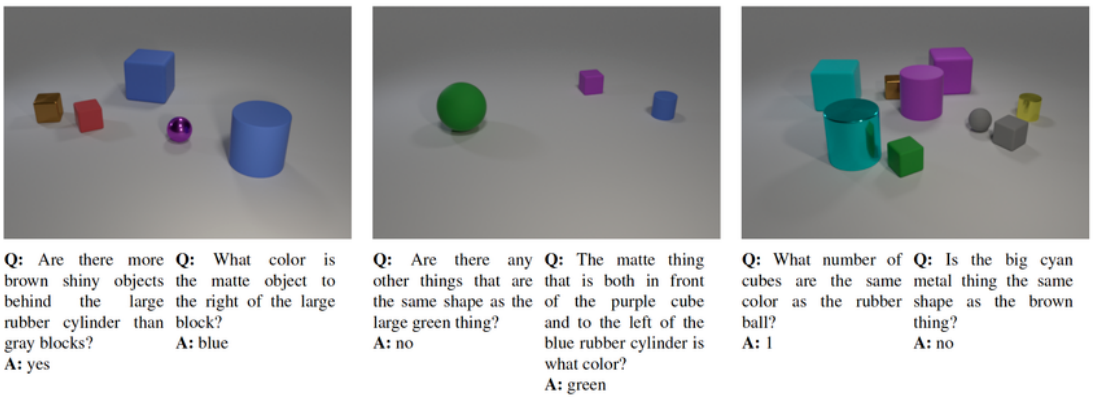


FIG. 3.1 – Example clevr dataset.

3.2.2 Équilibrage des classes par duplication

Dans le cadre de notre tâche de Visual Question Answering (VQA), le déséquilibre entre les classes de réponses peut entraîner un apprentissage biaisé, où le modèle favorise les réponses les plus fréquentes au détriment des classes rares. Pour contrer ce problème, nous avons mis en œuvre une stratégie d'équilibrage consistant à dupliquer artificiellement les échantillons des classes minoritaires. Concrètement, après avoir chargé les questions du dataset CLEVR, nous avons regroupé celles-ci par type de réponse, puis analysé la distribution. Afin d'uniformiser cette distribution, nous avons fixé un seuil de 2000 échantillons par classe. Les classes contenant plus de 2000 exemples ont été tronquées, tandis que celles en contenant moins ont été complétées par des duplications aléatoires (avec random.choices) jusqu'à atteindre ce seuil. Cette méthode, bien que simple, garantit une représentation équitable de chaque réponse dans l'ensemble de données, ce qui contribue à un entraînement plus stable et à une meilleure précision globale du modèle. Il est important de souligner que cette étape n'est pas de la data augmentation à proprement parler, car aucun nouveau contenu n'a été généré : seules des copies d'instances existantes ont été ajoutées pour rétablir un équilibre.

3.2.3 Redimensionnement des données

Dans tous les modèles, les images sont redimensionnées à une taille standard de 224x224 pixels. Cela est essentiel pour garantir la compatibilité avec les architectures CNN préentraînées , qui attendent des entrées de dimensions fixes.

3.2.4 Normalisation des données

Les valeurs de pixel sont mises à l'échelle automatiquement grâce à la transformation transforms.ToTensor(), qui convertit les pixels d'une image de [0, 255] à une échelle normalisée entre [0.0, 1.0]. Cela accélère l'entraînement et améliore la convergence du modèle.

3.2.5 Mise à l'échelle

Bien que les réseaux préentraînés de PyTorch normalisent automatiquement les données dans certaines architectures, nous avons appliqué une transformation standard à nos images : la conversion en tenseurs, suivie d'une normalisation implicite via l'utilisation des poids préentraînés. Cela garantit que les valeurs des pixels sont comprises dans un intervalle standardisé adapté aux modèles convolutifs, ce qui contribue à une convergence plus rapide pendant l'apprentissage.

3.2.6 Prétraitement des questions

Le prétraitement des questions dans notre système est une étape fondamentale pour transformer le langage naturel en une forme que le modèle peut comprendre. Dans notre approche, ce processus inclut principalement deux opérations : la tokenisation et le padding/troncature.

A. Tokenisation

La première étape consiste à découper chaque question en mots individuels, ou tokens. Cette opération, appelée tokenisation, est réalisée en convertissant la question en minuscules, en supprimant les signes de ponctuation comme le point d'interrogation, puis en séparant la chaîne de caractères selon les espaces. Par exemple, la question : « What is the color of the cube ? » sera convertie en la liste de mots : [”what”, ”is”, ”the”, ”color”, ”of”, ”the”, ”cube”]. Ce traitement permet de constituer un vocabulaire à partir de tous les mots rencontrés dans les questions d'entraînement. Chaque mot du vocabulaire se voit attribuer un identifiant numérique unique. Une valeur spéciale est réservée pour le padding (<PAD>), avec l'identifiant 0. Les séquences de mots sont ensuite converties en séquences d'entiers en remplaçant chaque mot par son identifiant correspondant. Dans notre implémentation, la longueur maximale d'une question est limitée à 30 mots, les séquences plus longues étant tronquées, et les plus courtes remplies avec des zéros (<PAD>).

B. Embedding et Passage dans le LSTM

Une fois les questions converties en séquences d'entiers, elles sont passées à une couche Embedding. Cette couche transforme chaque identifiant de mot en un vecteur dense de

taille fixe (128 dimensions dans notre cas), apprenant ainsi une représentation vectorielle optimisée des mots pendant l'entraînement. Ces vecteurs sont ensuite transmis à un réseau LSTM, qui lit la séquence mot par mot et en extrait une représentation contextuelle globale de la question. Cette sortie du LSTM capture la structure et le sens de la question, ce qui est crucial pour que le modèle puisse la combiner efficacement avec les caractéristiques visuelles de l'image.

