

## Projet de Deep Learning

Master Sciences et Techniques  
en Intelligence Artificielle et Informatique Digitale

### Titre

Deep learning for video captioning

Réalisé par :

-Mimoun Ouhda -Walid Ait Moussa -Yassine Boujnan

Sous l'encadrement de :

Pr. SAADI Youssef

Presentez le **21 juin 2025** devant le jury composé de :

Mr. SAADI Youssef  
Mr. EL MOURABIT yousef  
Mr. ZOUGAGH Hicham

Professeur Chercheur  
Professeur Chercheur  
Professeur Chercheur

FST de Béni Mellal  
FST de Béni Mellal  
FST de Béni Mellal

## TABLE DES MATIÈRES

<b>1</b>	<b>Introduction Générale</b>	<b>4</b>
1.1	Contexte et Motivation : . . . . .	4
1.2	Problématique : . . . . .	5
1.3	Objectifs du Projet : . . . . .	6
1.4	Structure du Rapport : . . . . .	8
<b>2</b>	<b>Compréhension du Contenu Vidéo Fondamentaux</b>	<b>9</b>
2.0.1	Qu'est-ce qu'une Vidéo ? . . . . .	9
2.0.2	Richesse de l'information vidéo . . . . .	10
2.1	Caractéristiques et Représentations Vidéo : . . . . .	11
2.1.1	Représentations Visuelles : L'Extraction de Caractéristiques Spatiales . .	11
2.1.2	Représentations Temporelles/Mouvement : L'Importance de la Dynamique Vidéo . . . . .	14
2.2	Défis de l'Analyse Vidéo . . . . .	17
<b>3</b>	<b>Introduction au Deep Learning pour le Traitement Séquentiel et Multimodal</b>	<b>20</b>
3.1	Principes Fondamentaux du Deep Learning . . . . .	20
3.2	Introduction au Deep Learning pour le Traitement Séquentiel et Multimodal . .	20
3.2.1	Principes Fondamentaux du Deep Learning . . . . .	20
3.3	Réseaux de Neurones Convolutifs (CNN) . . . . .	24
3.3.1	Le Rôle Stratégique des CNN dans l'Extraction de Caractéristiques Spatiales . . . . .	24
3.3.2	Opérations Fondamentales Constituant un CNN . . . . .	25
3.3.3	L'Essence "Profonde" des CNN : L'Apprentissage Hiérarchique de Représentations Visuelles . . . . .	28
3.3.4	Architectures Populaires de CNN et leur Utilisation comme Extracteurs de Caractéristiques (Transfert d'Apprentissage) . . . . .	28
3.4	Réseaux de Neurones Récurrents (RNN) . . . . .	31
3.4.1	Problèmes Inhérents aux RNN Simples : Vanishing/Exploding Gradients	33
3.5	Mécanismes d'Attention ( <i>Attention Mechanisms</i> ) . . . . .	39
3.5.1	Principe Fondamental des Mécanismes d'Attention . . . . .	39
3.5.2	Le Rôle Crucial des Mécanismes d'Attention dans le Sous-titrage Vidéo .	40
3.5.3	Types d'Attention dans les Systèmes de Sous-titrage Vidéo . . . . .	41
3.6	Modèles Encodeur-Décodeur ( <i>Encoder-Decoder Models</i> ) . . . . .	42
3.6.1	Explication Détaillée : Le Mécanisme Encodeur-Décodeur . . . . .	42
3.6.2	Le Paradigme Encodeur-Décodeur Appliqué au Sous-titrage Vidéo . . . .	45

<b>4 Architectures de Deep Learning pour le Sous-Titrages Vidéo</b>	<b>47</b>
4.1 Approches Générales du Video Captioning : Décoder le Récit Visuel . . . . .	47
4.1.1 "Show and Tell" (Encoder-Decoder) : Les Architectures Pionnières . . . . .	47
4.1.2 Approches Basées sur l'Attention : Rendre les Descriptions Pertinentes . . . . .	49
4.2 Architectures de Décodage Textuel : Transformer la Vision en Langage . . . . .	50
4.2.1 RNN/LSTM/GRU : La Génération Séquentielle de Mots . . . . .	50
4.2.2 Mécanismes d'Attention : Moduler le Décodeur pour un Focus Pertinent	51
4.3 Architectures Avancées et Tendances Actuelles : Repousser les Limites du Sous-titrage Vidéo . . . . .	52
4.3.1 Transformers pour le Video Captioning : L'Ascension de l'Auto-Attention	52
4.3.2 Approches Multimodales : Vers une Compréhension Enrichie . . . . .	53
4.3.3 Génération de Descriptions Diversifiées : Au-delà d'une Unique Vérité .	54
<b>5 Bases de Données et Métriques d'Évaluation</b>	<b>56</b>
5.1 Bases de Données Communes pour le Video Captioning : Le Carburant de l'Apprentissage . . . . .	56
5.1.1 MSVD (Microsoft Research Video Description Corpus) . . . . .	56
5.1.2 MSR-VTT (Microsoft Research Video to Text) . . . . .	56
5.1.3 VATEX (Video and Text) . . . . .	57
5.1.4 LSMDC (Large-Scale Movie Description Challenge) . . . . .	57
5.1.5 Défis Associés à Ces Bases de Données . . . . .	58
5.2 Métriques d'Évaluation des Sous-Titrages : Mesurer la Qualité de la Narration Automatique . . . . .	58
5.2.1 BLEU (Bilingual Evaluation Understudy) : Le Pionnier . . . . .	59
5.2.2 METEOR (Metric for Evaluation of Translation With Explicit Ordering) : L'Amélioration du Rappel . . . . .	59
5.2.3 ROUGE (Recall-Oriented Understudy for Gisting Evaluation) : L'Évaluation Axée sur le Rappel . . . . .	60
5.2.4 CIDEr (Consensus-based Image Description Evaluation) : Spécifique à la Description Visuelle . . . . .	61
5.2.5 SPICE (Semantic Propositional Image Captioning Evaluation) : L'Évaluation Sémantique . . . . .	62
5.2.6 L'Importance d'Utiliser Plusieurs Métriques pour une Évaluation Complète	62
<b>6 Architecture et Implémentation du Modèle de Video Captioning</b>	<b>64</b>
6.1 Acquisition et Structuration du Jeu de Données MSR-VTT : Le Cœur de notre Apprentissage Multimodal . . . . .	64
6.2 Architecture du Modèle : Le Système Encodeur-Décodeur et le Rôle Crucial de l'Attention . . . . .	68
6.2.1 Architecture ResNet50 et Adaptation au Légendage Vidéo . . . . .	74
6.3 Analyse Déttaillée des Résultats et Métriques de Performance du Modèle . . . . .	75
6.3.1 Évolution de la Fonction de Perte (Loss) . . . . .	75
6.3.2 Évolution de l'Exactitude (Accuracy) . . . . .	76
6.3.3 Analyse de la Métrique BLEU-4 . . . . .	76
6.4 Interface du Générateur de Légendes Vidéo par IA . . . . .	77
6.4.1 Page d'Accueil et Sélection de Modèle . . . . .	78
6.4.2 Résultats de Légendage Global . . . . .	78
6.4.3 Résultats de Légendage Segmenté . . . . .	79
6.4.4 Discussion Comparative . . . . .	81

---

<b>7 Conclusion Générale</b>	<b>82</b>
Annexe . . . . .	85

# CHAPITRE 1

## INTRODUCTION GÉNÉRALE

### 1.1 Contexte et Motivation :

Dans le paysage numérique actuel, le contenu vidéo s'est imposé comme le médium dominant pour la diffusion d'informations, le divertissement, l'éducation et la communication. Selon les dernières études, plus de **500 heures de vidéo** sont téléchargées chaque minute sur la seule plateforme YouTube, tandis que des services comme Netflix, TikTok et les systèmes de vidéosurveillance génèrent un flux continu de données visuelles. Cette explosion du contenu vidéo a créé un défi majeur : comment analyser, organiser et comprendre efficacement cette masse croissante de données multimédias ?

#### Limites des méthodes traditionnelles

Les approches conventionnelles d'indexation vidéo révèlent des insuffisances criantes face à cette nouvelle réalité :

- **Balises manuelles (tags)** : Processus subjectif et chronophage où des opérateurs humains attribuent des mots-clés. Une même vidéo montrant "un chat jouant avec une balle" pourrait être taguée différemment ("animal", "jouet", "compagnon") selon l'interprétation de chaque annotateur.
- **Descriptions textuelles** : La rédaction manuelle de résumés ou de métadonnées descriptives ne peut suivre le rythme de production actuel. Un documentaire d'une heure peut nécessiter plusieurs heures de travail pour être correctement résumé.
- **Métadonnées techniques** : Les informations basiques (date, format, résolution) ne fournissent aucune indication sur le contenu sémantique de la vidéo.

Ces méthodes présentent trois lacunes fondamentales :

1. Elles ne sont pas **évolutives** face au volume exponentiel de vidéos
2. Elles manquent de **précision** dans la description du contenu
3. Elles ignorent la **dimension temporelle** des séquences vidéo

#### Enjeux du sous-titrage automatique

Le *Video Captioning* par apprentissage profond répond à ces défis par une triple révolution :

**Accessibilité** Permet aux personnes malentendantes d'accéder au contenu (conformément aux lois comme l'ADA aux États-Unis). Exemple : les sous-titres automatiques sur YouTube ont accru de 30% l'engagement des utilisateurs sourds.

**Recherche sémantique** Transforme la recherche vidéo en permettant des requêtes complexes comme "trouver le moment où le personnage principal entre dans la pièce" au lieu de simples mots-clés.

**Interaction homme-machine** Donne aux systèmes IA la capacité de comprendre et décrire des environnements visuels complexes, crucial pour :

- La vidéosurveillance intelligente
- Les assistants personnels
- Les véhicules autonomes

Les avancées récentes en réseaux neuronaux profonds (3D-CNN, Transformers multimodaux) ont permis des progrès spectaculaires, avec des systèmes capables aujourd’hui de générer des descriptions comme "*Un enfant lance une balle à son chien dans un parc ensoleillé, tandis qu’un couple s’assoit sur un banc à l’arrière-plan*". Pourtant, des défis majeurs persistent, notamment dans la compréhension fine du contexte et la gestion des séquences longues, ce qui ouvre un vaste champ de recherche pour les années à venir.

## 1.2 Problématique :

La problématique fondamentale à laquelle ce projet cherche à répondre est la suivante : Comment peut-on, de manière automatisée, générer des descriptions textuelles à la fois précises, sémantiquement riches et contextuellement pertinentes à partir de séquences vidéo dynamiques ?

Cette question, simple en apparence, cache une complexité considérable qui la distingue radicalement des tâches de traitement d’images ou de texte isolées. Le défi réside dans la capacité d’une machine à imiter la cognition humaine, qui non seulement identifie ce qui est visible (objets, personnes, environnements), mais aussi ce qui se passe (actions, événements, interactions) et le pourquoi cela se passe, en le restituant dans un langage naturel et cohérent.

Pour mieux saisir l’ampleur de cette problématique, il est crucial d’identifier et d’analyser les défis inhérents à cette tâche :

### La Temporalité : Gérer la Dynamique et la Séquence des Événements

Une vidéo n’est pas une simple succession d’images statiques ; c’est une entité dynamique où les événements se déroulent et évoluent au fil du temps. Le système de sous-titrage doit donc être capable de :

- **Capturer le mouvement et les changements d’état** : Il ne suffit pas de détecter une "balle" et un "joueur", mais de comprendre que le "joueur frappe la balle" ou que la "balle roule".
- **Modéliser les dépendances temporelles à long terme** : Des actions importantes peuvent s’étendre sur de longues périodes ou dépendre d’événements survenus beaucoup plus tôt dans la vidéo. Par exemple, décrire la "célébration du but" nécessite d’avoir identifié le "but" lui-même, qui a pu se produire plusieurs secondes auparavant. Les modèles doivent être capables de maintenir un "contexte" temporel.
- **Comprendre l’ordre des actions** : "La voiture heurte le mur" n’est pas la même chose que "le mur heurte la voiture". L’ordre chronologique des actions est vital pour la précision de la description.

## La Multimodalité : Fusionner l'Information Visuelle, Temporelle et (Potentielle-ment) Sonore

Les vidéos sont intrinsèquement multimodales. Une description complète et pertinente doit intégrer les informations provenant de diverses sources :

- **Information visuelle (spatiale)** : Reconnaître les objets, les scènes, les personnes, leurs attributs (couleur, taille, position). Cela nécessite des capacités de classification et de détection d'objets robustes.
- **Information temporelle (mouvement)** : Comprendre les actions et les interactions entre les entités au fil du temps. Cela implique de suivre les trajectoires, de quantifier les vitesses et de reconnaître les gestes ou les activités.
- **Information sonore (optionnel mais pertinent)** : Bien que souvent mise de côté dans les approches de base, l'information audio (paroles, bruits d'ambiance, musique) peut enrichir considérablement la compréhension contextuelle et la précision du sous-titrage. Par exemple, le son d'une "explosion" ou d'une "ovation" peut aider à mieux décrire une scène. Le défi ici est de développer des mécanismes de fusion efficaces qui permettent aux modèles d'intégrer harmonieusement ces différentes sources d'information pour former une représentation cohérente et riche du contenu vidéo.

## La Sémantique : Aller Au-Delà de la Détection pour Atteindre la Compréhension

La tâche de sous-titrage vidéo dépasse la simple énumération d'objets ou d'actions détectées. Elle exige une compréhension sémantique profonde et la capacité à générer un langage naturel :

- **Compréhension conceptuelle de haut niveau** : Le modèle doit inférer le sens derrière les pixels et les mouvements. Par exemple, au lieu de juste "homme", "table", "ordinateur", il doit pouvoir générer "un homme travaille sur son ordinateur portable à une table".
- **Maîtrise du langage naturel** : La description générée doit être grammaticalement correcte, fluide, et naturelle. Cela implique de choisir les bons verbes, les bons adjectifs, et d'organiser les mots en phrases cohérentes.
- **Gestion de l'ambiguïté et du contexte** : Une même action peut avoir différentes significations selon le contexte. Par exemple, "courir" peut être une activité sportive, une fuite, ou un mode de déplacement. Le modèle doit être capable de déduire le contexte pertinent.
- **Génération de descriptions pertinentes et discriminantes** : Le système ne doit pas seulement générer une description "valide", mais la meilleure description qui capture l'essence de ce qui est unique dans la vidéo, évitant les descriptions trop génériques ou incomplètes.

### 1.3 Objectifs du Projet :

L'objectif cardinal de ce projet transcende la simple implémentation technologique pour viser la concrétisation d'une capacité cognitive : développer un système de sous-titrage vidéo sophistiqué et autonome, ancré sur les principes et les architectures de pointe du Deep Learning, capable de déchiffrer le contenu visuel et temporel d'une vidéo pour le traduire en descriptions textuelles riches, pertinentes et d'une fluidité naturelle.

Cet objectif n'est pas qu'une aspiration technique ; c'est une quête pour doter les machines d'une forme d'intelligence narrative. En exploitant la puissance inégalée du Deep Learning, nous cherchons à surmonter les limitations des approches classiques et à adresser directement les défis de temporalité, de multimodalité et de sémantique précédemment évoqués. Le Deep Learning offre la capacité d'apprendre des représentations hiérarchiques complexes directement à partir des données brutes, une condition *sine qua non* pour la compréhension nuancée du contenu vidéo et la génération de langage humain.

Pour matérialiser cette vision ambitieuse, nous nous sommes fixés des objectifs spécifiques et mesurables, chacun étant une pierre angulaire de la réussite de ce projet :

### **Exploration Approfondie et Comparative des Modèles d'Apprentissage Profond de Pointe**

Il ne s'agit pas simplement d'appliquer un modèle existant, mais d'entreprendre une revue critique et une analyse comparative des architectures de Deep Learning les plus prometteuses dans le domaine du traitement des données séquentielles et multimodales.

- Cela inclut une immersion détaillée dans les **Réseaux de Neurones Convolutifs (CNN)** pour leur excellence dans l'extraction de caractéristiques spatiales (objets, scènes, visages) à partir des images vidéo.
- Nous nous pencherons également sur les **Réseaux de Neurones Récurrents (RNN)**, en particulier les variantes avancées comme les **Long Short-Term Memory (LSTM)** et les **Gated Recurrent Unit (GRU)**, pour leur capacité intrinsèque à modéliser les dépendances temporelles dans les séquences vidéo et à générer des séquences de texte cohérentes.
- Une attention particulière sera accordée aux **mécanismes d'attention**, véritables phares dans le brouillard des données, qui permettent aux modèles de concentrer leurs ressources computationnelles sur les segments les plus pertinents de la vidéo lors de la génération de chaque mot.
- Enfin, l'étude s'étendra aux architectures **Transformer**, révolutionnaires dans le domaine du Traitement du Langage Naturel (TLN) et de la vision, pour évaluer leur potentiel à capturer des relations à longue portée et à gérer la multimodalité de manière plus efficace que les architectures récurrentes traditionnelles. L'objectif est de choisir, ou de proposer, l'architecture la plus synergique et performante pour la tâche de sous-titrage vidéo.

### **Maîtrise et Optimisation des Techniques d'Encodage Vidéo et de Décodage Textuel**

- Ce projet vise à comprendre et à affiner les stratégies permettant de transformer une séquence vidéo brute en une représentation latente compacte et sémantiquement riche (encodage vidéo). Cela implique l'étude des différentes méthodes d'extraction de caractéristiques visuelles (par exemple, des caractéristiques pré-entraînées de CNN 2D ou des CNN 3D pour la capture spatio-temporelle) et de leur agrégation sur la dimension temporelle.
- Parallèlement, nous nous concentrerons sur le processus de décodage textuel, c'est-à-dire comment cette représentation vidéo latente est ensuite convertie, mot par mot, en une

phrase grammaticalement correcte et contextuellement précise. L'objectif est d'explorer les algorithmes de recherche (comme le *beam search*) et les fonctions de perte qui maximisent la qualité et la diversité des sous-titres générés, assurant une transition fluide et intelligente du "voir" au "dire".