3.2.7 Prétraitement des réponses

Dans le cadre de notre système de Visual Question Answering (VQA), le prétraitement des réponses est une étape indispensable pour transformer les réponses textuelles du dataset en une forme exploitable par le modèle d'apprentissage automatique. Les réponses fournies dans le jeu de données CLEVR sont des éléments simples tels que des couleurs, des formes, des chiffres ou des expressions courtes comme « yes » ou « no ». Cependant, les modèles de deep learning ne peuvent pas traiter directement des chaînes de caractères. Il est donc nécessaire de convertir ces réponses en entiers. Pour cela, nous avons commencé par identifier toutes les réponses uniques présentes dans le dataset. Ensuite, nous avons attribué à chaque réponse un identifiant numérique unique grâce à un dictionnaire nommé answer2idx. Ce dictionnaire permet de faire le lien entre chaque réponse sous forme de texte et son équivalent sous forme d'indice. Par exemple, si les réponses sont « red », « blue » et « green », le dictionnaire pourrait leur associer les indices 0, 1 et 2 respectivement. Lors de la construction du dataset, chaque réponse est ainsi remplacée par son indice correspondant. Cette conversion permet d'utiliser ces réponses comme étiquettes cibles dans un problème de classification. Cette représentation est directement compatible avec les fonctions de perte classiques comme CrossEntropyLoss, qui attend des classes sous forme d'entiers.

3.2.8 Data Augmentation

Afin de renforcer la robustesse des modèles face à des variations naturelles ou artificielles dans les données d'entrée, nous avons mis en œuvre une stratégie d'augmentation des données visuelles. Cette technique consiste à appliquer des transformations aléatoires aux images, telles que des rotations légères, des flips horizontaux, des ajustements de la luminosité, du contraste ou de la saturation, ainsi que des translations mineures. Ces augmentations sont réalisées uniquement pendant l'entraînement afin de simuler des conditions visuelles variées, ce qui permet au modèle d'apprendre des représentations plus invariantes et donc plus généralisables. Cela s'est avéré particulièrement utile pour les trois modèles testés (ResNet18, MobileNet et EfficientNet), en réduisant le surapprentissage et en améliorant la capacité du modèle à traiter des cas non vus.

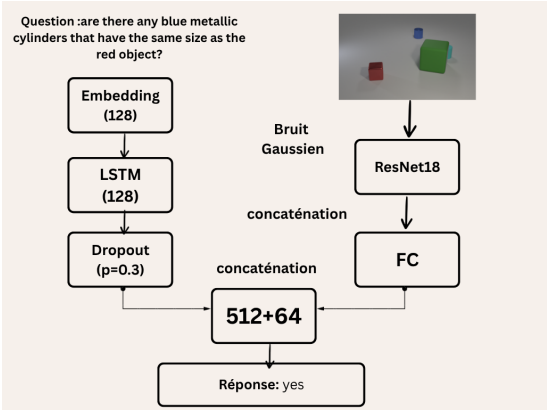
3.3 Système proposé

Le système proposé repose sur une architecture bimodale combinant à la fois des données visuelles et textuelles pour résoudre la tâche de Visual Question Answering (VQA). Plus précisément, un réseau de neurones convolutionnel préentraîné (tel que ResNet18, MobileNetV2 ou EfficientNet-B0/B1) est utilisé pour extraire les caractéristiques visuelles de l'image, tandis qu'un réseau LSTM traite la question posée en langage naturel après une phase d'embedding. Ces deux représentations – visuelle et textuelle – sont ensuite

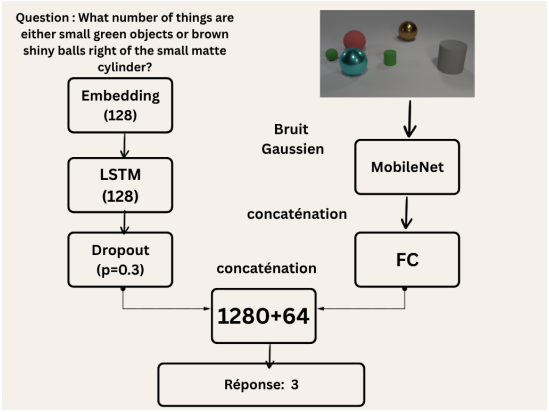
concaténées et introduites dans une couche fully connected qui prédit la réponse finale parmi un ensemble de classes. De plus, un bruit gaussien est injecté aux représentations visuelles (et parfois textuelles) durant l'entraînement afin de renforcer la robustesse du modèle face au surapprentissage. Cette approche permet de capturer efficacement les interactions entre les modalités image et texte, tout en tirant parti des capacités des modèles préentraînés et séquentiels.

3.3.1 Architecture bimodale (images + questions)

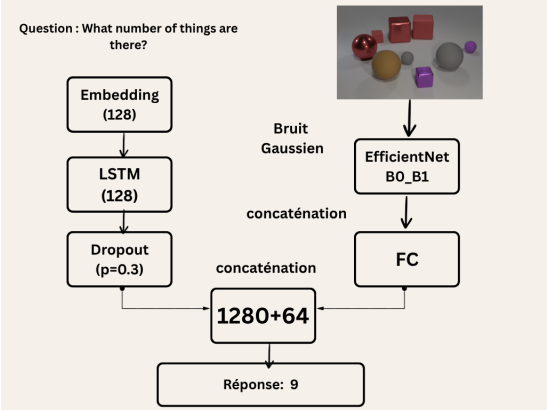
Le système de VQA proposé repose sur une architecture bimodale, ce qui signifie qu'il traite deux types de données distincts mais complémentaires : les images et les questions en langage naturel. Le traitement des images est assuré par un réseau de neurones convolutifs (CNN) préentraîné tel que ResNet18, MobileNet ou EfficientNet, qui extrait des caractéristiques visuelles discriminantes à partir des images du dataset CLEVR. Ces caractéristiques encodent des informations sur les formes, les couleurs, les positions et les relations spatiales entre les objets présents dans l'image. En parallèle, les questions sont d'abord tokenisées puis encodées via une couche d'embedding, avant d'être analysées par un réseau LSTM (Long Short-Term Memory) qui capture la structure séquentielle et sémantique du langage naturel. Les deux représentations – visuelle et textuelle – sont ensuite concaténées pour former une représentation conjointe qui est transmise à une couche entièrement connectée chargée de prédire la réponse. Cette architecture permet ainsi d'apprendre les interactions entre les modalités et d'optimiser la prédiction de réponses pertinentes.