### **Établissement d'une Méthodologie Rigoureuse d'Évaluation des Performances**

- Un aspect crucial du projet est de ne pas se contenter de générer des sous-titres, mais de les évaluer de manière objective et quantitative. Cela implique la familiarisation et l'application systématique des métriques d'évaluation standard reconnues par la communauté scientifique (telles que **BLEU**, **METEOR**, **ROUGE**, **CIDEr** et **SPICE**).
- L'objectif est d'analyser non seulement la précision syntaxique et le chevauchement de mots, mais aussi la pertinence sémantique et la cohérence narrative des descriptions générées. Cette évaluation rigoureuse permettra de comparer les performances des différentes architectures explorées, d'identifier leurs forces et faiblesses, et de valider l'efficacité du système proposé face à l'état de l'art.
- Au-delà des métriques, une évaluation qualitative par des observateurs humains sera également envisagée pour s'assurer que les descriptions générées sont naturelles et fidèles à l'expérience humaine de compréhension vidéo.

## **1.4 Structure du Rapport :**

Ce rapport théorique est conçu pour offrir une exploration structurée et progressive du domaine du sous-titrage automatique de vidéos par Deep Learning. Il s'articule autour des sections suivantes :

**Section 2 Compréhension du Contenu Vidéo :** Établit les fondations en détaillant la nature du contenu vidéo, ses caractéristiques intrinsèques et les défis fondamentaux liés à son analyse.

**Section 3 Deep Learning pour le Traitement Séquentiel et Multimodal :** Introduit les concepts clés du Deep Learning, se concentrant sur les architectures essentielles (CNN, RNN, LSTM, GRU, mécanismes d'attention, et modèles Encodeur-Décodeur) qui sont cruciales pour le traitement des données séquentielles et multimodales.

**Section 4 Architectures de Deep Learning pour le Sous-Titrage Vidéo :** Plonge au cœur des architectures spécifiquement conçues pour le sous-titrage vidéo, examinant les approches d'encodage vidéo, de décodage textuel, ainsi que les tendances et architectures avancées comme les Transformers.

**Section 5 Bases de Données et Métriques d'Évaluation :** Se consacre aux aspects pratiques de l'évaluation, présentant les principales bases de données utilisées pour le développement et la validation des systèmes de sous-titrage, ainsi que les métriques standard pour mesurer leur performance.

**Section 6 Conclusion et Perspectives :** Conclut le rapport en récapitulant les concepts fondamentaux, en identifiant les défis actuels et les pistes de recherche.

**Section 7 Approche Proposée :** Détaille notre approche pour la conception du système de sous-titrage vidéo, en présentant l'architecture et les choix méthodologiques envisagés.

**Section 8 Implémentation :** Trace les grandes lignes de l'implémentation du projet, incluant l'environnement de développement et les étapes pratiques.

**Section 9 Références :** Liste toutes les références bibliographiques utilisées pour la rédaction de ce rapport.

## CHAPITRE 2

### COMPRÉHENSION DU CONTENU VIDÉO FONDAMENTAUX

#### 2.0.1 Qu'est-ce qu'une Vidéo ?

Une vidéo, à son niveau le plus élémentaire, est une illusion de mouvement créée par la présentation rapide d'une séquence d'images fixes. Imaginez un flipbook : chaque page est une image légèrement différente de la précédente. Lorsqu'on feuilleter rapidement les pages, notre cerveau interprète la succession d'images comme un mouvement continu. C'est exactement le principe d'une vidéo.

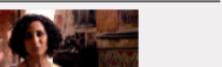
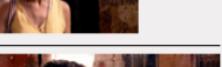
Format	Entrelacé/ Progressif	Rapport largeur/ hauteur en pixels	Représentation (pixels virtuels)
<b>SD PAL</b>	e, p	720 × 576* (5:4)	<b>4:3</b> (768 × 576) 
	Anamorphosé	720 × 576	<b>16:9</b> (1024 × 576)  
	Letterbox	720 × 434	<b>16:9</b> (1024 × 576)  
<b>SD NTSC</b>	e, p	640 × 480** (4:3)	<b>4:3</b> (640 × 480) 
	Standard plus moderne	720 × 480 (3:2)	<b>4:3</b> (640 × 480)  
<b>HD</b> «Full HD»	e, p	1920 × 1080 (16:9)	<b>16:9</b> (1920 × 1080)  
<b>HD</b>	p	1280 × 720 (16:9)	<b>16:9</b> (1280 × 720)  
<b>HDV</b> «Full HD» anamorphosé	e	1440 × 1080 (4:3)	<b>16:9</b> (1920 × 1080)  

FIGURE 2.1 – Illustration de la décomposition temporelle d'une vidéo montrant les frames à différents intervalles (0ms, 200ms, 400ms, 800ms). Cette séquence démontre comment la succession rapide d'images fixes crée l'illusion du mouvement.

**Composition technique** Chaque image individuelle dans une vidéo est appelée un « frame » ou un « cadre ». Ces frames sont capturées à des intervalles réguliers, et la vitesse à laquelle ils sont affichés détermine la fluidité du mouvement perçu. Cette vitesse est mesurée en **images par seconde** (FPS, *Frames Per Second*).

- Un FPS élevé (par exemple, 60 FPS) donne un mouvement très fluide et réaliste
- Un FPS plus faible (par exemple, 15 FPS) peut rendre le mouvement saccadé
- Standards courants :
  - 24 FPS (cinéma)
  - 30 FPS (télévision)
  - 60 FPS (jeux vidéo, ralenti)

**Dimension temporelle** L'aspect fondamental qui distingue une vidéo d'une simple image est sa dimension temporelle.

- Une **image** capture un instant figé, un seul point dans le temps
- Une **vidéo** enregistre un flux d'événements qui se déroulent sur une durée

Ce flux temporel est crucial pour notre compréhension du monde : il nous permet de voir des actions se dérouler, des objets se déplacer, des scènes changer, et des histoires se raconter.

## 2.0.2 Richesse de l'information vidéo

La richesse de l'information contenue dans une vidéo peut être décomposée en plusieurs couches :

**Information Visuelle (Spatiale)** Chaque frame est une image complexe, contenant :

- **Objets** : Personnes, animaux, véhicules, bâtiments, etc.
- **Scènes** : Intérieurs, extérieurs, paysages, etc.
- **Attributs** : Couleurs, textures, formes, tailles, positions relatives
- **Éclairage** : Luminosité, ombres, reflets
- **Composition** : Agencement des éléments visuels dans le cadre

**Information de Mouvement (Temporelle)** Ce qui donne vie à la vidéo :

- **Déplacements** : Trajectoires des objets et des personnes
- **Actions** : Verbes (courir, sauter, frapper, ouvrir, fermer)
- **Gestes** : Mouvements spécifiques du corps
- **Transitions** : Changements de plan (coupes, fondus)

**Information Sonore (Optionnelle)** Bien que non essentielle pour le sous-titrage :

- **Dialogues** : Paroles prononcées
- **Musique** : Ambiance, émotions
- **Bruits** : Sons d'impact, bruits d'ambiance

**Compréhension par le Deep Learning** Pour un système de Deep Learning, « comprendre » une vidéo signifie :

- Décoder ces couches d'information
- Relier les éléments entre eux
- Saisir les interactions, mouvements et évolutions temporelles
- Aller au-delà de la simple reconnaissance d'objets isolés

## 2.1 Caractéristiques et Représentations Vidéo :

### 2.1.1 Représentaions Visuelles : L'Extraction de Caractéristiques Spatiales

Pour qu'une machine puisse "comprendre" le contenu d'une vidéo, elle ne peut pas travailler directement avec les pixels bruts. Ces données sont trop brutes et trop sensibles aux variations (lumière, angle, échelle) pour capturer des informations de haut niveau comme un objet ou une action. L'étape essentielle consiste donc à extraire des caractéristiques (*features*), c'est-à-dire des représentations compactes et significatives des informations visuelles présentes dans chaque image (*frame*) de la vidéo. Ces caractéristiques doivent être robustes aux variations non pertinentes et mettre en évidence les aspects discriminants de l'image.

Historiquement, l'extraction de caractéristiques visuelles a évolué depuis des méthodes manuelles, où les algorithmes étaient explicitement conçus par des experts, vers des approches automatiques basées sur l'apprentissage profond.

#### Descripteurs Traditionnels (Ingénierie de Caractéristiques Manuelle)

Pendant des décennies, le traitement d'images et la vision par ordinateur ont reposé sur des algorithmes conçus pour détecter des motifs spécifiques ou des points d'intérêt. Ces méthodes, souvent appelées "ingénierie de caractéristiques manuelle", se concentrent sur des propriétés locales de l'image.

**SIFT (Scale-Invariant Feature Transform)** Principe : SIFT est un algorithme puissant qui détecte des points d'intérêt (*keypoints*) distinctifs dans une image. Ces points sont remarquables car ils restent invariants face aux changements d'échelle, de rotation et sont partiellement robustes aux variations d'éclairage. Pour chaque *keypoint* détecté, SIFT calcule un descripteur qui représente les orientations de gradient de pixels dans sa région voisine.

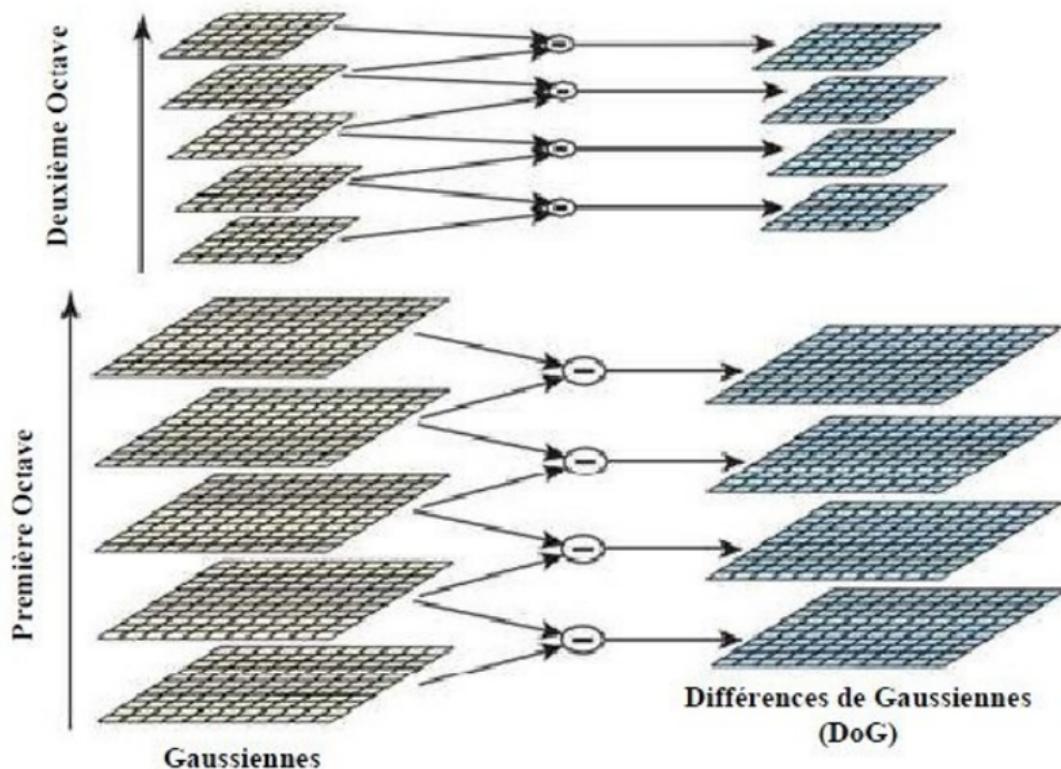


FIGURE 2.2 – Processus de calcul des caractéristiques SIFT : construction d'un espace d'échelle gaussien et extraction d'un vecteur de caractéristiques de 128 dimensions basé sur les orientations de gradient.

**Méthode :** Le processus commence par la détection des extrema dans un espace d'échelle (à l'aide de différences de Gaussiennes), suivie par un filtrage des points peu stables. Autour de chaque point sélectionné, des histogrammes d'orientations de gradient sont construits, résultant en un vecteur de 128 dimensions décrivant la "texture" locale.

**Utilité :** Bien que conçu pour les images, SIFT peut s'appliquer frame par frame dans une vidéo pour identifier des repères visuels stables, utiles pour le suivi d'objets ou la reconnaissance de scènes à travers différentes vues.

**HOG (Histogram of Oriented Gradients)** **Principe :** HOG est un descripteur de forme qui capture l'apparence d'un objet en quantifiant les occurrences des orientations de gradient de pixels dans des régions localisées de l'image. L'idée est que la forme d'un objet peut être efficacement caractérisée par la distribution des intensités des contours (gradients).

**Méthode :** L'image est divisée en petites cellules. Pour chaque cellule, un histogramme des orientations de gradient est calculé. Ces histogrammes sont ensuite regroupés et normalisés sur des blocs plus grands pour compenser les variations d'éclairage. La concaténation de tous ces histogrammes forme le descripteur final.

**Utilité :** HOG a été très populaire pour la détection de personnes et de certains objets dans des vidéos, grâce à sa robustesse aux déformations et aux variations de pose.

### Caractéristiques Apprises (Deep Learning - CNN Features)

L'avènement du Deep Learning, et en particulier des Réseaux de Neurones Convolutifs (CNN), a révolutionné l'extraction de caractéristiques visuelles. Contrairement aux méthodes manuelles, les CNN sont capables d'apprendre automatiquement des représentations hiérar-

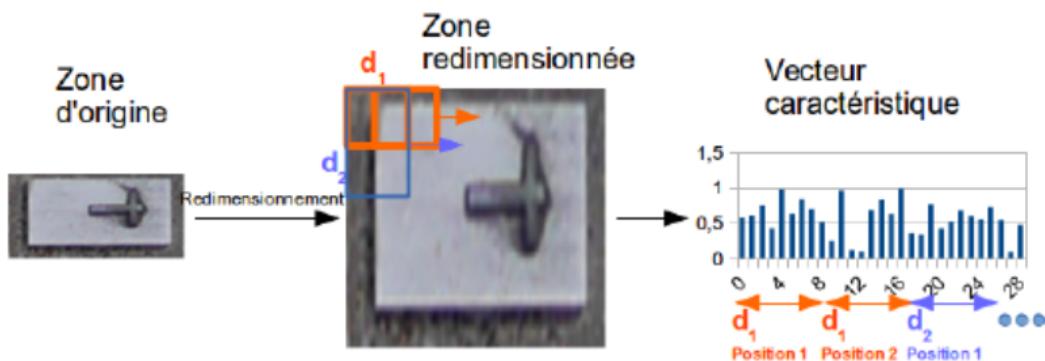


FIGURE 2.3 – Calcul des caractéristiques HOG : division de l'image en cellules, calcul d'histogrammes d'orientation par cellule, et normalisation par blocs.

chiques et de plus en plus abstraites directement à partir des données brutes (les pixels).

**Principe Fondamental** Un CNN est une architecture composée de multiples couches qui effectuent des opérations de convolution, de pooling et des transformations non linéaires. Les premières couches du réseau apprennent à détecter des caractéristiques de bas niveau, comme les bords, les coins et les textures. À mesure que l'information progresse à travers les couches plus profondes, le réseau apprend à combiner ces caractéristiques simples pour former des représentations de plus haut niveau, telles que des parties d'objets (une roue, un œil), puis des objets entiers (une voiture, un visage) et même des scènes complexes (une plage, un intérieur de maison).

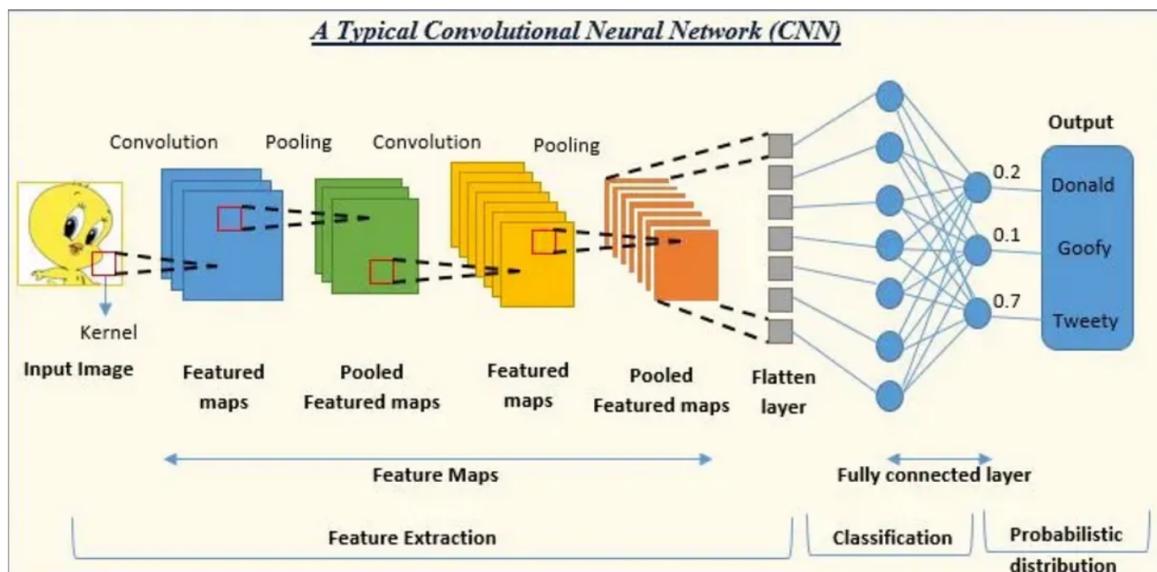


FIGURE 2.4 – Architecture typique d'un CNN montrant les couches de convolution, de pooling et les couches fully connected pour la classification.

**Caractéristiques issues de CNN pré-entraînés (Feature Extractors - "Transfer Learning")** Concept : L'une des applications les plus puissantes et efficaces des CNN pour l'extraction de caractéristiques est l'utilisation de modèles déjà pré-entraînés sur d'énormes jeux de données d'images, comme ImageNet. ImageNet contient des millions d'images réparties en

milliers de catégories, permettant à ces modèles (tels que VGG, ResNet, Inception, EfficientNet, DenseNet) d'apprendre à reconnaître un éventail extrêmement large et générique de motifs visuels.