(a) Architecture ResNet18



(b) Architecture MobileNetV2



(c) Architecture EfficientNet-B0/B1

Fig. 3.2 – Architectures des modèles CNN utilisés dans le système proposé.

3.3.2 Procédure d'entraînement

Pour entraîner notre système de Visual Question Answering (VQA), nous avons construit un jeu de données équilibré contenant un total de **56 000 échantillons**, soit **2000 exemples par classe** pour **28 classes de réponses**.

Chaque image a été redimensionnée à **224×224 pixels** et normalisée. Les questions textuelles ont été prétraitées par tokenisation, padding, et encodage via une couche d'embedding. Les réponses ont été converties en entiers correspondant aux différentes classes pour une tâche de classification.

L'entraînement a été réalisé sur **10 époques**, avec une taille de lot (*batch size*) de **64**, un taux d'apprentissage (*learning rate*) de **1e-4**, et un taux de **dropout** de **0.3** appliqué à la sortie du LSTM afin de réduire le surapprentissage.

L'optimisation a été effectuée avec l'algorithme **Adam**, et la fonction de perte utilisée est l'**entropie croisée** (*cross-entropy loss*). Une validation croisée a été utilisée avec un découpage des données : 80% pour l'entraînement, 10% pour la validation et 10% pour le test.

Les performances ont été mesurées à l'aide de plusieurs métriques, notamment la **précision**, les **matrices de confusion**, et les **courbes d'apprentissage**.

3.3.3 Métriques de performance

L'évaluation des performances du système repose sur plusieurs métriques classiques de classification supervisée. Ces métriques permettent d'évaluer dans quelle mesure le modèle répond correctement aux questions à partir des images, en considérant les réponses attendues. Nous présentons ci-dessous les principales mesures utilisées : *accuracy*, *precision*, *recall*, *F1-score*, ainsi que la matrice de confusion.

Accuracy (précision globale) L'accuracy mesure la proportion des prédictions correctes parmi l'ensemble des échantillons testés. Elle est définie comme suit :

$$\text{Accuracy} = \frac{\text{Nombre de bonnes prédictions}}{\text{Nombre total d'échantillons}} = \frac{TP + TN}{TP + TN + FP + FN}$$

où :

- TP : Vrais positifs,
- TN : Vrais négatifs,
- FP : Faux positifs,
- FN : Faux négatifs.

Précision (Precision) La précision correspond à la proportion des éléments prédits comme appartenant à une classe donnée qui sont effectivement corrects :

$$\text{Precision} = \frac{TP}{TP + FP}$$

Rappel (Recall) Le rappel (ou sensibilité) mesure la capacité du modèle à identifier correctement tous les exemples pertinents d'une classe :

$$\text{Recall} = \frac{TP}{TP + FN}$$

F1-score Le F1-score est la moyenne harmonique entre la précision et le rappel. Il constitue un bon compromis lorsque les classes sont déséquilibrées :

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Matrice de confusion La matrice de confusion est une table $n \times n$, où n est le nombre de classes. Chaque cellule (i, j) représente le nombre d'instances de la classe réelle i qui ont été prédites comme appartenant à la classe j. Elle permet d'analyser en détail les erreurs du modèle et les confusions fréquentes entre certaines classes.

Ces métriques sont calculées à l'aide de la bibliothèque `sklearn.metrics` dans notre code, en utilisant les prédictions générées par chaque modèle sur l'ensemble de test. Elles nous ont permis de comparer objectivement les modèles ResNet18, MobileNetV2, EfficientNet-B0 et B1.

3.4 Configuration expérimentale

Le code a été exécuté sur un ordinateur portable avec la configuration suivante :

— **Processeur** : Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz, 2.40GHz
— **Mémoire vive (RAM)** : 8 Go (7,88 Go utilisables)
— **Carte graphique** : Intel(R) HD Graphics 520
— **Stockage** : SSD Samsung MZNLN256HMHQ de 256 Go

3.5 Résultats expérimentaux et discussions

Cette section présente et analyse les résultats obtenus par les différents modèles testés dans le cadre de notre système de Visual Question Answering (VQA). Les modèles sélectionnés pour cette étude sont : **ResNet18**, **MobileNetV2** et **EfficientNet-B0/B1**, chacun étant couplé à un module LSTM pour le traitement des questions textuelles.

Nous évaluons leurs performances à l'aide de plusieurs métriques de classification : *accuracy*, *precision*, *recall*, *F1-score*, ainsi que les matrices de confusion. L'objectif est de comparer ces architectures en termes de capacité à modéliser la relation entre image et question, tout en tenant compte de leur complexité computationnelle.

3.5.1 Courbes d'apprentissage : précision et perte

Nous commençons par l'analyse des courbes d'apprentissage illustrant l'évolution de la précision (*accuracy*) et de la fonction de perte (*loss*) au fil des époques pour chaque modèle. Ces courbes permettent de juger de la vitesse de convergence et de la capacité du modèle à généraliser sans sur-apprentissage.

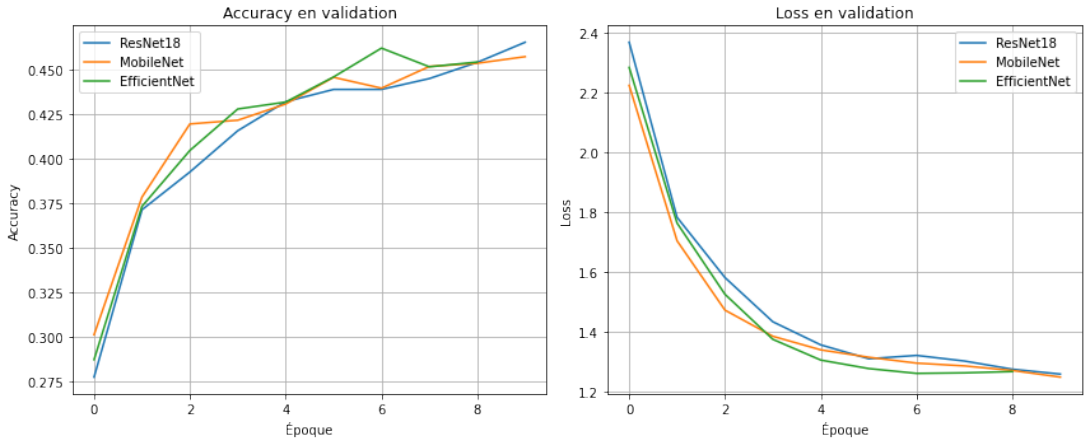


FIG. 3.3 – Comparaison globale des courbes de précision et de perte pour ResNet18, MobileNetV2 et EfficientNet-B0