**Processus d'Extraction :** Pour obtenir les caractéristiques d'une frame vidéo, cette frame est simplement passée à travers le CNN pré-entraîné. On récupère alors la sortie d'une des dernières couches du réseau (généralement juste avant la couche de classification finale). Ce vecteur de sortie, souvent de plusieurs centaines ou milliers de dimensions, est une représentation dense, compacte et sémantiquement riche de l'image. Il encode des informations de haut niveau sur le contenu visuel (par exemple, la présence d'un chat, l'activité d'une personne, le type de paysage).

Avantages Cruciaux pour le Video Captioning :

- **Apprentissage Automatique Profond** : Élimine le besoin d'ingénierie manuelle de caractéristiques, un processus complexe et fastidieux.
- **Généralisation et Robustesse** : Les caractéristiques apprises sur de vastes ensembles de données sont exceptionnellement génériques et performantes, transférables à diverses tâches visuelles. Elles sont également très robustes aux variations telles que l'occlusion partielle, les changements de pose, les déformations ou les variations d'éclairage.
- **Haute Abstraction Sémantique** : Ces caractéristiques capturent des informations de plus haut niveau que les descripteurs traditionnels, ce qui est directement pertinent pour la compréhension d'objets, d'actions et de scènes complexes nécessaires au sous-titrage.

**Application au Video Captioning :** Pour le sous-titrage vidéo, les caractéristiques visuelles issues de ces CNN pré-entraînés sont extraites pour chaque frame clé ou pour un sous-ensemble de frames de la vidéo. Ces vecteurs de caractéristiques temporels sont ensuite transmis à la partie séquentielle du modèle (généralement un Réseau de Neurones Récursifs ou une architecture Transformer) pour que le système puisse non seulement identifier les éléments visuels, mais aussi comprendre leur évolution et leurs relations dans le temps afin de générer une description textuelle cohérente.

### 2.1.2 Représentations Temporelles/Mouvement : L'Importance de la Dynamique Vidéo

Au-delà de la capture des caractéristiques visuelles statiques de chaque image, la dimension temporelle est ce qui donne vie à une vidéo. Comprendre le mouvement est absolument fondamental pour le sous-titrage vidéo, car les verbes d'action (*"courir"*, *"sauter"*, *"parler"*, *"ouvrir"*) et les interactions entre objets ou personnes sont intrinsèquement liés à la dynamique temporelle. Une description pertinente ne peut se limiter à une liste d'objets présents ; elle doit expliquer *ce qui se passe*.

L'extraction de caractéristiques temporelles vise à capturer ces changements et flux d'information entre les frames successives.

#### Le Flux Optique (Optical Flow)

**Principe** Le flux optique est une représentation du mouvement apparent des objets, des surfaces et des contours d'une image à l'autre dans une séquence vidéo. Il calcule le déplacement de chaque pixel (ou de groupes de pixels) entre deux frames consécutives. Imaginez que vous suivez un point sur la surface d'un objet en mouvement ; le flux optique vous donne la direction et la magnitude de ce déplacement.



**FIGURE 2.5 – Visualisation du flux optique (Optical Flow) pour la vidéosurveillance intelligente.**  
À gauche : détection et suivi du mouvement des véhicules dans une séquence vidéo de trafic.  
À droite : carte de mouvement (motion map) où les couleurs représentent les directions et les vitesses du flux optique.

**Méthode** Les algorithmes de flux optique (comme Lucas-Kanade ou Farnebäck) analysent les variations d'intensité des pixels pour estimer ces vecteurs de mouvement. Le résultat est généralement une « carte » de vecteurs de mouvement, où la longueur et la direction de chaque vecteur indiquent la vitesse et la direction du mouvement à ce point de l'image.

### Utilité pour le sous-titrage

- **Détection d'Actions** : Le flux optique est un indicateur direct de l'activité. Un fort flux optique dans une zone spécifique peut signaler une action rapide (par exemple, un coup de pied, un geste de la main).
- **Suivi d'Objets** : Il aide à suivre les objets et les personnes à travers les frames, ce qui est essentiel pour comprendre leurs trajectoires et leurs interactions continues.
- **Complément aux Caractéristiques Visuelles** : Il fournit des informations sur le *comment* les objets se déplacent, complétant les informations sur le *quoi* (les objets eux-mêmes) fournies par les caractéristiques visuelles.

### Réseaux de Neurones Convolutifs 3D (3D CNNs)

**Principe** Alors que les CNN 2D (vus précédemment) sont conçus pour traiter les images planes (2D), les CNN 3D étendent le concept de convolution pour opérer sur des volumes spatio-temporels. Au lieu d'utiliser des filtres 2D, les CNN 3D utilisent des filtres 3D qui se déplacent non seulement sur la largeur et la hauteur des images, mais aussi sur la dimension temporelle (à travers plusieurs frames consécutives). Cela leur permet de capturer simultanément les caractéristiques spatiales (apparence) et les caractéristiques temporelles (mouvement).

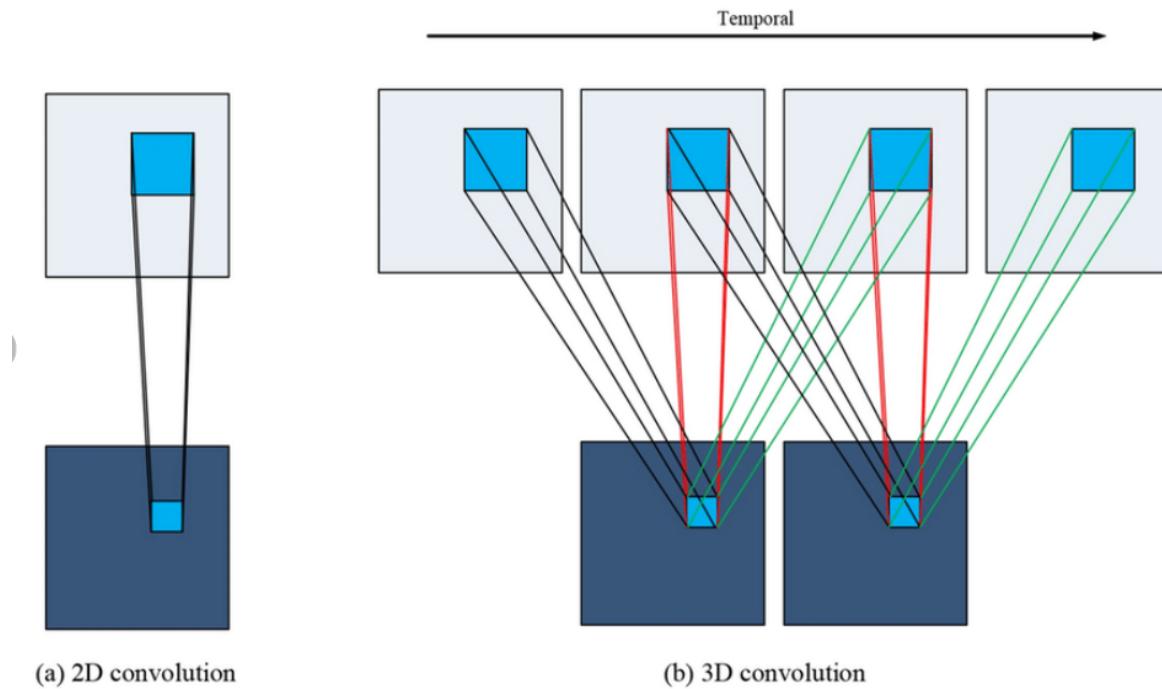


FIGURE 2.6 – Comparaison entre la convolution 2D et la convolution 3D. (a) La convolution 2D opère sur les dimensions spatiales d'une seule image (hauteur et largeur). (b) La convolution 3D capture les relations spatio-temporelles entre plusieurs frames successives, ce qui est essentiel pour comprendre les séquences vidéo.

**Méthode** Un filtre 3D appliqué à un « cube » de pixels (un petit volume 3D formé de pixels sur plusieurs frames adjacentes) peut apprendre des motifs tels que :

- La détection d'une arête qui bouge
- La progression d'une texture

En empilant ces couches 3D, le réseau apprend des représentations de plus en plus abstraites des actions et des événements complexes.

### Avantages Majeurs pour le sous-titrage

- **Apprentissage End-to-End** : Les 3D CNNs peuvent apprendre directement des représentations spatio-temporelles des pixels bruts de la vidéo, sans nécessiter d'extraction de flux optique séparée.
- **Intégration Naturelle du Mouvement** : Ils intègrent naturellement l'information de mouvement dès les premières couches, ce qui est plus difficile à réaliser avec des CNN 2D suivis d'un RNN.
- **Compréhension d'Actions Complexes** : Ils sont particulièrement efficaces pour reconnaître des actions complexes et des interactions qui se déroulent sur plusieurs frames, car ils modélisent directement la dynamique spatio-temporelle.

**Utilisation en Video Captioning** Les caractéristiques extraites par les couches profondes d'un 3D CNN (pré-entraîné sur de larges jeux de données d'actions vidéo comme Kinetics) peuvent servir de représentation riche de l'information vidéo, directement utilisable par le décodeur textuel.

## 2.2 Défis de l'Analyse Vidéo

Cette section explore les principaux défis inhérents à l'analyse et la compréhension automatique des contenus vidéo.

### La Grande Dimensionnalité des Données Vidéo : Le Fardeau de l'Abondance

Le premier choc pour tout système d'analyse vidéo est l'immense volume de données brutes qu'une vidéo représente. Une seule image numérique est déjà une matrice de pixels (largeur  $\times$  hauteur  $\times$  canaux de couleur). Une vidéo ajoute à cela une dimension supplémentaire : le temps.

TABLE 2.1 – Calcul de la dimensionnalité d'une vidéo HD (1080p, 30 fps)

Paramètre	Valeur
Résolution par frame	$1920 \times 1080$ pixels
Canaux de couleur	3 (RGB)
Fréquence d'images	30 fps
Données par frame (non compressé)	$\approx 6$ Mo
Données par minute (non compressé)	$\approx 10.8$ Go

Cette explosion de données pose plusieurs problèmes majeurs :

- **Charge computationnelle** : Le traitement de telles quantités de données exige une puissance de calcul colossale, nécessitant souvent des unités de traitement graphique (GPU) haut de gamme avec des mémoires de 16Go ou plus pour l'entraînement de modèles profonds.
- **Malédiction de la dimensionnalité (Curse of Dimensionality)** : Dans un espace de grande dimension tel que  $\mathbb{R}^{1920 \times 1080 \times 3 \times T}$  (où  $T$  est le nombre de frames), les données deviennent extrêmement éparses. Cela rend difficile pour les algorithmes de trouver des motifs significatifs et augmente le risque de sur-apprentissage (overfitting) sans une quantité astronomique de données d'entraînement pour "remplir" cet espace.
- **Coût de stockage** : La gestion et le stockage de ces volumes de données brutes impliquent des coûts significatifs et nécessitent l'emploi de solutions de compression sophistiquées (par exemple, H.265/HEVC) pour les rendre gérables.

### La Variabilité Intra-Classe et Inter-Classe : Le Spectre de l'Identique et du Distinct

Les données vidéo sont caractérisées par une variabilité stupéfiante, même pour des concepts apparemment identiques.

**Variabilité Intra-Classe** Elle désigne la diversité des manifestations d'une *même* catégorie d'objet ou d'action.

- **Objets** : Un "chat" peut se présenter sous d'innombrables apparences (races, couleurs, poses variées, angles de vue différents, occlusions partielles, variations d'éclairage et d'arrière-plan). Pour une machine, distinguer que toutes ces variations représentent toujours un "chat" est une tâche monumentale, exigeant du système qu'il généralise au-delà des différences superficielles.
- **Actions** : L'action de "marcher" peut varier considérablement selon la personne, sa vitesse, sa démarche, ses vêtements ou l'environnement. Un système doit capturer l'essence de l'action malgré ces différences apparentes.

- **Artéfacts** : La qualité des vidéos peut être compromise par des artéfacts de compression (par exemple, JPEG), du bruit de capteur ou une faible luminosité, ajoutant des "parasites" qui compliquent l'extraction des caractéristiques significatives.

**Variabilité Inter-Classe** Elle concerne la subtilité dans la distinction entre des catégories ou actions *differentes* mais superficiellement similaires.

- **Actions similaires** : Des actions comme "courir" et "marcher rapidement" peuvent présenter des mouvements très proches, exigeant une analyse très fine des nuances pour les distinguer. De même, différencier "parler" de "murmurer" ou "saisir" de "pousser" demande une grande précision.
- **Contextes culturels** : Un même geste (par exemple, un signe de la main) peut avoir des significations diamétralement opposées selon le contexte culturel, rendant l'interprétation universelle complexe et sujette à erreurs sans une connaissance contextuelle approfondie.
- **Occlusions** : La présence d'objets ou d'éléments qui masquent partiellement ou totalement des entités ou des actions clés représente un défi majeur, car le système doit inférer l'information manquante.

### Gestion de la Temporalité et des Dépendances à Long Terme : Le Défi de la Mémoire et de la Causalité

La nature séquentielle de la vidéo est à la fois sa force et son plus grand défi. Comprendre une vidéo, c'est comprendre une histoire qui se déroule dans le temps, nécessitant de capturer les relations dynamiques et la causalité des événements. Pour modéliser l'importance des informations à différents moments, des mécanismes d'attention peuvent être employés, conceptualisés par :

$$\mathcal{L}(t) = \sum_{\tau=0}^t \alpha_\tau \cdot f(x_\tau) \quad \text{où} \quad \alpha_\tau = \frac{e^{w_\tau}}{\sum_{i=0}^t e^{w_i}} \quad (2.1)$$

où  $\mathcal{L}(t)$  représente l'information temporelle pondérée agrégée jusqu'au temps  $t$ ,  $f(x_\tau)$  les caractéristiques extraites au temps  $\tau$ , et  $\alpha_\tau$  les poids d'attention calculés à partir des scores  $w_\tau$ .

Les défis spécifiques liés à la temporalité incluent :

- **Vanishing Gradient (Disparition du Gradient)** : Les architectures de réseaux neuronaux récurrents traditionnelles peuvent souffrir de ce problème, où les informations des premiers pas de temps sont progressivement "oubliées" à mesure que la séquence s'allonge (typiquement après quelques dizaines ou centaines de frames), rendant l'apprentissage des dépendances à long terme extrêmement difficile.
- **Attention Temporelle** : Le modèle doit être capable d'identifier et de se concentrer dynamiquement sur les moments clés (actions significatives, changements d'état, interactions cruciales) de la vidéo qui sont pertinents pour la description en cours de génération, plutôt que de traiter toutes les frames avec la même importance.
- **Actions Hiérarchiques et Compositionnelles** : Reconnaître des actions complexes qui se composent de sous-actions ordonnées (par exemple, savoir que "servir au tennis" implique séquentiellement "lancer la balle", puis "frapper la balle", et non l'inverse). Cela requiert une compréhension et une modélisation hiérarchique du temps.

**Solutions Émergentes en Deep Learning** Pour relever ces défis temporels et sémantiques, des architectures avancées ont été développées :

- **Transformers vidéo** : Basés sur des mécanismes d'attention multi-têtes, les Transformers sont intrinsèquement conçus pour capturer des relations à longue portée et des interactions complexes entre tous les éléments d'une séquence, ce qui les rend particulièrement efficaces pour la modélisation des dépendances temporelles dans les vidéos.
- **Mémoires externes** : L'intégration de modules de mémoire différentiables permet aux modèles de stocker et de récupérer activement des informations pertinentes sur de très longues séquences, simulant une forme de "mémoire à long terme" explicite.
- **Architectures hiérarchiques** : Des modèles qui traitent la vidéo à plusieurs échelles temporelles (par exemple, analyse des frames individuelles, puis des segments courts, puis des segments longs) peuvent mieux appréhender la structure hiérarchique des événements et la progression narrative.

## Synthèse des Défis

TABLE 2.2 – Comparatif des défis majeurs de l'analyse vidéo et des solutions potentielles en Deep Learning.

Défi	Impact Principal	Solutions Générales en Deep Learning
Dimensionnalité	Coût computationnel, éparseté des données	Modèles légers (ex : MobileNetV3), compression de caractéristiques
Variabilité	Manque de robustesse, difficulté de généralisation	Augmentation de données radicale, apprentissage de représentations robustes
Temporalité	Perte d'information, manque de cohérence	LSTMs/GRUs bidirectionnels, Transformers, attention temporelle

# CHAPITRE 3

## INTRODUCTION AU DEEP LEARNING POUR LE TRAITEMENT SÉQUENTIEL ET MULTIMODAL

### 3.1 Principes Fondamentaux du Deep Learning

### 3.2 Introduction au Deep Learning pour le Traitement Séquentiel et Multimodal

Le **Deep Learning** (apprentissage profond) est une sous-catégorie puissante de l'apprentissage automatique (Machine Learning) qui se distingue par l'utilisation de **réseaux de neurones artificiels profonds**. S'inspirant de la structure et du fonctionnement du cerveau humain, le Deep Learning permet aux systèmes d'apprendre des représentations complexes directement à partir de données brutes, plutôt que de s'appuyer sur une programmation explicite pour la reconnaissance de motifs. Son efficacité et sa capacité à traiter des informations multimodales et séquentielles reposent sur trois piliers fondamentaux : les architectures de réseaux neuronaux, l'apprentissage par descente de gradient, et la capacité à découvrir des représentations hiérarchiques.

#### 3.2.1 Principes Fondamentaux du Deep Learning

##### Réseaux de Neurones : Modélisation Mathématique de l'Intelligence Artificielle

Au cœur du Deep Learning se trouvent les **réseaux de neurones artificiels (RNA)**, qui sont des modèles computationnels inspirés par la structure du cerveau biologique. Un réseau de neurones peut être conceptualisé comme une **fonction paramétrée**  $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ , où  $\mathcal{X}$  est l'espace des entrées,  $\mathcal{Y}$  l'espace des sorties, et  $\theta$  représente l'ensemble de tous les **paramètres apprenables** (poids et biais) du réseau. Ces réseaux sont organisés en couches interconnectées de "neurones" (ou unités de calcul) qui transforment l'information de manière séquentielle.