3.5.2 Courbes individuelles par modèle

ResNet18 :

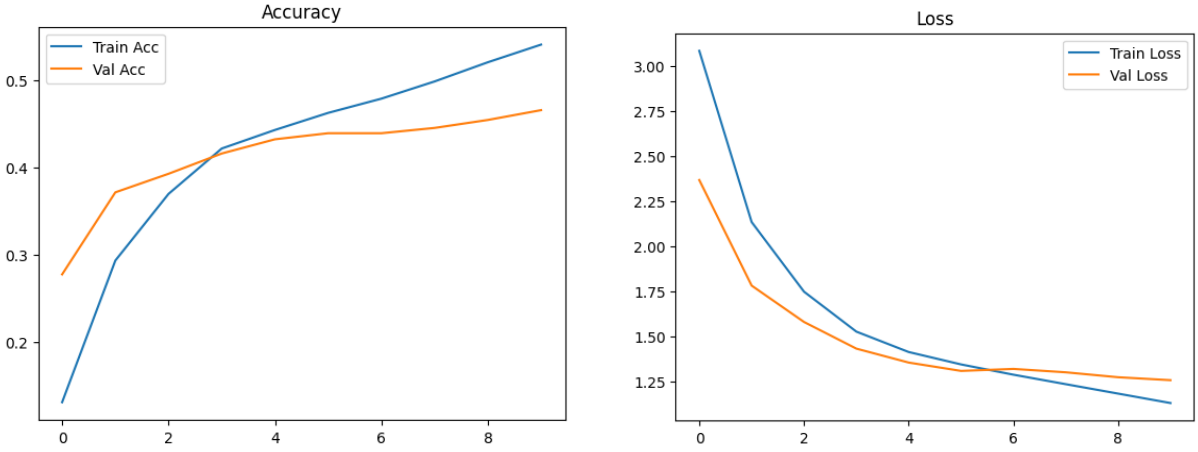


FIG. 3.4 – Courbes d'apprentissage pour le modèle ResNet18

MobileNetV2 :

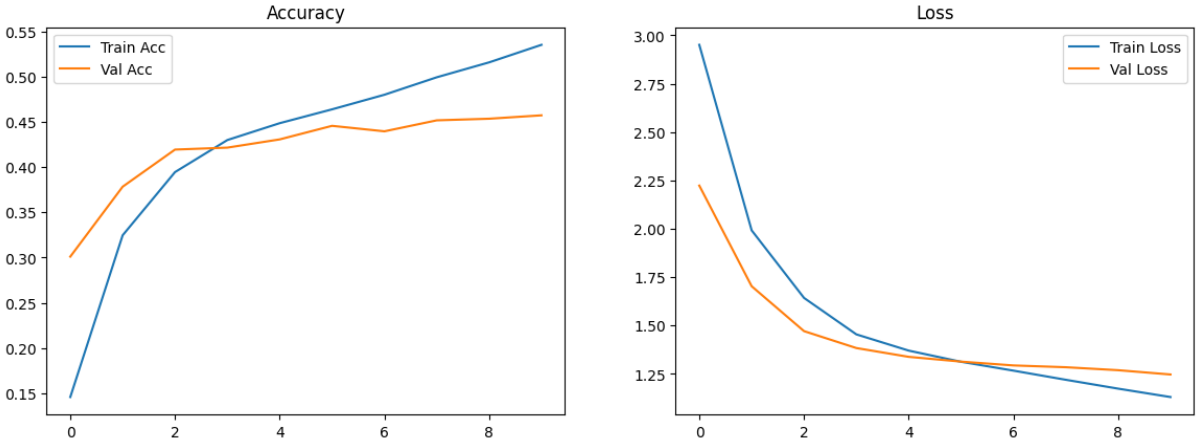


FIG. 3.5 – Courbes d'apprentissage pour le modèle MobileNetV2

EfficientNet-B0 :

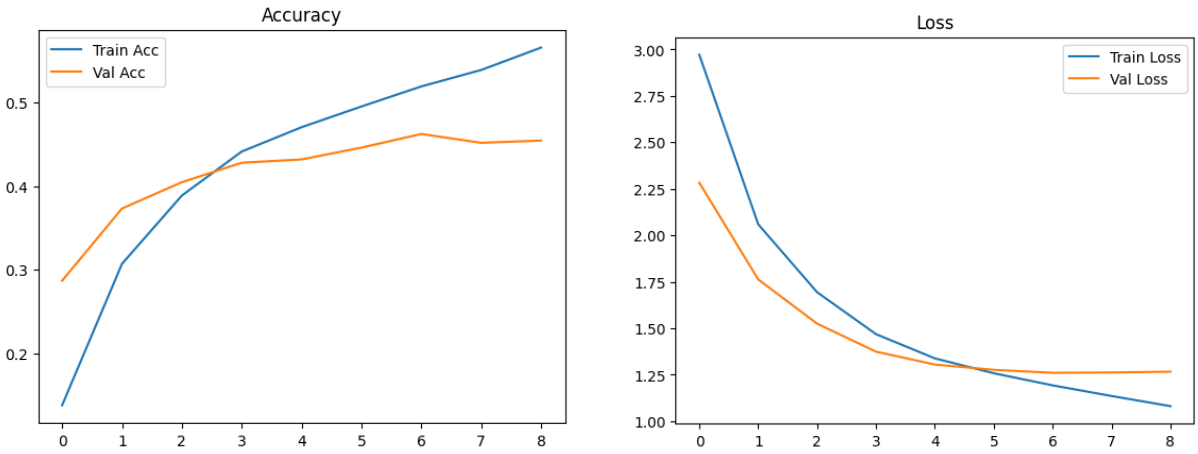


FIG. 3.6 – Courbes d'apprentissage pour le modèle EfficientNet-B0

3.5.3 Tableau comparatif des performances

Le tableau suivant synthétise les performances globales atteintes par chaque modèle à la fin de l'entraînement, en termes de précision sur l'ensemble de validation et de perte finale.

TAB. 3.1 – Comparaison des performances des modèles sur le dataset CLEVR

Modèle	**Accuracy**	**Loss**
ResNet18	46.5%	1.2575
EfficientNet-B0	46.2%	1.2598
MobileNetV2	45.7%	1.2474

Ces résultats révèlent que les trois modèles obtiennent des scores comparables. **ResNet18** atteint la meilleure précision, suggérant une capacité d'apprentissage robuste. Toutefois, **MobileNetV2** présente une perte plus faible, signe d'une optimisation plus stable. Enfin, **EfficientNet-B0** propose un bon compromis entre performance et efficacité computationnelle, avec des résultats proches de ResNet18.

3.5.4 Analyse via les matrices de confusion

Afin d'approfondir l'évaluation des performances des modèles, nous avons généré les matrices de confusion correspondant aux prédictions finales sur l'ensemble de test pour chaque architecture : **ResNet18**, **MobileNetV2** et **EfficientNet-B0/B1**.

Ces matrices permettent de visualiser, pour chaque classe, le nombre de prédictions correctes (présentes sur la diagonale principale) ainsi que les erreurs de classification (valeurs en dehors de la diagonale). Elles offrent une compréhension plus détaillée de la qualité des prédictions que ne le ferait une métrique globale telle que la précision (*accuracy*).

Ce type d'analyse est particulièrement pertinent pour la tâche de Visual Question Answering, où certaines réponses peuvent être confondues en raison de similarités visuelles

ou linguistiques. Les figures ci-dessous présentent les matrices de confusion obtenues pour les trois modèles.