**Modèle d'un Neurone Artificiel** L'unité de base d'un réseau de neurones est le **neurone artificiel**. Pour un neurone  $i$  situé dans une couche  $l$  du réseau, son fonctionnement peut être décrit en deux étapes :

1. **Sommation pondérée** : Le neurone reçoit des entrées de neurones de la couche précédente ( $l - 1$ ). Chaque entrée  $a_j^{(l-1)}$  est multipliée par un **poids synaptique**  $w_{ij}^{(l)}$ , qui représente la force de la connexion entre le neurone  $j$  de la couche  $l - 1$  et le neurone  $i$  de

la couche  $l$ . Tous ces produits pondérés sont sommés, et un **biais**  $b_i^{(l)}$  est ajouté au total. Le résultat de cette somme pondérée est appelé l'entrée nette ou **potentiel d'activation**  $z_i^{(l)}$  du neurone :

$$z_i^{(l)} = \sum_{j=1}^n w_{ij}^{(l)} a_j^{(l-1)} + b_i^{(l)} \quad (3.1)$$

où  $n$  est le nombre de neurones dans la couche précédente ( $l - 1$ ).

2. **Activation** : Le potentiel d'activation  $z_i^{(l)}$  est ensuite transformé par une **fonction d'activation non linéaire**  $\sigma$ . Cette fonction introduit la non-linéarité essentielle qui permet au réseau de modéliser des relations complexes dans les données, au-delà de simples transformations linéaires. Le résultat  $a_i^{(l)}$  est l'activation ou la sortie du neurone  $i$  :

$$a_i^{(l)} = \sigma(z_i^{(l)}) \quad (3.2)$$

**Architecture des Réseaux** Les neurones sont arrangés en couches :

- **Couche d'entrée** : C'est la première couche qui reçoit les données brutes du problème (par exemple, les pixels d'une image, les échantillons d'un signal audio, ou des embeddings de mots).
- **Couches cachées** : Ces couches intermédiaires sont le cœur du processus d'apprentissage et de transformation des données. Dans le Deep Learning, leur nombre est élevé (d'où le terme "profond"), permettant au réseau de construire des représentations de plus en plus abstraites et complexes.
- **Couche de sortie** : La dernière couche du réseau qui produit le résultat final, dont la structure dépend de la tâche (par exemple, une probabilité pour la classification, une valeur numérique pour la régression, ou une séquence de symboles pour la génération de texte).

**Fonctions d'Activation Courantes** Le choix de la fonction d'activation  $\sigma$  est crucial car elle introduit la non-linéarité, permettant au réseau de capturer des motifs complexes. La fonction

TABLE 3.1 – Fonctions d'activation non linéaires courantes et leurs propriétés

Fonction	Formule	Dérivée
Sigmoïde	$\sigma(z) = \frac{1}{1+e^{-z}}$	$\sigma(z)(1-\sigma(z))$
ReLU (Rectified Linear Unit)	$\text{ReLU}(z) = \max(0, z)$	$\mathbb{I}(z > 0)$ (où $\mathbb{I}$ est la fonction indicatrice)
Tanh (Tangente Hyperbolique)	$\tanh(z)$	$1 - \tanh^2(z)$

ReLU est devenue un standard pour les couches cachées grâce à sa simplicité et sa capacité à atténuer le problème du gradient évanescents.

### Apprentissage par Descente de Gradient : Le Moteur de l'Optimisation

L'apprentissage dans un réseau de neurones consiste à ajuster les paramètres (poids  $w$  et biais  $b$ ) afin que le réseau puisse effectuer la tâche désirée avec la plus grande précision. Ceci est réalisé en minimisant une **fonction de perte** (ou **fonction coût**)  $\mathcal{L}(\theta)$ , qui quantifie l'erreur entre les prédictions du réseau et les valeurs réelles (vérité terrain). L'objectif est de trouver l'ensemble optimal de paramètres  $\theta^*$  qui minimise cette fonction de perte :

$$\theta^* =_{\theta} \mathcal{L}(\theta) \quad (3.3)$$

La méthode la plus répandue pour cette minimisation est la **descente de gradient**.

**Processus de la Descente de Gradient** Le processus d'apprentissage itératif se déroule comme suit :

1. **Propagation avant (Forward Pass)** : Une donnée d'entrée est fournie au réseau, et l'information traverse les couches, de l'entrée à la sortie, pour produire une prédiction.
2. **Calcul de la Perte** : La fonction de perte  $\mathcal{L}$  évalue la différence entre la prédiction du réseau et la cible réelle.
3. **Rétropropagation (Backpropagation)** : C'est l'algorithme clé qui calcule les gradients de la fonction de perte par rapport à chaque paramètre du réseau. Il utilise la règle de la chaîne du calcul différentiel pour propager l'erreur de la couche de sortie vers les couches d'entrée. Pour un poids  $w_{ij}^{(l)}$  connectant le neurone  $j$  de la couche  $l - 1$  au neurone  $i$  de la couche  $l$ , le gradient est donné par :

$$\frac{\partial \mathcal{L}}{\partial w_{ij}^{(l)}} = \frac{\partial \mathcal{L}}{\partial a_i^{(l)}} \cdot \frac{\partial a_i^{(l)}}{\partial z_i^{(l)}} \cdot \frac{\partial z_i^{(l)}}{\partial w_{ij}^{(l)}} \quad (3.4)$$

Cette étape est fondamentale car elle indique dans quelle direction et avec quelle magnitude chaque paramètre doit être ajusté pour réduire l'erreur.

4. **Mise à Jour des Poids** : Les paramètres du réseau sont ajustés dans la direction opposée au gradient de la fonction de perte, de manière à se déplacer vers un minimum. Cette mise à jour est régulée par un hyperparamètre appelé le **taux d'apprentissage**  $\eta$ , qui contrôle la taille du pas à chaque itération :

$$w_{ij}^{(l)} \leftarrow w_{ij}^{(l)} - \eta \frac{\partial \mathcal{L}}{\partial w_{ij}^{(l)}} \quad (3.5)$$

Un taux d'apprentissage approprié est crucial pour la convergence de l'apprentissage.

Ce processus est répété sur de nombreux "époques" (passages complets sur le jeu de données d'entraînement) et "mini-lots" (petits sous-ensembles de données) jusqu'à ce que la fonction de perte converge vers un minimum, indiquant que le réseau a appris les motifs sous-jacents aux données.

## Représentations Hiérarchiques : L'Essence de la Profondeur

La caractéristique la plus distinctive et la plus puissante du Deep Learning est sa capacité à apprendre des **représentations hiérarchiques** des données. Grâce à ses multiples couches cachées, un réseau profond ne se contente pas d'une transformation unique des données d'entrée ; il construit progressivement des représentations de plus en plus abstraites et complexes. Chaque couche apprend à transformer les représentations de la couche précédente en de nouvelles représentations plus informatives.

$$\text{Représentation finale} = f_L(f_{L-1}(\dots f_1(\mathbf{x}) \dots)) \quad (3.6)$$

où  $\mathbf{x}$  est l'entrée brute, et chaque  $f_l$  est la fonction (potentiellement non linéaire) apprise par la  $l$ -ième couche du réseau.

**Exemple pour l'Analyse d'Images (et par extension, Vidéos)** Pour une image, ce processus hiérarchique peut être visualisé comme suit :

- **Niveau 1 (Couches d'entrée)** : Apprentissage de **caractéristiques de bas niveau** comme la détection de bords, de textures fines et de motifs de couleurs simples.

- **Niveau 2** : Combinaison de ces caractéristiques de bas niveau pour former des **motifs plus complexes** et des formes géométriques de base (par exemple, des coins, des courbes, des petites régions texturées).
- **Niveau 3** : Reconnaissance de **parties d'objets** ou de motifs intermédiaires (par exemple, un œil, une oreille, une roue, une partie de texte).
- **Niveau 4** : Assemblage de ces parties pour former des **objets complets** (par exemple, un visage, une voiture, un animal).
- **Niveau 5 (Couches plus profondes)** : Compréhension de **scènes complexes** et de leurs contextes (par exemple, un parc avec des personnes, une rue animée avec des véhicules).

### Avantages Clés des Représentations Hiérarchiques

- **Abstraction Croissante** : Chaque couche successive construit sur les représentations apprises par la précédente, permettant de capturer des concepts de plus en plus abstraits et sémantiquement riches à partir des données brutes.
- **Invariance et Robustesse** : Les couches profondes apprennent des représentations qui sont de plus en plus insensibles aux variations non pertinentes dans les données d'entrée (par exemple, position, échelle, orientation, éclairage, petites déformations). Cela rend le modèle plus robuste face à la variabilité inhérente aux données du monde réel.
- **Compositionnalité** : Les réseaux profonds peuvent apprendre à combiner de manière non linéaire des caractéristiques simples pour former des représentations de concepts complexes, ce qui est essentiel pour la compréhension de données comme la vidéo où les informations sont imbriquées et interdépendantes.
- **Découverte Automatique de Caractéristiques** : Contrairement aux approches traditionnelles où des experts devaient manuellement concevoir des extracteurs de caractéristiques (comme SIFT ou HOG), les réseaux profonds sont capables d'apprendre ces caractéristiques pertinentes de manière autonome, directement à partir des données d'entraînement.

**Formalisation Mathématique des Couches Profondes** Pour un réseau à  $L$  couches, la transformation d'une couche à l'autre peut être généralisée vectoriellement :

$$\mathbf{h}^{(l)} = \sigma(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}), \quad l = 1, \dots, L \quad (3.7)$$

où  $\mathbf{h}^{(0)} = \mathbf{x}$  est le vecteur d'entrée,  $\mathbf{W}^{(l)}$  est la matrice des poids de la  $l$ -ième couche,  $\mathbf{b}^{(l)}$  est le vecteur des biais de la  $l$ -ième couche,  $\sigma$  est la fonction d'activation appliquée élément par élément, et  $\mathbf{h}^{(L)}$  est le vecteur de sortie finale (ou les représentations profondes utilisées pour une tâche subséquente).

## Synthèse des Concepts Fondamentaux

TABLE 3.2 – Comparaison des concepts clés du Deep Learning

Concept	Description Mathématique Clé
Neurone	$a = \sigma(\mathbf{w}^T \mathbf{x} + b)$
Couche Linéaire	$\mathbf{z} = \mathbf{Wx} + \mathbf{b}$ (avant activation)
Couche Activée	$\mathbf{h} = \sigma(\mathbf{Wx} + \mathbf{b})$
Rétropropagation (gradient)	$\nabla_{\theta}\mathcal{L} = \frac{\partial \mathcal{L}}{\partial \mathbf{y}} \prod_{l=L}^1 \frac{\partial \mathbf{h}^{(l)}}{\partial \mathbf{h}^{(l-1)}}$ (Règle de la chaîne)
Mise à jour des poids	$\theta \leftarrow \theta - \eta \nabla_{\theta}\mathcal{L}$
Représentation (embeddings)	$\phi(\mathbf{x}) = \mathbf{h}^{(L-1)}$ (sortie de la dernière couche cachée)

## 3.3 Réseaux de Neurones Convolutifs (CNN)

Les **Réseaux de Neurones Convolutifs (CNN)** constituent une classe spécialisée de réseaux de neurones profonds, spécifiquement conçus pour traiter et analyser des données présentant une structure de grille topologique, dont les exemples les plus courants sont les images 2D (photographies) et, par extension, les séquences vidéo (qui peuvent être conceptualisées comme des données 3D spatio-temporelles, où le temps est la troisième dimension). Leur succès phénoménal dans des applications variées, allant de la reconnaissance faciale à la détection d'objets dans la conduite autonome, en passant par la segmentation d'images et la classification médicale, les rend absolument indispensables pour toute tâche nécessitant une compréhension profonde du contenu visuel, y compris le sous-titrage automatique de vidéos.

### 3.3.1 Le Rôle Stratégique des CNN dans l'Extraction de Caractéristiques Spatiales

Avant l'avènement des CNN, les tentatives d'appliquer des réseaux de neurones traditionnels (les Perceptrons Multicouches ou MLP) directement aux données d'images se heurtaient à des obstacles insurmontables, limitant sévèrement leurs capacités à extraire des caractéristiques pertinentes et à généraliser. Ces limitations incluaient :

- 1. L'Explosion Combinatoire des Paramètres** : Imaginez une image de taille relativement modeste, par exemple  $100 \times 100$  pixels, avec trois canaux de couleur (Rouge, Vert, Bleu - RGB). Cela représente  $100 \times 100 \times 3 = 30\,000$  valeurs d'entrée. Si la première couche cachée d'un MLP contient seulement 100 neurones, le nombre de connexions (et donc de poids apprenables) entre la couche d'entrée et cette première couche cachée s'élèverait à  $30\,000 \times 100 = 3$  millions. Pour des images de haute résolution (comme le Full HD ou le 4K), ce nombre devient astronomique et ingérable, rendant l'entraînement du réseau pratiquement impossible en raison des exigences de calcul et de mémoire, et le rendant extrêmement susceptible au sur-apprentissage (overfitting), car le modèle peut simplement "mémoriser" les données d'entraînement au lieu d'apprendre des motifs généralisables.
- 2. La Destruction de la Structure Spatiale Intrinsèque** : Pour qu'une image soit traitée par un MLP, elle doit être "aplatis" en un unique vecteur unidimensionnel (c'est-à-dire que tous les pixels sont alignés les uns après les autres). Cette opération efface toute l'information intrinsèque sur la proximité spatiale des pixels. Or, la sémantique visuelle est fondamentalement construite sur les relations locales : un bord est défini par

des pixels voisins, une texture par des motifs répétitifs à petite échelle, et un objet par la disposition relative de ses parties. Un MLP, qui traite chaque entrée indépendamment des autres (sauf via les poids globaux), ne peut pas exploiter cette richesse structurelle locale.

3. **Le Manque d'Invariance Intrinsèque aux Transformations Géométriques** : Les MLP étaient extrêmement sensibles aux petites variations géométriques des objets (légères translations, rotations, changements d'échelle ou déformations). Si un objet reconnaissable (par exemple, un visage) apparaissait à un endroit légèrement différent dans l'image, le réseau le percevait comme une entrée totalement nouvelle et distincte, exigeant des quantités gargantuesques de données d'entraînement pour apprendre toutes les variantes possibles de chaque objet, rendant l'apprentissage inefficace et impraticable.

Les CNN ont été spécifiquement développés pour surmonter ces défis en exploitant de manière inhérente et efficace la **structure spatiale locale** des données imagées. Leur conception architecturale est basée sur trois concepts fondamentaux : les **opérations de convolution**, les **opérations de pooling (ou sous-échantillonnage)**, et l'intégration de couches entièrement connectées pour la prise de décision finale.

---

### 3.3.2 Opérations Fondamentales Constituant un CNN

Un CNN typique est agencé comme une succession de couches, chacune appliquant des transformations spécifiques aux données qu'elle reçoit.

#### a) Les Couches de Convolution (Convolutional Layers)

La **couche de convolution** est la pierre angulaire de tout CNN et le mécanisme primaire d'extraction de caractéristiques. Elle permet de détecter des motifs locaux et de générer des "cartes de caractéristiques" qui mettent en évidence la présence de ces motifs.

**Le Filtre (Kernel ou Masque de Convolution)** C'est une petite matrice tridimensionnelle de poids numériques (par exemple,  $3 \times 3 \times$  nombre de canaux d'entrée,  $5 \times 5 \times$  nombre de canaux d'entrée). Chaque filtre est spécialisé pour réagir fortement à un type très spécifique de motif visuel : il peut s'agir d'un bord horizontal, d'un bord vertical, d'un coin, d'une texture donnée, ou d'un gradient de couleur particulier. Les valeurs numériques (poids) à l'intérieur de cette matrice sont les **paramètres apprenables** du réseau qui seront ajustés et optimisés pendant la phase d'entraînement via la rétropropagation.

**L'Opération de Convolution** L'opération consiste à faire "glisser" (ou parcourir) le filtre sur l'image d'entrée (ou la carte de caractéristiques de la couche précédente). Le filtre se déplace systématiquement sur la grille, généralement de gauche à droite, puis de haut en bas, selon un pas défini. À chaque position où le filtre est superposé à une région de l'entrée, il effectue un **produit élément par élément** entre ses propres poids et les valeurs des pixels (ou activations) de la région correspondante de l'entrée. La somme de ces produits est ensuite stockée comme une seule valeur dans une nouvelle matrice, appelée **carte de caractéristiques (feature map)** ou **carte d'activation**. Cette carte de caractéristiques représente l'intensité de la détection du motif du filtre à chaque position spatiale de l'entrée. Des valeurs élevées indiquent une forte correspondance avec le motif du filtre.

**Formalisation Mathématique de la Convolution Discrète** : Soit  $I$  l'image d'entrée (ou une carte de caractéristiques précédente) de dimension  $H \times W \times C_{in}$ , et  $K$  un filtre (kernel)

de dimension  $F_H \times F_W \times C_{in}$ . La sortie  $S$  (carte de caractéristiques) de dimension  $H' \times W' \times 1$  de l'opération de convolution à la position  $(i, j)$  est calculée comme :

$$S(i, j) = \sum_{c=0}^{C_{in}-1} \sum_{m=0}^{F_H-1} \sum_{n=0}^{F_W-1} I(i \cdot \text{stride}_H + m, j \cdot \text{stride}_W + n, c) \cdot K(m, n, c) \quad (3.8)$$

où les indices  $i, j$  parcourent les dimensions spatiales de la sortie, et  $\text{stride}_H, \text{stride}_W$  sont les pas (voir ci-dessous).

### Paramètres Clés de l'Opération de Convolution

- **Stride (Pas)** : Le **stride** détermine le nombre de pixels dont le filtre se décale à chaque mouvement (horizontalement et verticalement). Un stride de 1 pixel est courant et conserve la résolution spatiale relative. Un stride de 2 (ou plus) réduit la taille spatiale de la carte de caractéristiques de sortie, ce qui peut diminuer la complexité computationnelle et aider à capturer des caractéristiques à plus grande échelle.
- **Padding (Remplissage)** : Le **padding** consiste à ajouter des pixels (généralement de valeur zéro, appelé "zero-padding") autour des bords de l'image d'entrée avant d'appliquer la convolution. Les motivations sont doubles :
  1. **Contrôle de la Taille Spatiale** : Sans padding, la taille spatiale de la carte de caractéristiques diminue à chaque couche de convolution. Le padding permet de maintenir la taille spatiale de la sortie égale à celle de l'entrée ("same" padding) ou de la contrôler précisément.
  2. **Traitement des Pixels de Bord** : Les pixels situés sur les bords de l'image sont autrement "vus" moins souvent par le filtre que les pixels centraux. Le padding assure que tous les pixels de l'image d'entrée contribuent de manière équitable aux cartes de caractéristiques.
- **Nombre de Filtres** : Une couche de convolution n'utilise pas un seul filtre, mais généralement plusieurs (par exemple, 32, 64, 128, 256, ou plus). Chaque filtre est indépendant et apprend à détecter un motif distinct. Le nombre de filtres détermine donc la "profondeur" (le nombre de canaux) de la carte de caractéristiques de sortie, créant ainsi une représentation multi-dimensionnelle des motifs détectés.

### Propriétés Fondamentales et Avantages des Couches de Convolution

- **Connectivité Locale** : Chaque neurone dans la carte de caractéristiques de sortie n'est connecté qu'à une petite région locale (appelée le **champ réceptif**) de la couche d'entrée. Cela est biologiquement inspiré et réduit considérablement le nombre de paramètres par rapport aux réseaux entièrement connectés, car un neurone n'a pas besoin de "voir" toute l'image.
- **Partage de Paramètres (Poids Partagés)** : Le même filtre (la même matrice de poids) est appliqué à toutes les positions spatiales de l'image d'entrée. Cela signifie que les poids utilisés pour détecter un motif spécifique sont partagés à travers toute l'image. Ce partage de paramètres réduit drastiquement le nombre total de paramètres du modèle (en le rendant indépendant de la taille de l'image d'entrée) et confère aux CNNs une propriété cruciale : l'**invariance à la translation**. Si un motif (par exemple, un œil) apparaît à un endroit différent dans l'image, il sera toujours détecté par le même filtre, sans qu'il soit nécessaire de l'apprendre spécifiquement pour chaque position.
- **Hiérarchie de Détection** : Les premières couches de convolution apprennent à détecter des caractéristiques de bas niveau (bords simples, coins, textures primitives). Les couches plus profondes combinent ces caractéristiques simples pour détecter des motifs plus complexes et abstraits.

## b) Fonctions d'Activation Non Linéaires

Immédiatement après chaque opération de convolution (et parfois de pooling), une **fonction d'activation non linéaire** (telle que ReLU, Sigmoïde, ou Tanh) est appliquée élément par élément aux valeurs de la carte de caractéristiques. Cette non-linéarité est absolument essentielle, car elle permet au réseau de modéliser des relations complexes dans les données qui ne sont pas linéaires. Sans elle, un réseau de neurones profond ne serait qu'une série de transformations linéaires équivalentes à une seule transformation linéaire, limitant considérablement sa capacité expressive. La fonction **ReLU (Rectified Linear Unit)** est la plus répandue pour les couches cachées en raison de son efficacité computationnelle (elle ne fait que tronquer les valeurs négatives à zéro) et de sa capacité à atténuer le problème du vanishing gradient.

## c) Couches de Pooling (Pooling Layers ou Sous-échantillonnage)

Les **couches de pooling (ou sous-échantillonnage)** sont utilisées pour réduire la dimensionnalité spatiale (largeur et hauteur) des cartes de caractéristiques. Elles jouent un rôle crucial dans la réduction de la complexité computationnelle et l'amélioration de la robustesse du modèle.

### Avantages du Pooling

- **Réduction de la Dimensionnalité** : Diminue significativement le nombre de paramètres et la charge de calcul pour les couches suivantes du réseau, rendant l'entraînement plus rapide et le modèle moins sujet au sur-apprentissage.
- **Invariance à la Translation Locale** : En sélectionnant la caractéristique la plus dominante (ou une moyenne) dans une petite région, le pooling rend le modèle plus tolérant aux petites variations de position des motifs détectés. Même si un motif est légèrement décalé dans l'image d'entrée, il produira une activation similaire après l'opération de pooling, car seul le maximum (ou la moyenne) de la région est retenu.
- **Augmentation du Champ Réceptif** : En réduisant la taille des cartes de caractéristiques, le pooling permet aux couches supérieures du réseau d'avoir un champ réceptif effectif plus grand sur l'image d'entrée originale, les aidant à capturer des informations contextuelles plus larges.

### Mécanismes de Pooling Courants

- **Max Pooling** : C'est l'opération de pooling la plus fréquemment utilisée. Pour une fenêtre donnée (par exemple,  $2 \times 2$  pixels), elle sélectionne la valeur maximale de cette fenêtre et la place dans la carte de caractéristiques de sortie. Cela permet de conserver les informations les plus importantes (les activations les plus fortes) et les caractéristiques les plus discriminantes.
- **Average Pooling** : Calcule la moyenne des valeurs dans la fenêtre. Moins courant que le max pooling pour l'extraction de caractéristiques intermédiaires, mais parfois utilisé dans les dernières étapes du réseau (par exemple, Global Average Pooling) pour réduire les dimensions avant les couches entièrement connectées.

## d) Couches Entièrement Connectées (Fully Connected Layers ou Couches Denses)

Après plusieurs blocs de couches de convolution et de pooling (qui sont les couches d'extraction de caractéristiques), les cartes de caractéristiques de haut niveau générées sont généralement "aplatis" (transformées en un unique vecteur unidimensionnel). Ce vecteur est ensuite alimenté à une ou plusieurs **couches entièrement connectées (ou denses)**. Ces couches

fonctionnent comme les couches d'un réseau de neurones traditionnel, où chaque neurone d'une couche est connecté à *tous* les neurones de la couche précédente. Elles sont responsables de la phase de **raisonnement de haut niveau** et de la classification ou de la régression finale, utilisant les caractéristiques abstraites et discriminantes apprises par les couches convolutives pour faire des prédictions concrètes (par exemple, déterminer la classe d'un objet, ou générer des probabilités pour des mots dans un sous-titre).

---

### 3.3.3 L'Essence "Profonde" des CNN : L'Apprentissage Hiérarchique de Représentations Visuelles

La véritable puissance et l'avantage concurrentiel des CNN résident dans leur capacité intrinsèque à construire et à apprendre une **hiérarchie de caractéristiques** de manière automatique et adaptative. Ceci est le fondement même de la "profondeur" du Deep Learning en vision.

- Les **premières couches** convolutives (les plus proches de l'entrée pixel brut) sont chargées d'apprendre et de détecter des motifs visuels très simples, génériques et locaux. Il s'agit typiquement de la détection de bords dans différentes orientations (horizontaux, verticaux, diagonaux), de coins, de taches de couleurs spécifiques, ou de textures primitives. Ces caractéristiques sont considérées comme "universelles" car elles sont présentes dans presque toutes les images, quel que soit leur contenu sémantique.
- Les **couches intermédiaires** du réseau combinent les caractéristiques de bas niveau apprises par les couches précédentes pour former des motifs plus complexes et sémantiquement plus significatifs. Par exemple, des bords et des coins peuvent être combinés pour former des formes géométriques simples, des textures spécifiques (comme la fourrure d'un animal, le grain du bois), ou des parties reconnaissables d'objets (comme un œil, une roue, une aile d'oiseau).
- Les **dernières couches** (les plus profondes du réseau) synthétisent ces caractéristiques intermédiaires pour reconnaître des concepts de très haut niveau, directement pertinents pour la tâche finale. Il peut s'agir d'objets complets (un chat, une voiture, un arbre), de la reconnaissance de l'identité de personnes, de la compréhension de scènes entières (par exemple, "un parc ensoleillé avec des enfants qui jouent"), ou même d'actions complexes se déroulant dans une vidéo (par exemple, "une personne qui saute par-dessus un obstacle").

Cette capacité à apprendre des représentations de plus en plus abstraites, discriminantes et sémantiquement riches à travers des couches successives est ce qui permet aux CNN de "comprendre" des images visuelles complexes avec une précision et une robustesse remarquables. Crucialement, cette extraction de caractéristiques est **automatique**; le réseau apprend les meilleures caractéristiques directement à partir des données d'entraînement, éliminant le besoin pour des experts humains de concevoir manuellement des extracteurs de caractéristiques (une tâche souvent appelée "feature engineering").

---

### 3.3.4 Architectures Populaires de CNN et leur Utilisation comme Extracteurs de Caractéristiques (Transfert d'Apprentissage)

L'innovation architecturale a été un moteur essentiel des progrès des CNN. Cependant, la complexité et les besoins en données pour entraîner un CNN de pointe à partir de zéro sont immenses. C'est pourquoi la technique du **transfert d'apprentissage** est devenue une pratique standard et hautement efficace en Deep Learning appliquée.

Le principe du transfert d'apprentissage repose sur l'idée que les modèles CNN qui ont été **pré-entraînés sur des jeux de données très vastes et diversifiés** (comme ImageNet, qui contient des millions d'images étiquetées dans 1000 catégories d'objets différentes) ont déjà appris à extraire des caractéristiques visuelles génériques de très haute qualité. Ces caractéristiques sont considérées comme "universelles" car elles sont utiles pour une grande variété de tâches de vision, même celles qui sont différentes de la tâche originale d'entraînement (classification d'ImageNet).

Pour utiliser un CNN pré-entraîné comme **extracteur de caractéristiques** :

1. Le modèle CNN pré-entraîné est chargé.
2. Généralement, la dernière couche (la couche de classification, qui est spécifique aux 1000 classes d'ImageNet ou à une tâche particulière) est **retirée**.
3. L'image d'entrée (ou une frame vidéo) est passée à travers toutes les couches restantes du réseau.
4. La sortie d'une des dernières couches cachées (par exemple, la dernière couche convulsive ou une couche entièrement connectée juste avant celle qui a été retirée) est capturée. Ce résultat est le **vecteur de caractéristiques (feature vector)** ou l'**embedding** de l'image. Ce vecteur est une représentation dense, compacte, et sémantiquement riche de l'image originale, capturant l'essence visuelle de manière bien plus significative que les simples valeurs de pixels.

Ces vecteurs de caractéristiques peuvent ensuite être utilisés comme entrées pour des modèles subséquents (par exemple, des réseaux récurrents pour le sous-titrage vidéo, des classificateurs plus simples, ou d'autres modules de traitement).

Voici quelques-unes des architectures CNN les plus populaires et les plus influentes, largement utilisées pour l'extraction de caractéristiques :

## VGG (Visual Geometry Group)

- **Année de Publication** : 2014
- **Innovation Clé** : VGG, développé par l'Université d'Oxford, a mis en évidence l'importance capitale de la **profondeur uniforme** dans les architectures de CNN. Au lieu d'utiliser des filtres de tailles variées, les réseaux VGG (notamment VGG16 et VGG19, comportant respectivement 16 et 19 couches pondérées) sont caractérisés par l'utilisation répétée et empilée de **petits filtres convolutifs de taille  $3 \times 3$**  (avec un stride de 1 et un padding de 1) tout au long du réseau, alternant avec des couches de Max Pooling pour la réduction de la dimensionnalité. L'idée était qu'une séquence de  $3 \times 3$  convolutions pouvait simuler l'effet d'un filtre plus grand (par exemple, deux  $3 \times 3$  donnent un champ réceptif de  $5 \times 5$ , et trois  $3 \times 3$  donnent un  $7 \times 7$ ), mais avec plus de non-linéarités et moins de paramètres.
- **Importance et Rôle comme Extracteur** : VGG a démontré de manière empirique et convaincante que l'augmentation de la profondeur était un facteur clé pour améliorer significativement la précision des tâches de classification sur de grands ensembles de données. Bien que plus coûteux en calcul et en mémoire que des architectures plus récentes, les modèles VGG pré-entraînés sont encore largement utilisés comme **extracteurs de caractéristiques robustes et polyvalents** en raison de leur capacité à produire des représentations visuelles de haute qualité et de leur architecture relativement simple et intuitive.

## ResNet (Residual Network - Réseau Résiduel)

- **Année de Publication :** 2015
- **Innovation Clé :** Développé par Microsoft Research, ResNet a résolu un problème majeur des réseaux très profonds : le **vanishing gradient (disparition du gradient)** et la dégradation des performances. L'innovation architecturale majeure est l'introduction des **connexions résiduelles (Residual Connections)** ou "skip connections". Ces connexions permettent aux informations (et aux gradients) de "sauter" des blocs de couches convolutives (appelés "blocs résiduels"), en ajoutant directement l'entrée d'un bloc à sa sortie. Cela permet au réseau d'apprendre des fonctions résiduelles (la différence entre la sortie idéale et l'entrée), ce qui est plus facile que d'apprendre la fonction complète.
- **Impact et Rôle comme Extracteur :** Cette innovation a permis d'entraîner des réseaux avec une profondeur sans précédent (ResNet-50, ResNet-101, ResNet-152, et même des versions avec plus de 1000 couches), ce qui a conduit à des améliorations significatives des performances et à des architectures plus stables. Grâce à leur capacité à apprendre des représentations très profondes, robustes et hautement discriminantes, les modèles ResNet pré-entraînés sont devenus l'un des **extracteurs de caractéristiques par excellence** en Deep Learning, omniprésents dans presque toutes les applications nécessitant une compréhension visuelle de pointe.

## Inception (GoogleNet)

- **Année de Publication :** 2014
- **Innovation Clé :** Le réseau Inception (souvent associé à GoogleNet, développé par Google) a introduit le concept du **module Inception**. Ce module permet au réseau d'effectuer des opérations de convolution avec **différentes tailles de filtre (par exemple,  $1 \times 1$ ,  $3 \times 3$ ,  $5 \times 5$ ) et des opérations de pooling, en parallèle** sur la même entrée. Les sorties de ces chemins parallèles sont ensuite concaténées pour former la carte de caractéristiques de la couche suivante. L'utilisation stratégique de convolutions  $1 \times 1$  est également clé pour la réduction de la dimensionnalité (bottleneck layer) avant les filtres plus grands, ce qui optimise l'efficacité computationnelle.
- **Impact et Rôle comme Extracteur :** Les modules Inception ont permis de construire des réseaux qui utilisent les ressources computationnelles plus efficacement, d'élargir le réseau et de capturer des informations à **différentes échelles spatiales simultanément** au sein d'une même couche. Les caractéristiques générées par les réseaux Inception (comme InceptionV3) sont reconnues pour leur richesse et leur capacité à capturer des détails multi-échelles, les rendant très efficaces comme extracteurs de caractéristiques dans des contextes variés, où la détection de motifs à diverses échelles est cruciale.

## Autres Architectures Influentes et Types d'Extracteurs de Caractéristiques CNN

Bien qu'il soit impossible de lister *tous* les algorithmes et architectures de CNN existants en raison de l'évolution rapide du domaine, voici d'autres familles d'architectures qui ont eu un impact significatif et sont fréquemment utilisées comme extracteurs de caractéristiques :

- **AlexNet (2012)** : Historiquement important pour avoir remporté le concours ImageNet en 2012, marquant le début de l'ère moderne du Deep Learning en vision. Bien que moins performant que VGG ou ResNet pour le transfert d'apprentissage moderne, il a prouvé la viabilité des CNN avec les GPU.

- **DenseNet (Densely Connected Convolutional Networks, 2017)** : Pousse l'idée des connexions résiduelles plus loin. Chaque couche reçoit les entrées de *toutes* les couches précédentes et passe sa propre carte de caractéristiques (via concaténation) à *toutes* les couches suivantes. Cela favorise une réutilisation massive des caractéristiques, une réduction du problème du vanishing gradient, et des modèles souvent plus compacts et efficaces en termes de paramètres.
- **EfficientNet (2019)** : Une famille de modèles conçue en utilisant une technique de mise à l'échelle composée (compound scaling) pour ajuster uniformément la profondeur, la largeur et la résolution d'un CNN. EfficientNet a démontré une performance de pointe avec une efficacité de paramètres et de calcul bien supérieure à celle de ses prédecesseurs, les rendant très populaires pour l'extraction de caractéristiques dans les applications à ressources limitées.
- **MobileNet (V1, V2, V3 - 2017-2019)** : Une famille d'architectures optimisées pour les appareils mobiles et les environnements à ressources limitées. Elles atteignent cette efficacité grâce à l'utilisation de **convolutions séparables en profondeur (depthwise separable convolutions)**, qui décomposent une convolution standard en deux étapes plus petites. Les MobileNets sont d'excellents extracteurs de caractéristiques lorsque l'efficacité computationnelle est primordiale.
- **RegNet (2020)** : Proposé par Facebook AI, RegNet est une famille de modèles conçue en utilisant une approche de recherche d'architecture par la régularisation de l'espace de conception, plutôt que la recherche heuristique ou par blocs manuels. Ils offrent une bonne performance et sont une preuve de l'évolution des méthodologies de conception des CNN.
- **Vision Transformers (ViT) / Swin Transformers (post-2020)** : Bien qu'ils ne soient pas des CNN au sens strict (ils utilisent des mécanismes d'auto-attention au lieu de convolutions), il est crucial de les mentionner car ils ont montré des performances supérieures aux CNN sur de nombreuses tâches de vision. Pour les tâches comme le sous-titrage vidéo, ils peuvent être utilisés comme des extracteurs de caractéristiques visuelles (en traitant les images comme des séquences de patchs) ou directement pour la modélisation spatio-temporelle. Leur émergence est un développement majeur qui supplante ou combine les CNN dans certains pipelines de vision.