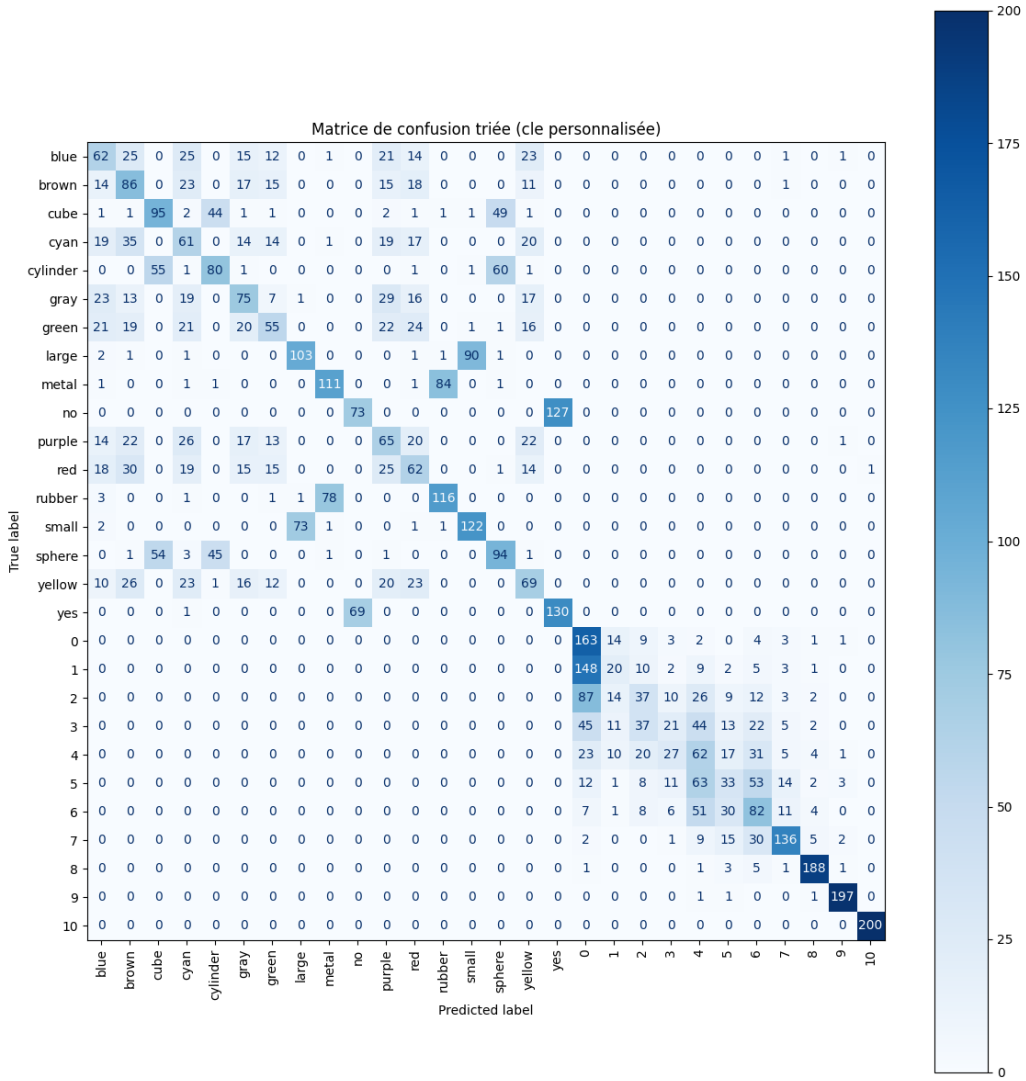


FIG. 3.7 – Matrice de confusion pour le modèle ResNet18

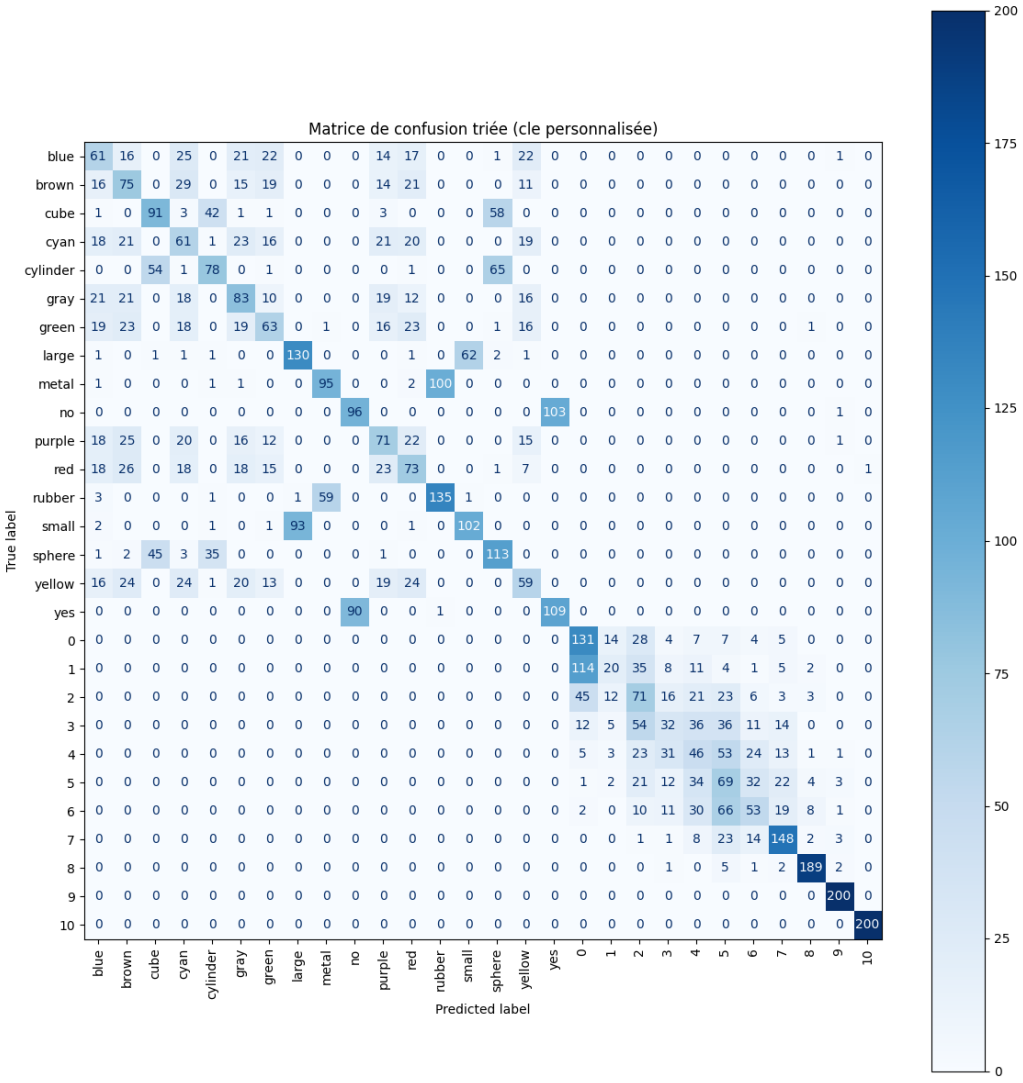


FIG. 3.8 – Matrice de confusion pour le modèle MobileNetV2

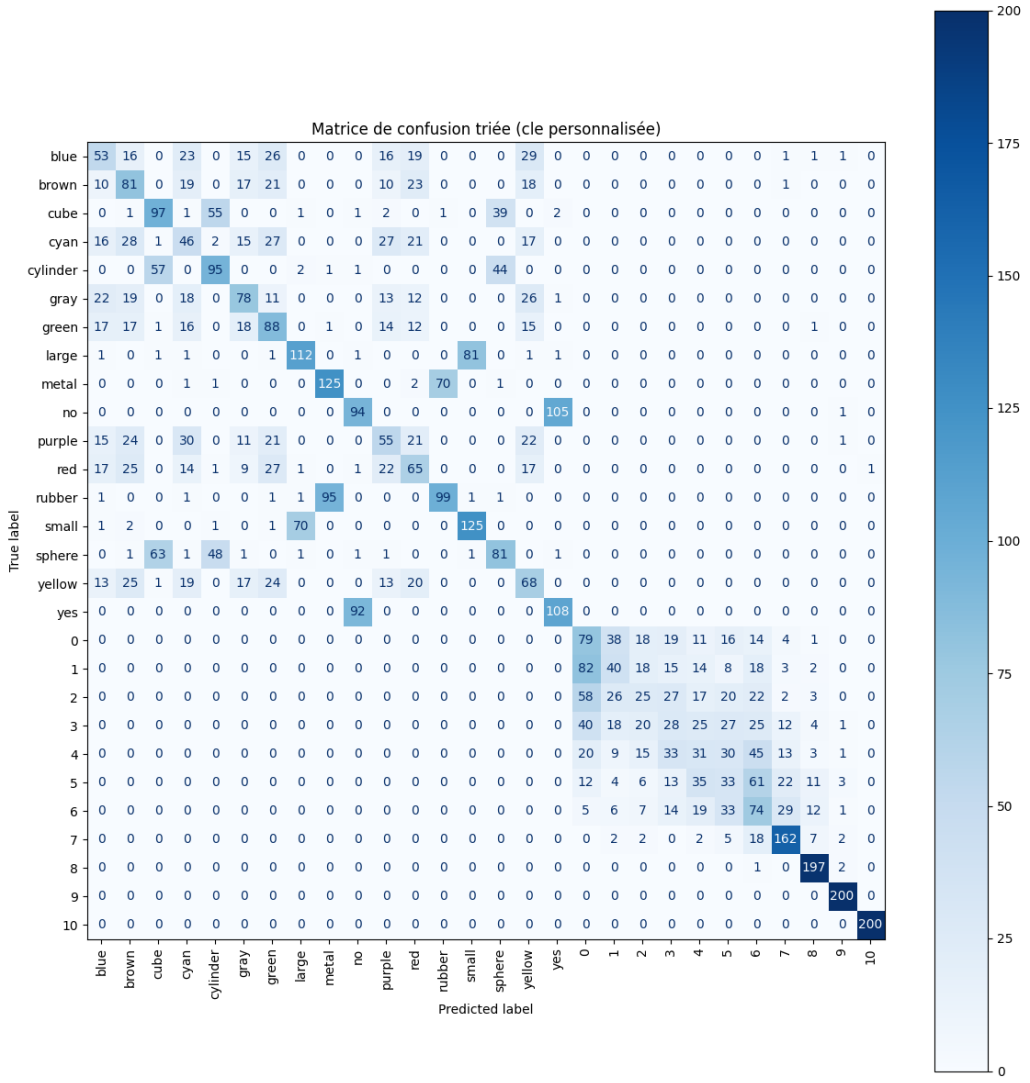


FIG. 3.9 – Matrice de confusion pour le modèle EfficientNet-B0

3.5.5 Discussions

L'analyse conjointe des courbes d'apprentissage (*accuracy* et *loss*) et des matrices de confusion nous permet d'évaluer en profondeur le comportement de chaque modèle sur la tâche de *Visual Question Answering* (VQA) avec le dataset CLEVR.