En synthèse, pour votre projet de sous-titrage vidéo, les CNNs pré-entraînés sont des outils fondamentaux. Ils agissent comme la "couche de perception" visuelle, extrayant des vecteurs de caractéristiques spatiales hautement sémantiques de chaque frame vidéo. Ces représentations visuelles sont ensuite combinées avec des modèles de séquençage (comme les RNN/LSTMs ou les Transformers) pour capturer les dépendances temporelles et générer la description textuelle finale, formant ainsi un système de sous-titrage multimodal.

### 3.4 Réseaux de Neurones Récurrents (RNN)

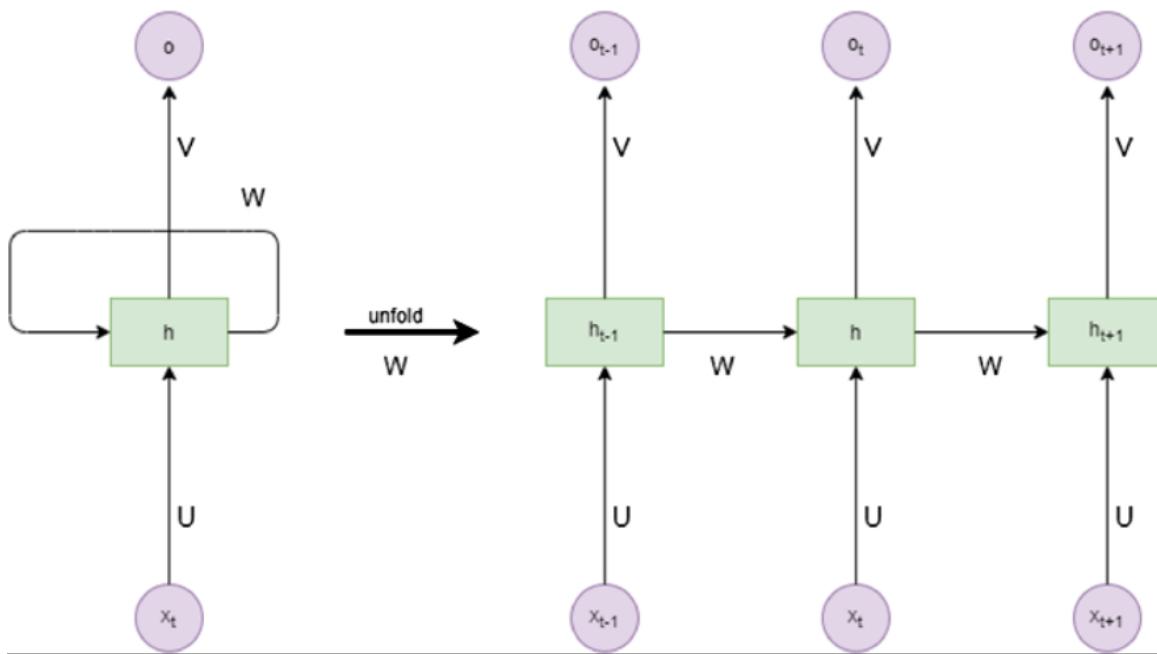


FIGURE 3.1 – Architecture RNN (Récurrent Neural Network) dépliée dans le temps, montrant la connexion séquentielle entre les états cachés ( $h_{t-1}$ ,  $h_t$ ,  $h_{t+1}$ ) et les poids partagés ( $W$ ,  $U$ ) à chaque pas de temps.

Les **Réseaux de Neurones Récurrents (RNN)** sont une classe fondamentale et distincte de réseaux de neurones profonds, spécifiquement conçus pour la modélisation et le traitement des **données séquentielles**. Contrairement aux architectures feed-forward (comme les Perceptrons Multicouches ou les CNN) où les informations ne circulent que dans une seule direction (de l'entrée vers la sortie, sans boucles), les RNN possèdent des **connexions récurrentes** ou des **boucles de rétroaction**. Ces boucles permettent aux informations de persister dans le réseau à travers le temps, conférant aux RNN une forme de "mémoire" interne. Cette capacité est cruciale pour des tâches où l'ordre des éléments et les dépendances passées sont essentiels, telles que le traitement du langage naturel (NLP – par exemple, traduction automatique, génération de texte), la reconnaissance vocale, l'analyse de séries temporelles (prévisions météorologiques, financières), et, de manière très pertinente pour votre projet, la **modélisation de la dynamique temporelle des caractéristiques visuelles** pour le sous-titrage vidéo.

L'idée centrale derrière les RNN est qu'ils appliquent la même opération sur chaque élément d'une séquence, mais les résultats de cette opération sont influencés par les calculs précédents. À chaque pas de temps  $t$ , un RNN prend en entrée l'élément courant de la séquence  $x_t$  et combine cette information avec un **état caché (hidden state)** précédent  $h_{t-1}$ . Cette combinaison produit un nouvel état caché  $h_t$  (qui sert de "mémoire" du réseau jusqu'à ce point dans la séquence) et, souvent, une sortie  $y_t$  spécifique à ce pas de temps.

Le processus peut être conceptualisé comme le "dépliage" (*unrolling*) du réseau dans le temps. Pour une séquence de longueur  $T$ , le RNN se comporte comme un réseau feed-forward de  $T$  couches, où chaque "couche" correspond à un pas de temps et partage les mêmes poids. C'est ce **partage de poids à travers le temps** qui permet aux RNN de traiter des séquences de longueurs variables et de généraliser à des patterns qui apparaissent à différents moments dans une séquence.

Les équations fondamentales d'un RNN simple sont :

$$h_t = f(W_{hh}h_{t-1} + W_{xh}x_t + b_h)$$

$$y_t = W_{hy}h_t + b_y$$

Où :

- $x_t$  est le vecteur d'entrée au pas de temps  $t$ .
- $h_t$  est le vecteur de l'état caché au pas de temps  $t$ , représentant la mémoire cumulative du réseau.
- $y_t$  est le vecteur de sortie au pas de temps  $t$ .
- $W_{hh}$  est la matrice de poids pour la connexion récurrente (de l'état caché précédent à l'état caché actuel). C'est la "mémoire" qui est transmise.
- $W_{xh}$  est la matrice de poids pour la connexion de l'entrée à l'état caché.
- $W_{hy}$  est la matrice de poids pour la connexion de l'état caché à la sortie.
- $b_h$  et  $b_y$  sont des vecteurs de biais pour l'état caché et la sortie, respectivement.
- $f$  est une fonction d'activation non linéaire, souvent la tangente hyperbolique (tanh) ou ReLU, appliquée élément par élément.

Le processus d'apprentissage dans les RNN utilise une variante de la rétropropagation appelée **Backpropagation Through Time (BPTT)**. BPTT calcule les gradients des poids du réseau par rapport à la perte totale sur l'ensemble de la séquence, en déroulant le réseau sur tous les pas de temps.

---

### 3.4.1 Problèmes Inhérents aux RNN Simples : Vanishing/Exploding Gradients

Bien que les RNN simples aient théoriquement la capacité de traiter des dépendances à long terme, ils souffrent de problèmes pratiques majeurs qui entravent leur apprentissage sur des séquences de longueur significative : les problèmes de **disparition du gradient (vanishing gradient)** et d'**explosion du gradient (exploding gradient)**. Ces problèmes surviennent directement de la nature de la rétropropagation à travers le temps (BPTT) et du partage de poids.

#### Le Problème du Vanishing Gradient (Disparition du Gradient)

Ce phénomène se manifeste lorsque les gradients (les signaux d'erreur utilisés pour ajuster les poids du réseau pendant l'entraînement) deviennent exponentiellement petits à mesure qu'ils sont rétropropagés en arrière à travers de nombreux pas de temps (ou "couches profondes" du réseau déroulé).

- **Mécanisme** : Lors de la rétropropagation, les gradients sont multipliés par les dérivées des fonctions d'activation (par exemple, la dérivée de tanh ou sigmoid est toujours inférieure ou égale à 1, et souvent bien inférieure à 1 sur la majeure partie de leur domaine) et par les matrices de poids récurrents  $W_{hh}$  à chaque pas de temps. Si les valeurs singulières de  $W_{hh}$  sont inférieures à 1 et que les dérivées des activations sont également petites, cette multiplication répétée entraîne une décroissance exponentielle du gradient. Par exemple, si à chaque pas de temps, un gradient est multiplié par 0.5, après 10 pas de temps, il sera réduit à  $0.5^{10} \approx 0.001$ . Après 50 pas, il sera  $0.5^{50} \approx 8.8 \times 10^{-16}$ .
- **Conséquences** :
  - **Oubli des Informations à Long Terme** : Les poids responsables des connexions avec les pas de temps très éloignés (au début d'une longue séquence) reçoivent des gradients si infimes qu'ils ne sont presque pas mis à jour. Le réseau "oublie" efficacement les informations importantes qui sont apparues loin dans le passé de la séquence. Par exemple, dans la phrase "Le chat, qui était très joueur et avait des poils soyeux, mais qui était aussi très indépendant et aimait se promener seul, a

attrapé la souris.", l'information "Le chat" est cruciale pour comprendre le sujet de l'action "a attrapé la souris", mais elle est séparée par de nombreux mots. Un RNN simple aurait du mal à relier ces deux informations.

- **Incapacité à Apprendre des Dépendances à Long Terme** : Le réseau est incapable d'établir des relations de causalité ou de dépendance entre des éléments qui sont éloignés dans la séquence.
- **Difficulté de Convergence** : Les poids apprennent très lentement, ce qui rend l'entraînement inefficace et la convergence difficile, ou conduit à des modèles qui ne peuvent pas généraliser.

### Le Problème de l'Exploding Gradient (Explosion du Gradient)

C'est le phénomène inverse, où les gradients deviennent exponentiellement grands.

- **Mécanisme** : Cela se produit si les valeurs singulières des matrices de poids récurrents  $W_{hh}$  sont supérieures à 1. Dans ce cas, la multiplication répétée par ces valeurs (et les dérivées des activations) provoque une croissance exponentielle du gradient à mesure qu'il est rétropropagé.
- **Conséquences** :
  - **Mises à Jour Instables** : Les mises à jour des poids deviennent massivement grandes, provoquant des changements très brusques et chaotiques dans les paramètres du réseau.
  - **Divergence** : Le réseau peut rapidement diverger, les valeurs des poids et de la perte peuvent atteindre l'infini (NaN/Inf), rendant l'entraînement instable et la convergence impossible.
  - **Détection et Correction** : Bien que moins fréquent que le vanishing gradient, l'exploding gradient est souvent plus facile à détecter (la perte explose soudainement) et à gérer. Une technique courante est le **gradient clipping**, qui limite la norme (magnitude) des gradients à un seuil maximum prédéfini. Si un gradient dépasse ce seuil, il est mis à l'échelle pour rester dans les limites acceptables.

Ces problèmes ont sévèrement limité l'efficacité pratique des RNN simples pour des applications réelles nécessitant de comprendre des contextes longs. C'est pour surmonter ces défis fondamentaux que des architectures de RNN plus sophistiquées, dotées de mécanismes de contrôle du flux d'informations, ont été développées.

---

Pour remédier aux problèmes de vanishing/exploding gradients et permettre aux RNN de capturer efficacement les dépendances à long terme dans les séquences, des architectures plus complexes et dotées de "**mécanismes de gating**" (**portes**) ont été introduites. Ces portes agissent comme des régulateurs intelligents du flux d'informations, permettant au réseau de décider dynamiquement quelles informations doivent être stockées, oubliées, ou transmises à l'étape suivante de la séquence. Elles offrent un chemin de gradient plus direct et stable à travers de nombreux pas de temps.

#### a) LSTM (Long Short-Term Memory)

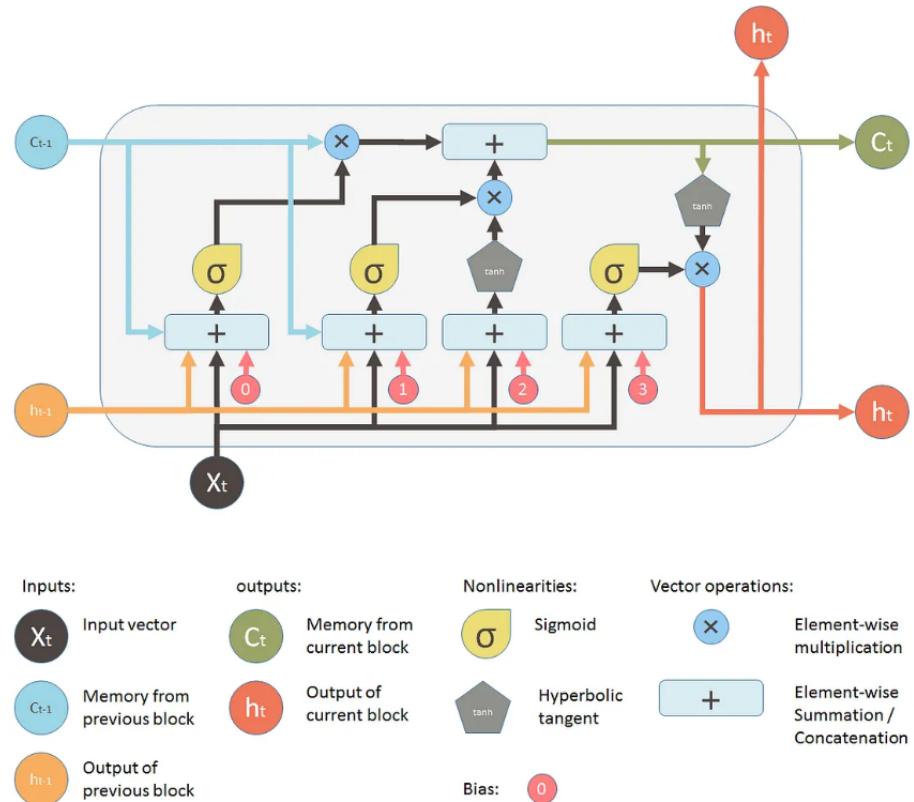


FIGURE 3.2 – Cellule LSTM (Long Short-Term Memory) détaillant les mécanismes de mémoire ( $C_{t-1}$ ,  $C_t$ ), les portes et les transformations non-linéaires (sigmoïde et tanh) qui permettent de résoudre le problème de gradient vanishing.

Les LSTMs, introduits par Sepp Hochreiter et Jürgen Schmidhuber en 1997, sont l'une des innovations les plus significatives et largement adoptées dans le domaine des RNN. Leur succès repose sur l'introduction d'une **cellule de mémoire (cell state –  $C_t$ )** distincte, qui agit comme un "tapis roulant d'informations" traversant la séquence. Cette cellule de mémoire peut transporter des informations pertinentes sur de très longues périodes sans dégradation, et son contenu est contrôlé par un ensemble de **trois portes** distinctes.

L'unité LSTM à l'instant  $t$  prend en entrée :

- $x_t$  : l'entrée courante de la séquence.
- $h_{t-1}$  : l'état caché (ou sortie) de l'unité LSTM précédente à l'instant  $t - 1$ .
- $C_{t-1}$  : l'état de la cellule de mémoire précédent à l'instant  $t - 1$ .

Et elle produit :

- $h_t$  : le nouvel état caché (qui est aussi la sortie de l'unité LSTM).
- $C_t$  : le nouvel état de la cellule de mémoire.

Les trois portes sont des réseaux de neurones feed-forward qui utilisent la fonction d'activation sigmoïde ( $\sigma$ ) pour produire des valeurs entre 0 et 1. Ces valeurs agissent comme des "multiplicateurs" ou des "filtres", permettant à l'information de passer partiellement ou entièrement.

**1. La Porte d'Oubli (Forget Gate –  $f_t$ )** Cette porte est responsable de décider quelles informations de l'état de la cellule précédente ( $C_{t-1}$ ) sont pertinentes et doivent être conservées, et quelles informations doivent être "oubliées" (jetées). Elle prend en entrée l'état caché

précédent  $h_{t-1}$  et l'entrée courante  $x_t$ , les concatène, les multiplie par une matrice de poids  $W_f$ , ajoute un biais  $b_f$ , puis applique une fonction sigmoïde.

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

Le résultat  $f_t$  est un vecteur de valeurs entre 0 et 1. Une valeur proche de 0 indique que l'information correspondante dans  $C_{t-1}$  doit être complètement oubliée, tandis qu'une valeur proche de 1 indique qu'elle doit être conservée.

**2. La Porte d'Entrée (*Input Gate* –  $i_t$ ) et le Candidat d'État de Cellule ( $\tilde{C}_t$ )** Cette porte est responsable de décider quelles nouvelles informations, provenant de l'entrée courante  $x_t$ , doivent être stockées dans l'état de la cellule  $C_t$ . Elle est composée de deux sous-parties :

- La porte d'entrée ( $i_t$ ) elle-même : une couche sigmoïde qui décide quelles valeurs candidates seront mises à jour.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

- La couche candidate d'état de cellule ( $\tilde{C}_t$ ) : une couche tanh qui crée un vecteur de nouvelles valeurs candidates qui pourraient potentiellement être ajoutées à l'état de la cellule.

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

**3. Mise à Jour de l'État de la Cellule ( $C_t$ )** C'est l'étape où l'état de la cellule de mémoire est actualisé. L'état précédent  $C_{t-1}$  est d'abord "filtré" par la porte d'oubli ( $f_t \odot C_{t-1}$ ), puis les nouvelles informations candidates ( $\tilde{C}_t$ ) sont "filtrées" par la porte d'entrée ( $i_t \odot \tilde{C}_t$ ). Ces deux résultats sont ensuite additionnés pour former le nouvel état de la cellule  $C_t$ .

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

Où  $\odot$  représente la multiplication élément par élément (produit d'Hadamard). Ce mécanisme additif est crucial car il permet au gradient de circuler plus librement à travers les pas de temps le long du chemin de l'état de la cellule, atténuant ainsi le problème du vanishing gradient.

**4. La Porte de Sortie (*Output Gate* –  $o_t$ )** Cette porte détermine quelles parties de l'état de la cellule *mis à jour* ( $C_t$ ) doivent être exposées comme l'état caché de sortie ( $h_t$ ) pour le pas de temps actuel. L'état caché ( $h_t$ ) est la sortie effective de l'unité LSTM à ce pas de temps, et c'est ce qui est passé à la prochaine couche ou utilisé comme entrée pour le pas de temps suivant.