Tout d'abord, en observant les courbes de précision et de perte, on remarque que les trois modèles atteignent des performances relativement proches, convergeant tous vers une précision comprise entre **45 %** et **47 %** après 10 époques. Le modèle **ResNet18** se distingue légèrement par une précision de validation finale de **46.5 %**, légèrement supérieure à celles de **EfficientNet-B0** (46.2 %) et **MobileNetV2** (45.7 %). Toutefois, cette différence reste marginale et ne constitue pas un avantage décisif en termes de généralisation.

D'un autre côté, la fonction de perte (*loss*) tend à diminuer de manière régulière pour les trois architectures, indiquant une bonne convergence de l'apprentissage. À ce niveau, **MobileNetV2** affiche une perte de validation légèrement plus faible (**1.2474**) que **ResNet18** (1.2575) et **EfficientNet-B0** (1.2598), ce qui suggère une optimisation plus stable, même si cela ne se traduit pas directement par une meilleure précision.

L'étude des *matrices de confusion* complète ces observations. On y remarque que la majorité des classes sont bien reconnues par les trois modèles, notamment les réponses fréquentes comme "yes", "no", ou les couleurs primaires. Néanmoins, certaines confusions persistent entre des classes proches sémantiquement ou visuellement (par exemple, entre les couleurs "cyan", "blue" et "gray", ou encore entre les formes "cube", "sphere" et "cylinder"). Ces confusions reflètent à la fois les similarités entre objets dans les images CLEVR et les limites des modèles à distinguer des concepts fins sans un raisonnement explicite.

Le modèle **EfficientNet-B0** semble légèrement plus robuste dans la distinction des objets liés à des attributs complexes (comme les formes ou les tailles), ce qui peut s'expliquer par sa structure plus avancée et son meilleur rapport entre complexité et performance. Toutefois, cette amélioration reste limitée dans le contexte d'un petit nombre d'époques et d'un dataset artificiel comme CLEVR.

3.6 Conclusion

Dans ce chapitre, nous avons présenté en détail l'étude expérimentale menée dans le cadre de notre système de Visual Question Answering (VQA). Après avoir décrit les étapes de préparation des données, incluant le prétraitement des images et des questions, l'équilibrage des classes, ainsi que les techniques de data augmentation, nous avons introduit l'architecture bimodale utilisée, combinant un CNN pour les images et un LSTM pour les questions.

Nous avons ensuite évalué et comparé les performances de trois architectures de convolution différentes : ResNet18, MobileNetV2 et EfficientNet-B0/B1. Ces modèles ont été testés sur le dataset CLEVR, et leurs performances ont été mesurées à l'aide de métriques standards telles que l'accuracy, la loss, la précision, le rappel, le F1-score, ainsi que les matrices de confusion.

Les résultats obtenus montrent que les trois modèles atteignent des performances similaires, avec une légère supériorité de ResNet18 en termes de précision, et de MobileNetV2 en termes de stabilité de la fonction de perte. EfficientNet-B0/B1, quant à lui, a démontré une bonne capacité de généralisation, en particulier dans la classification de concepts visuellement plus complexes.

Enfin, les visualisations des courbes d'apprentissage et des matrices de confusion ont permis de mieux comprendre les comportements de chaque modèle, leurs forces respectives ainsi que leurs limites face à la complexité du raisonnement visuel et linguistique. Ces observations serviront de base pour affiner les choix architecturaux et explorer de nouvelles pistes d'amélioration dans les chapitres suivants.

Conclusion générale

Le Visual Question Answering (VQA) constitue l'un des domaines les plus actifs et complexes de l'intelligence artificielle, car il exige la compréhension simultanée d'images et de langage naturel. Plusieurs approches ont été proposées dans la littérature, allant des architectures classiques CNN+LSTM jusqu'aux systèmes plus avancés intégrant des mécanismes d'attention, des réseaux Transformers, ou encore des modèles préentraînés comme BERT pour le texte et ViT ou CLIP pour les images.

Dans ce projet, nous avons exploré une approche bimodale fondée sur l'extraction de caractéristiques visuelles via des réseaux CNN (ResNet18, MobileNetV2, EfficientNet-B0/B1) combinés à un LSTM pour modéliser les séquences de mots des questions. Cette structure a été appliquée au dataset CLEVR, qui propose un environnement contrôlé pour l'évaluation de systèmes de raisonnement visuel.

L'ensemble des étapes nécessaires au traitement des données a été soigneusement mis en œuvre : prétraitement des images, tokenisation des questions, conversion des réponses, équilibrage des classes, ainsi que des techniques de régularisation telles que l'ajout de bruit gaussien et la data augmentation. Les performances des trois modèles CNN testés ont été comparées sur plusieurs métriques : précision, perte, F1-score et matrice de confusion.

Les résultats obtenus sont proches entre les architectures, avec une légère supériorité de ResNet18 en précision (46.5%) et une stabilité remarquable de MobileNetV2 en termes de perte (1.2474). Les courbes d'apprentissage ont montré une bonne convergence, et les matrices de confusion ont révélé une capacité générale à reconnaître les classes fréquentes, bien que des confusions subsistent entre des réponses similaires.

Ce projet nous a permis de comprendre en profondeur les mécanismes qui sous-tendent les systèmes VQA et les défis qu'ils posent. Il ouvre des perspectives prometteuses, telles que l'intégration de mécanismes d'attention, l'utilisation de modèles de langage plus puissants (BERT, T5, GPT) et la généralisation vers des bases de données plus complexes du monde réel (comme VQA-v2 ou GQA). L'évolution vers des architectures entièrement basées sur les Transformers ou des approches multimodales préentraînées pourrait également marquer une avancée significative dans les futures versions de notre système.