- La porte de sortie ( $o_t$ ) : une couche sigmoïde qui décide quelles informations de  $C_t$  sont pertinentes pour la sortie.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

- L'état caché final ( $h_t$ ) : l'état de la cellule  $C_t$  est d'abord passé à travers une tanh (pour normaliser les valeurs), puis multiplié élément par élément par la sortie de la porte de sortie.

$$h_t = o_t \odot \tanh(C_t)$$

### Avantages Majeurs des LSTMs :

- **Gestion Robuste des Dépendances à Long Terme** : Grâce à la cellule de mémoire et aux portes, les LSTMs peuvent sélectivement stocker, manipuler et récupérer des informations sur des périodes beaucoup plus longues que les RNN simples. Le chemin direct à travers l'état de la cellule (le terme  $f_t \odot C_{t-1}$ ) permet aux gradients de se propager plus facilement.

- **Flux de Gradient Stable** : La structure des portes et le chemin additif de l'état de la cellule contribuent à un flux de gradient plus stable et plus constant à travers la séquence, réduisant considérablement les problèmes de vanishing et d'exploding gradients.
- **Performances Supérieures** : Ils ont démontré une performance et une robustesse nettement supérieures aux RNN simples dans une vaste gamme d'applications séquentielles complexes, devenant un standard de facto pour de nombreuses tâches de NLP et de séries temporelles.
- **Flexibilité** : Les LSTMs peuvent être empilés en plusieurs couches (*stacked LSTMs*) pour apprendre des représentations séquentielles plus abstraites ou utilisés dans des architectures bidirectionnelles (*Bi-LSTMs*) pour capturer le contexte des deux directions.

### b) GRU (Gated Recurrent Unit)

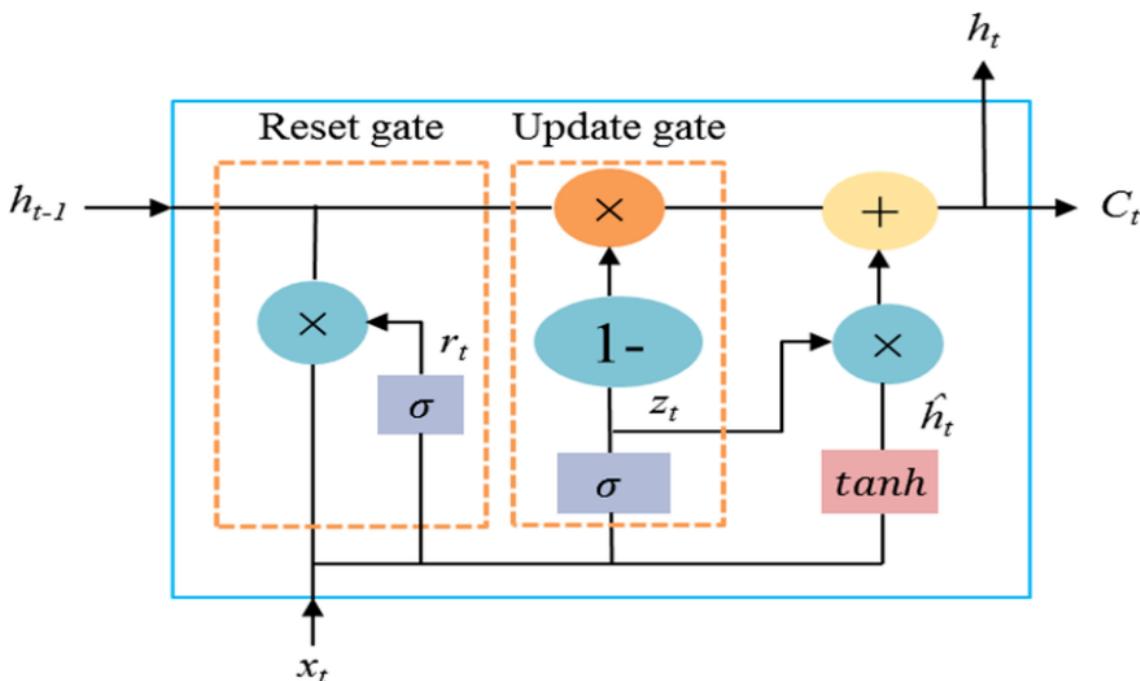


FIGURE 3.3 – Structure GRU (Gated Recurrent Unit) simplifiée avec ses deux portes principales (reset gate et update gate), combinant mécanismes de mémoire et de mise à jour en une architecture plus légère que la LSTM.

Les GRUs, introduits par Kyunghyun Cho et al. en 2014, sont une variante plus récente et simplifiée des LSTMs. Ils ont été développés dans le but d'offrir des performances comparables à celles des LSTMs tout en étant plus efficaces en termes de calcul et de nombre de paramètres, grâce à une architecture de gating fusionnée. Les GRUs combinent la cellule de mémoire et la porte de sortie des LSTMs en un seul état caché ( $h_t$ ) et n'utilisent que deux portes principales : la porte de mise à jour et la porte de réinitialisation.

L'unité GRU à l'instant  $t$  prend en entrée :

- $x_t$  : l'entrée courante de la séquence.
- $h_{t-1}$  : l'état caché (et la sortie) de l'unité GRU précédente à l'instant  $t - 1$ .

Et elle produit :

- $h_t$  : le nouvel état caché (et la nouvelle sortie de l'unité GRU).

**1. La Porte de Mise à Jour (*Update Gate* –  $z_t$ )** Cette porte est une combinaison de la porte d'oubli et de la porte d'entrée des LSTMs. Elle contrôle à la fois la quantité d'informations de l'état caché précédent ( $h_{t-1}$ ) qui doit être conservée et la quantité de nouvelles informations de l'entrée courante ( $x_t$ ) qui doit être ajoutée.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z)$$

Une valeur de  $z_t$  proche de 1 signifie que l'ancien état caché est majoritairement conservé (et que peu de nouvel état candidat est incorporé). Une valeur proche de 0 signifie que l'ancien état caché est majoritairement oublié et remplacé par le nouvel état candidat.

**2. La Porte de Réinitialisation (*Reset Gate* –  $r_t$ )** Cette porte décide quelle partie de l'état caché précédent ( $h_{t-1}$ ) est pertinente pour calculer le nouvel état candidat ( $\tilde{h}_t$ ).

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r)$$

Une valeur de  $r_t$  proche de 0 signifie "réinitialiser" l'état précédent : l'influence de  $h_{t-1}$  est presque entièrement ignorée lors du calcul du nouvel état candidat, forçant le GRU à se concentrer sur l'entrée courante. Une valeur proche de 1 signifie conserver pleinement l'influence de l'état précédent.

**3. Calcul de l'État Candidat ( $\tilde{h}_t$ )** Ce nouvel état candidat est un potentiel nouvel état caché. Il est calculé en utilisant l'entrée courante ( $x_t$ ) et une version "réinitialisée" de l'état caché précédent ( $r_t \odot h_{t-1}$ ), passées à travers une fonction tanh.

$$\tilde{h}_t = \tanh(W_h \cdot [r_t \odot h_{t-1}, x_t] + b_h)$$

**4. Calcul de l'État Caché Final ( $h_t$ )** L'état caché final ( $h_t$ ) est une combinaison linéaire pondérée de l'état caché précédent ( $h_{t-1}$ ) et du nouvel état candidat ( $\tilde{h}_t$ ), contrôlée par la porte de mise à jour ( $z_t$ ).

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

Si  $z_t$  est proche de 1,  $h_t$  sera très proche de  $h_{t-1}$  (l'état est majoritairement conservé). Si  $z_t$  est proche de 0,  $h_t$  sera très proche de  $\tilde{h}_t$  (l'état est majoritairement mis à jour avec le nouveau candidat).

### Avantages des GRUs :

- **Simplicité et Efficacité** : Avec moins de portes et de paramètres que les LSTMs (pas de cellule de mémoire séparée, et les portes d'oubli et d'entrée sont combinées), les GRUs sont généralement plus rapides à entraîner et nécessitent moins de ressources, tout en offrant souvent des performances comparables à celles des LSTMs.
- **Robustesse aux Dépendances à Long Terme** : Comme les LSTMs, les GRUs sont très efficaces pour gérer les dépendances à long terme et atténuer les problèmes de vanishing gradient grâce à leurs mécanismes de gating.
- **Polyvalence** : Les GRUs sont un excellent choix pour de nombreuses tâches séquentielles, en particulier lorsque la vitesse ou les ressources sont une contrainte.

**LSTMs et GRUs pour le sous-titrage vidéo :** Dans le contexte spécifique du sous-titrage vidéo, les LSTMs et GRUs sont des composants architecturaux critiques, formant le "décodeur" ou le "générateur de langage" du système. Le processus typique est le suivant :

1. **Extraction de Caractéristiques Visuelles** : Un CNN (comme VGG, ResNet ou Inception) est utilisé pour extraire un vecteur de caractéristiques spatiales (un *embedding* visuel) pour chaque *frame* significative de la vidéo.
2. **Modélisation Séquentielle** : Une séquence de ces vecteurs de caractéristiques visuelles est ensuite alimentée, pas à pas, à un LSTM ou un GRU. Le réseau récurrent apprend alors à comprendre la dynamique temporelle et les relations entre les événements visuels dans la vidéo. Son état caché encode le "contexte visuel" accumulé jusqu'à l'instant présent.
3. **Génération de Texte** : À chaque pas de temps (ou après avoir traité la séquence vidéo entière), le LSTM/GRU utilise son état caché pour prédire le mot suivant du sous-titre. Cette sortie est souvent passée à travers une couche *softmax* pour obtenir une distribution de probabilités sur le vocabulaire. Le mot avec la plus haute probabilité est choisi, et cette information peut être réintégrée comme entrée pour le pas de temps suivant (dans un mécanisme "encoder-decoder").

La capacité inhérente de ces architectures à maintenir un état contextuel riche et à long terme est ce qui permet de produire des sous-titres qui ne se contentent pas de décrire des objets isolés dans une *frame*, mais qui capturent les actions, les relations et la cohérence narrative sur l'ensemble de la séquence vidéo.

## 3.5 Mécanismes d'Attention (*Attention Mechanisms*)

Dans le monde du *deep learning*, en particulier pour les tâches impliquant des séquences longues ou des données multimodales comme le sous-titrage vidéo, la capacité à "se concentrer" sur les informations les plus pertinentes à un moment donné est cruciale. C'est précisément le rôle des **Mécanismes d'Attention**. Inspirés par la façon dont le cerveau humain se concentre sélectivement sur certaines parties d'une scène visuelle ou d'une conversation, les mécanismes d'attention permettent aux modèles neuronaux de pondérer l'importance de différentes parties de l'entrée lorsqu'ils génèrent une sortie.

Traditionnellement, les modèles de séquence-à-séquence (*Seq2Seq*) basés sur des encodeurs-décodeurs (souvent des RNN comme les LSTMs ou GRUs) devaient compresser toute l'information d'une séquence d'entrée (par exemple, une vidéo entière ou une longue phrase) en un seul vecteur d'état contextuel fixe. Ce vecteur unique, appelé le **vecteur de contexte**, était ensuite passé au décodeur pour générer la séquence de sortie. Cette approche posait des problèmes majeurs pour les séquences longues : le vecteur de contexte devenait un goulet d'étranglement informationnel, incapable de retenir toutes les nuances et détails nécessaires. Le modèle avait tendance à "oublier" les informations du début de la séquence.

Les mécanismes d'attention résolvent ce problème en permettant au décodeur de regarder directement toutes les parties de la séquence d'entrée à chaque étape de la génération de la sortie. Au lieu de s'appuyer sur un seul vecteur de contexte, le décodeur reçoit un "contexte pondéré" qui est dynamiquement ajusté en fonction de ce qu'il est en train de générer.

---

### 3.5.1 Principe Fondamental des Mécanismes d'Attention

Le cœur d'un mécanisme d'attention peut être décomposé en plusieurs étapes clés qui se déroulent à chaque pas de temps où le décodeur génère une nouvelle partie de la sortie (par exemple, un nouveau mot dans une légende) :

**1. Calcul des Scores d'Alignement (ou Scores de Similarité) :** À chaque pas de temps de la génération de la sortie, le décodeur est à un certain état interne (par exemple, l'état caché  $h_t$  d'un LSTM du décodeur). Cet état représente le contexte jusqu'à présent dans la séquence de sortie générée. Le modèle va comparer cet état du décodeur avec **chaque élément de la séquence d'entrée encodée** (par exemple, les états cachés de l'encodeur à chaque pas de temps, ou les caractéristiques visuelles extraites de chaque *frame* vidéo). Le calcul de la similarité entre l'état du décodeur et chaque élément d'entrée produit un **score d'alignement** (ou d'énergie). Plusieurs fonctions peuvent être utilisées pour calculer ces scores :

- **Dot product attention** :  $score(s_t, h_i) = s_t^T h_i$  (simple produit scalaire).
- **Concat attention (ou Additive attention)** :  $score(s_t, h_i) = V_a^T \tanh(W_a s_t + U_a h_i)$  (plus complexe, souvent utilisée pour plus de flexibilité).
- **Scaled Dot Product attention (dans les Transformers)** :  $score(Q, K) = \frac{QK^T}{\sqrt{d_k}}$ .

**2. Normalisation des Scores (Distribution de Probabilités) :** Les scores d'alignement bruts sont ensuite passés à travers une fonction **softmax**. Cette étape convertit les scores en une **distribution de probabilités** sur tous les éléments de la séquence d'entrée. Ces probabilités sont appelées **poids d'attention**. Un poids d'attention élevé pour un élément d'entrée particulier indique que cet élément est très pertinent pour la génération de la sortie actuelle.

$$\alpha_{t,i} = \frac{\exp(score(s_t, h_i))}{\sum_{j=1}^L \exp(score(s_t, h_j))}$$

Où  $s_t$  est l'état du décodeur au temps  $t$ ,  $h_i$  est le  $i$ -ème élément de la séquence d'entrée encodée, et  $L$  est la longueur de la séquence d'entrée.

**3. Calcul du Vecteur de Contexte Pondéré :** Les poids d'attention ( $\alpha_{t,i}$ ) sont ensuite utilisés pour calculer une somme pondérée des éléments de la séquence d'entrée encodée. Ce résultat est le **vecteur de contexte attentif** ( $c_t$ ).

$$c_t = \sum_{i=1}^L \alpha_{t,i} h_i$$

Ce vecteur  $c_t$  ne représente pas une compression fixe de toute l'entrée, mais une représentation dynamique des parties les plus pertinentes de l'entrée, adaptée à l'état actuel du décodeur.

**4. Génération de la Sortie :** Le vecteur de contexte attentif  $c_t$  est ensuite combiné avec l'état courant du décodeur  $s_t$  (et potentiellement l'entrée précédente de la sortie) pour générer le prochain élément de la séquence de sortie. Cette combinaison permet au modèle de prendre sa décision de génération en tenant compte à la fois de son propre état interne et des informations ciblées de l'entrée.

Ce processus est répété à chaque pas de temps de la génération, permettant au modèle d'adapter dynamiquement son "focus" sur l'entrée.

### 3.5.2 Le Rôle Crucial des Mécanismes d'Attention dans le Sous-titrage Vidéo

Dans le contexte du sous-titrage vidéo, les mécanismes d'attention sont absolument fondamentaux et ont révolutionné la qualité des légendes générées. Un système de sous-titrage typique se compose généralement de deux parties principales :

1. **Un Encodeur Visuel (CNN)** : Pour extraire des caractéristiques visuelles de chaque *frame* de la vidéo. Cela génère une séquence de vecteurs de caractéristiques, un pour chaque *frame* (ou groupe de *frames*).
2. **Un Décodeur Séquentiel (RNN, souvent LSTM/GRU)** : Pour générer le texte du sous-titre mot par mot.

Sans attention, le décodeur devrait créer une légende basée sur une représentation globale de toute la vidéo, ce qui est très difficile pour des vidéos de durée moyenne ou longue. Les informations visuelles importantes pourraient se noyer dans la masse.

Avec l'attention, le processus de sous-titrage devient beaucoup plus précis et interprétable :

- **Lien des Mots aux Moments Spécifiques de la Vidéo** : C'est la fonction la plus puissante de l'attention ici. Lorsque le décodeur génère un mot, le mécanisme d'attention lui permet de "regarder" les *frames* spécifiques de la vidéo qui sont les plus pertinentes pour ce mot.
  - Par exemple, si le modèle est en train de générer le mot "homme", il peut se concentrer sur les *frames* où un homme est visible.
  - Lorsque le modèle génère le mot "sauter", l'attention se portera sur les *frames* montrant l'action de sauter.
  - Quand il écrit "balle", il peut se concentrer sur l'objet "balle" dans l'image.
- **Amélioration de la Cohérence et de la Précision** : En se concentrant dynamiquement, le modèle peut générer des descriptions plus cohérentes et précises, évitant les erreurs ou les omissions dues à un manque de focus. Il n'est plus limité par la capacité d'un unique vecteur de contexte à résumer toute la vidéo.
- **Gestion des Longues Séquences Vidéo** : Les vidéos peuvent être très longues, avec des centaines, voire des milliers de *frames*. Sans attention, capturer toutes les informations temporelles et spatiales pertinentes pour chaque mot serait impossible. L'attention permet de "naviguer" à travers la séquence visuelle et de ne récupérer que les informations pertinentes au moment opportun.
- **Interprétabilité (Visualisation de l'Attention)** : Un avantage significatif de l'attention est qu'elle offre un certain degré d'interprétabilité. En visualisant les poids d'attention, on peut voir quelles parties de l'image ou de la vidéo le modèle "regarde" lorsqu'il génère chaque mot. Cela peut aider à comprendre pourquoi le modèle a généré une certaine légende et à déboguer les erreurs. On peut, par exemple, superposer la carte d'attention sur les *frames* vidéo pour voir les régions chaudes (les plus attendues).
- **Réduction du Goulot d'Étranglement Informationnel** : L'attention contourne le problème du goulot d'étranglement du vecteur de contexte fixe, car le décodeur a un accès "à la demande" à l'intégralité des représentations encodées.