Références

— Agrawal, A., Batra, D., & Parikh, D. (2016). Analyzing the behavior of visual question answering models. *EMNLP*. https://aclanthology.org/D16-1203

— Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., & Zhang, L. (2018). Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering. *CVPR*. https://doi.org/10.1109/CVPR.2018.00636

— Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C. L., & Parikh, D. (2015). VQA : Visual Question Answering. *ICCV*. https://doi.org/10.1109/ICCV.2015.279

— Canziani, A., Paszke, A., & Culurciello, E. (2016). An Analysis of Deep Neural Network Models for Practical Applications. *arXiv preprint* arXiv :1605.07678. https://arxiv.org/abs/1605.07678

— Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). An Image is Worth 16x16 Words : Transformers for Image Recognition at Scale. *arXiv*. https://arxiv.org/abs/2010.11929

— Dubey, S. R., Singh, S. K., Singh, A. K., & Chaudhuri, B. B. (2018). Local Wavelet Pattern : A New Feature Descriptor for Image Retrieval in the Spatial-Frequency Domain. *IEEE TIP*, 27(8), 3857–3868. https://doi.org/10.1109/TIP.2018.2828880

— Dumoulin, V., & Visin, F. (2016). A guide to convolution arithmetic for deep learning. *arXiv preprint* arXiv :1603.07285. https://arxiv.org/abs/1603.07285

— Gao, H., Mao, J., Zhou, J., Huang, L., Wang, L., & Xu, W. (2015). Are you talking to a machine ? Dataset and methods for multilingual image question answering. *NeurIPS*.

— Gargeya, R., & Leng, T. (2017). Automated Identification of Diabetic Retinopathy Using Deep Learning. *Ophthalmology*, 124(7), 962–969. https://doi.org/10.1016/j.ophtha.2017.02.008

— Geman, D., Geman, S., Hallonquist, N., & Younes, L. (2015). Visual Turing test for computer vision systems. *PNAS*. https://doi.org/10.1073/pnas.1422953112

— Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press. https://www.deeplearningbook.org

— Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. *NeurIPS*, 27. https://papers.nips.cc/paper_files/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html

— He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *CVPR*. https://arxiv.org/abs/1512.03385

— Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780.

— Howard, A. G., et al. (2017). MobileNets : Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv*. https://arxiv.org/abs/1704.04861

— Hudson, D. A., & Manning, C. D. (2019). GQA : A New Dataset for Real-World Visual Reasoning and Compositional Question Answering. *CVPR*, 6700–6709.

— Isensee, F., Jaeger, P. F., Kohl, S. A. A., Petersen, J., & Maier-Hein, K. H. (2021). nnU-Net : a self-configuring method for deep learning-based biomedical image segmentation. *Nature Methods*, 18(2), 203–211. https://doi.org/10.1038/s41592-020-01008-z

— Johnson, J., Hariharan, B., van der Maaten, L., Fei-Fei, L., Zitnick, C. L., & Girshick, R. (2017). CLEVR : A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning. *CVPR*, 2901–2910.

— Kingma, D. P., & Ba, J. (2014). Adam : A Method for Stochastic Optimization. *arXiv*. https://arxiv.org/abs/1412.6980

— Kingma, D. P., & Welling, M. (2014). Auto-Encoding Variational Bayes. *arXiv*. https://arxiv.org/abs/1312.6114

— Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., ... & Fei-Fei, L. (2017). Visual genome : Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 123(1), 32–73.

— Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *NeurIPS*, 25.

— LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444. https://doi.org/10.1038/nature14539

— Lin, M., Chen, Q., & Yan, S. (2014). Network In Network. *ICLR*. https://arxiv.org/abs/1312.4400

— Lu, J., Yang, J., Batra, D., & Parikh, D. (2016). Hierarchical Question-Image Co-Attention for Visual Question Answering. *NeurIPS*, 289–297. https://arxiv.org/abs/1606.00061

— Malinowski, M., & Fritz, M. (2014). A multi-world approach to question answering about real-world scenes based on uncertain input. *NeurIPS*, 1682–1690.

— McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5, 115–133.

— Rawat, W., & Wang, Z. (2017). Deep Convolutional Neural Networks for Image Classification : A Comprehensive Review. *Neural Computation*, 29(9), 2352–2449. https://doi.org/10.1162/neco_a_00990

— Ren, M., Kiros, R., & Zemel, R. S. (2015). Image question answering : A visual semantic embedding model and a new dataset. *NeurIPS*.

— Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088), 533–536.

— Russell, S., & Norvig, P. (2020). *Artificial Intelligence : A Modern Approach* (4th ed.). Pearson.

— Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). MobileNetV2: Inverted Residuals and Linear Bottlenecks. *arXiv*. https://arxiv.org/abs/1801.04381

— Santoro, A., Raposo, D., Barrett, D. G. T., Malinowski, M., Pascanu, R., Battaglia, P., & Lillicrap, T. (2017). A simple neural network module for relational reasoning. *NeurIPS*.

— Schmidhuber, J. (2015). Deep learning in neural networks : An overview. *Neural Networks*, 61, 85–117.

— Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... & Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587), 484–489. https://doi.org/10.1038/nature16961

— Simonyan, K., & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. *ICLR*. https://arxiv.org/abs/1409.1556

— Tan, M., & Le, Q. V. (2019). EfficientNet : Rethinking Model Scaling for Convolutional Neural Networks. *ICML*. https://arxiv.org/abs/1905.11946

— Tan, M., & Le, Q. V. (2020). EfficientNetV2: Smaller Models and Faster Training. *ICML*. https://arxiv.org/abs/2104.00298

— Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *NeurIPS*. https://arxiv.org/abs/1706.03762

— Zhang, Y., Goyal, Y., Summers-Stay, D., Batra, D., & Parikh, D. (2016). Yin and Yang : Balancing and answering binary visual questions. *CVPR*, 5014–5022.

— Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2016). Learning Deep Features for Discriminative Localization. *CVPR*, 2921–2929. https://doi.org/10.1109/CVPR.2016.319