### 3.5.3 Types d'Attention dans les Systèmes de Sous-titrage Vidéo

Dans les systèmes de sous-titrage, l'attention peut opérer de différentes manières :

- **Attention Spatiale** : Un mécanisme d'attention peut se concentrer sur des **régions spécifiques d'une seule image** (*frame*). Par exemple, pour générer le mot "chien", le modèle peut se concentrer sur les pixels correspondant à l'image du chien dans la *frame* actuelle. Ce type d'attention est souvent appliqué sur la carte de caractéristiques 2D (ou 3D si on considère les canaux comme une dimension) d'une *frame* générée par le CNN.

- **Attention Temporelle** : Un mécanisme d'attention peut se concentrer sur des *frames* spécifiques dans la séquence vidéo. Par exemple, si une action comme "sauter" se déroule sur quelques *frames* clés, le modèle peut apprendre à pondérer ces *frames* plus fortement lors de la génération des mots décrivant cette action.
- **Attention Spatio-Temporelle** : Les modèles les plus avancés combinent ces deux types, permettant au décodeur de se concentrer à la fois sur des régions spécifiques *et* sur des moments spécifiques de la vidéo. Cela permet une compréhension beaucoup plus fine du contenu visuel dynamique.

L'intégration des mécanismes d'attention dans les architectures encodeur-décodeur (CNN-RNN pour le sous-titrage) a été une avancée majeure, permettant de passer de descriptions génériques à des légendes contextuellement riches, précises et fluides, en établissant un lien direct et dynamique entre les informations visuelles et le langage généré. C'est l'une des raisons pour lesquelles les systèmes de sous-titrage modernes sont devenus si performants.

## 3.6 Modèles Encodeur-Décodeur (*Encoder-Decoder Models*)

Le paradigme **Encodeur-Décodeur** est une architecture neuronale fondamentale et incroyablement polyvalente, ayant révolutionné la façon dont nous abordons les problèmes dits de "séquence-à-séquence" (**Seq2Seq**). Ces modèles excellent dans les tâches où l'entrée et la sortie sont toutes deux des séquences, mais potentiellement de longueurs différentes et dans des modalités différentes. Que ce soit la traduction automatique (une séquence de mots dans une langue vers une autre séquence de mots dans une autre langue), la synthèse vocale (une séquence de texte vers une séquence de données audio), ou, le cœur de votre projet, le **sous-titrage vidéo** (une séquence d'images vidéo vers une séquence de mots), les modèles encodeur-décodeur sont la pierre angulaire de ces avancées.

L'élégance de cette architecture réside dans sa capacité à diviser un problème complexe en deux sous-problèmes gérables et distincts mais interdépendants :

1. **L'Encodeur** : Sa mission est de comprendre la séquence d'entrée et de la compresser en une représentation dense, riche en informations et de taille fixe (ou un ensemble de représentations). Cette représentation est souvent appelée le **vecteur de contexte** ou l'état de pensée. Il capture l'essence sémantique et syntaxique de l'entrée.
2. **Le Décodeur** : Sa mission est de prendre cette représentation concise de l'encodeur et de la "décompresser" ou de la "générer" séquentiellement en la séquence de sortie souhaitée, élément par élément.

Ce découplage permet à chaque composant d'être spécialisé dans sa tâche, tout en collaborant efficacement pour accomplir l'objectif global.

---

### 3.6.1 Explication Détaillée : Le Mécanisme Encodeur-Décodeur

Format	Entrelacé/ Progressif	Rapport largeur/ hauteur en pixels		Représentation (pixels virtuels)
<b>SD PAL</b>	e, p	720 × 576* (5:4)		<b>4:3</b> (768 × 576) 
		720 × 576		<b>16:9</b> (1024 × 576) 
		720 × 434		<b>16:9</b> (1024 × 576) 
<b>SD NTSC</b>	e, p	640 × 480** (4:3)		<b>4:3</b> (640 × 480) 
		720 × 480 (3:2)		<b>4:3</b> (640 × 480) 
<b>HD</b> «Full HD»	e, p	1920 × 1080 (16:9)		<b>16:9</b> (1920 × 1080) 
<b>HD</b>	p	1280 × 720 (16:9)		<b>16:9</b> (1280 × 720) 
<b>HDV</b> «Full HD» anamorphosé	e	1440 × 1080 (4:3)		<b>16:9</b> (1920 × 1080) 

FIGURE 3.4 – Architecture d'un auto-encodeur montrant les composants principaux : l'encodeur (transformant l'entrée  $x_1$  à  $x_{66}$  en code latent via les couches cachées  $h_1$  à  $h_n$ ) et le décodeur (reconstruisant les sorties  $x'_1$  à  $x'_{66}$  à partir des états cachés  $h'_1$  à  $h'_{n-1}$ ).

Décomposons plus finement le fonctionnement de ces deux piliers architecturaux.

## 1. L'Encodeur : Comprendre et Condenser l'Entrée

L'encodeur est la première partie du modèle Seq2Seq. Son rôle est de lire la séquence d'entrée (quelle que soit sa modalité : texte, images vidéo, audio) et d'en extraire les caractéristiques pertinentes, puis de les condenser en une représentation significative.

- **Entrée de l'Encodeur :** L'entrée est une séquence  $X = (x_1, x_2, \dots, x_N)$ , où  $N$  est la longueur de la séquence d'entrée. Chaque  $x_i$  est un élément de cette séquence.
  - **Pour le texte :** Chaque  $x_i$  serait un **embedding de mot** (une représentation vectorielle dense d'un mot).
  - **Pour la vidéo (sous-titrage) :** Chaque  $x_i$  serait un **vecteur de caractéristiques visuelles** (généralement extrait d'une *frame* ou d'un segment vidéo par un Réseau Neuronai Convolutif - CNN). Ces vecteurs peuvent représenter des objets, des actions, des scènes, etc.
  - **Pour l'audio :** Chaque  $x_i$  pourrait être un ensemble de **coefficients MFCC** ou d'autres caractéristiques acoustiques.
- **Architecture de l'Encodeur :** L'encodeur est typiquement un **Réseau de Neurones Récurrent (RNN)**, souvent un **LSTM** ou un **GRU**, ou une pile de ces unités. Pour des tâches modernes, il peut également s'agir de couches de **Transformers**.
  - L'encodeur parcourt la séquence d'entrée, élément par élément. À chaque pas de temps  $i$ , il traite  $x_i$  et l'état interne précédent  $h_{i-1}$  pour calculer un nouvel état interne  $h_i$ .

- Les RNN sont naturellement adaptés à cette tâche car ils peuvent capturer les dépendances temporelles et contextuelles au fur et à mesure qu'ils traitent la séquence.
- Les encodeurs bidirectionnels (Bi-RNN, Bi-LSTM, Bi-GRU) sont souvent utilisés pour capturer le contexte à la fois du passé et du futur de chaque élément d'entrée.
- **Sortie de l'Encodeur (le Vecteur de Contexte Initial)** : Après avoir traité toute la séquence d'entrée  $X$ , l'encodeur produit une représentation finale.
  - Dans les architectures Seq2Seq plus anciennes (sans attention), cette représentation est un unique **vecteur de contexte fixe** (souvent le dernier état caché de l'encodeur  $h_N$ , ou la concaténation des derniers états cachés pour un encodeur bidirectionnel). Ce vecteur est censé encapsuler toutes les informations pertinentes de la séquence d'entrée.
  - Dans les architectures avec attention (comme nous l'avons vu précédemment), l'encodeur ne produit pas un seul vecteur de contexte final, mais plutôt une **séquence de représentations intermédiaires** (par exemple, tous les états cachés  $h_1, h_2, \dots, h_N$ ). C'est cette séquence qui sera ensuite utilisée par le décodeur et le mécanisme d'attention. L'encodeur extrait donc un ensemble de "faits" ou de "caractéristiques" de l'entrée.

**En résumé, la fonction de l'encodeur est de transformer une séquence d'entrée  $X$  de longueur variable en une ou plusieurs représentations vectorielles qui encapsulent son information sémantique.**

## 2. Le Décodeur : Générer la Séquence de Sortie

Le décodeur est la deuxième partie du modèle. Son rôle est de prendre la représentation extraite par l'encodeur et de la transformer en une séquence de sortie  $Y = (y_1, y_2, \dots, y_M)$ , élément par élément, où  $M$  est la longueur de la séquence de sortie (qui peut être différente de  $N$ ).

- **Entrée Initiale du Décodeur** : Le décodeur est initialisé avec le vecteur de contexte fourni par l'encodeur. Dans les architectures sans attention, c'est ce vecteur unique qui est le point de départ. Dans les architectures avec attention, le décodeur est souvent initialisé avec un état initial dérivé des états de l'encodeur, et il a un accès continu à l'ensemble des représentations de l'encodeur via le mécanisme d'attention.
- **Architecture du Décodeur** : Le décodeur est également un **RNN** (LSTM ou GRU) ou une architecture basée sur les **Transformers**. Il fonctionne en mode "génératif", produisant un élément à la fois.
  - À chaque pas de temps  $t$  (pour la séquence de sortie), le décodeur prend en entrée l'élément  $y_{t-1}$  qu'il a généré précédemment (ou un jeton de début de séquence pour  $y_1$ ) et son état caché précédent  $s_{t-1}$ .
  - Il combine ces informations avec le **contexte fourni par l'encodeur**.
    - **Sans attention** : Le même vecteur de contexte fixe de l'encodeur est utilisé à chaque pas de temps. Cela est limitant car toutes les informations doivent être contenues dans ce seul vecteur.
    - **Avec attention** : C'est là que la magie opère. Au lieu d'un contexte fixe, le décodeur utilise son état courant  $s_{t-1}$  pour calculer dynamiquement un **vecteur de contexte attentif** ( $c_t$ ) en "regardant" toutes les représentations intermédiaires de l'encodeur. Ce  $c_t$  est une somme pondérée des éléments encodés, les poids étant déterminés par la pertinence de chaque élément par rapport à l'état actuel du décodeur (comme détaillé dans la section 3.4).

— **Génération de la Sortie :**

- Le décodeur utilise l'état courant ( $s_t$ ) et le vecteur de contexte (fixe ou attentif  $c_t$ ) pour calculer une distribution de probabilités sur le vocabulaire de sortie.
- Le mot (ou élément) avec la probabilité la plus élevée est sélectionné comme  $y_t$ .
- Ce mot généré  $y_t$  est ensuite réutilisé comme entrée pour le pas de temps suivant du décodeur, jusqu'à ce qu'un jeton de fin de séquence soit généré ou que la longueur maximale soit atteinte.

En résumé, la fonction du décodeur est de convertir la représentation condensée de l'encodeur en une séquence de sortie, en générant un élément à la fois et en s'appuyant sur les éléments précédemment générés.

---

### 3.6.2 Le Paradigme Encodeur-Décodeur Appliqué au Sous-titrage Vidéo

L'architecture Encodeur-Décodeur est le modèle dominant pour le sous-titrage vidéo, souvent appelée **CNN-RNN Encoder-Decoder with Attention**.

— **L'Encodeur (le "C" de CNN dans CNN-RNN) :**

- **Entrée :** Une séquence de *frames* vidéo (ou des clips vidéo courts).
- **Architecture :** Généralement, un **Réseau Neuronal Convolutif (CNN)** pré-entraîné (comme ResNet, VGG, Inception) est utilisé pour extraire des caractéristiques visuelles riches et sémantiques de chaque *frame* (ou groupe de *frames*).
- **Processus :** Chaque *frame* vidéo  $x_i^{frame}$  est passée à travers le CNN pour obtenir un vecteur de caractéristiques visuelles  $v_i$ . La séquence d'entrée pour le décodeur devient donc  $V = (v_1, v_2, \dots, v_N)$ , où  $N$  est le nombre de *frames* échantillonnées ou de segments vidéo.
- **Rôle :** L'encodeur visuel transforme le contenu visuel brut de la vidéo en une série de représentations compactes qui peuvent être comprises par le décodeur textuel. Il s'agit d'une étape cruciale d'abstraction.

— **Le Décodeur (le "RNN" de CNN-RNN) :**

- **Entrée :** La séquence de caractéristiques visuelles  $V$  (de l'encodeur) et, à chaque pas de temps, le mot précédemment généré.

- **Architecture :** Typiquement un **Réseau de Neurones Récurrent (RNN)**, comme un LSTM ou un GRU, ou un Transformer Decoder.

— **Processus (avec Attention) :**

1. Initialisation de l'état du décodeur.

2. *Loop* : Pour générer le mot  $y_t$  du sous-titre :

- Le décodeur calcule un **vecteur de contexte attentif**  $c_t$  en "regardant" la séquence de caractéristiques visuelles  $V = (v_1, \dots, v_N)$  de l'encodeur. Le mécanisme d'attention pondère les *frames* (ou segments vidéo) les plus pertinentes pour le mot en cours de génération, en fonction de l'état interne du décodeur.
- Le décodeur utilise  $c_t$ , son état interne  $s_{t-1}$ , et le mot précédent  $y_{t-1}$  pour calculer le nouvel état interne  $s_t$  et générer une distribution de probabilités sur les mots du vocabulaire.
- Le mot  $y_t$  est échantillonné (ou choisi avec le plus de probabilité) à partir de cette distribution.

- Le processus se répète jusqu'à la fin de la séquence.
- **Rôle :** Le décodeur textuel prend les informations visuelles pertinentes (focalisées par l'attention) et les transforme en une séquence de mots cohérente et sémantiquement fidèle au contenu vidéo. Il gère la grammaire, la sémantique linguistique et la fluidité de la légende générée.

**Implications pour le Sous-titrage Vidéo :** Le paradigme Encodeur-Décodeur, en particulier lorsqu'il est renforcé par les mécanismes d'attention, est exceptionnellement puissant pour le sous-titrage. L'encodeur se concentre sur la compréhension visuelle de la vidéo, tandis que le décodeur s'occupe de la traduction de cette compréhension en langage naturel. La capacité des mécanismes d'attention à lier dynamiquement les mots générés aux moments visuels spécifiques a permis des avancées majeures, produisant des sous-titres non seulement pertinents mais aussi chronologiquement alignés avec les événements de la vidéo. C'est l'architecture de base de la plupart des systèmes de sous-titrage vidéo de pointe.

## CHAPITRE 4

# ARCHITECTURES DE DEEP LEARNING POUR LE SOUS-TITRAGES VIDÉO

## 4.1 Approches Générales du Video Captioning : Décoder le Récit Visuel

Le **Video Captioning**, ou sous-titrage automatique de vidéos, est une tâche fascinante et complexe à l'intersection de la vision par ordinateur et du traitement du langage naturel. Son objectif est de générer des descriptions textuelles cohérentes et sémantiquement riches qui encapsulent le contenu visuel et temporel d'une vidéo. Ce n'est pas seulement reconnaître des objets, mais comprendre des actions, des interactions, et la progression narrative à travers le temps. Pour y parvenir, plusieurs approches générales ont émergé, chacune apportant des innovations cruciales.

### 4.1.1 "Show and Tell" (Encoder-Decoder) : Les Architectures Pionnières

L'une des premières et des plus influentes architectures pour le sous-titrage automatique, qu'il s'agisse d'images ou de vidéos, est le paradigme **Encodeur-Décodeur**, souvent popularisé sous le nom évocateur de "**Show and Tell**". Cette approche a établi la feuille de route pour de nombreux modèles ultérieurs en divisant la tâche complexe en deux phases distinctes mais complémentaires : l'observation (l'encodage visuel) et la narration (le décodage textuel).

#### L'Encodeur : Observer la Vidéo

Dans le contexte du *Video Captioning*, l'**encodeur** est responsable de la compréhension et de la transformation du contenu visuel brut de la vidéo en une représentation numérique que le modèle peut traiter. Contrairement au sous-titrage d'images où un simple CNN suffit, la vidéo introduit la dimension temporelle, rendant l'encodage plus nuancé.

##### — Extraction de Caractéristiques Visuelles (CNN) :

- La première étape consiste à échantillonner la vidéo en une séquence de **frames** (images individuelles) ou de **segments vidéo courts**.
- Chaque *frame* (ou chaque segment agrégé) est ensuite passée à travers un **Réseau Neuronale Convolutif (CNN)** pré-entraîné sur de vastes ensembles de données

de classification d'images (comme ImageNet). Des architectures populaires comme **VGG**, **ResNet**, **Inception** ou plus récemment **EfficientNet** sont couramment utilisées.

- Le CNN extrait un vecteur de caractéristiques spatiales haut niveau pour chaque *frame*. Ces vecteurs agissent comme des "embeddings visuels", capturant des informations sur les objets, les scènes, et les attributs visuels présents à cet instant précis.
- **Modélisation de la Séquence Temporelle (RNN/LSTM) :**
  - Puisque la vidéo est une séquence d'événements, ces vecteurs de caractéristiques extraits par le CNN pour chaque *frame* doivent être traités séquentiellement. C'est ici qu'intervient la composante récurrente de l'encodeur.
  - Une couche de **Réseaux de Neurones Récursifs (RNN)**, le plus souvent un **LSTM (Long Short-Term Memory)** ou un **GRU (Gated Recurrent Unit)**, est utilisée pour ingérer cette séquence de vecteurs visuels.
  - Le rôle du RNN de l'encodeur est de comprendre la **dynamique temporelle** des caractéristiques visuelles. Il apprend à relier les événements passés aux événements actuels, à identifier des actions qui se déroulent sur plusieurs *frames*, et à créer une représentation globale et contextuelle de la vidéo.
  - Dans sa forme la plus simple, l'état caché final de ce RNN encodeur est le "vecteur de contexte" qui encapsule toute l'information pertinente de la vidéo. C'est ce vecteur qui est ensuite passé au décodeur.

## Le Décodeur : Raconter l'Histoire

Le **décodeur** prend la représentation condensée de la vidéo fournie par l'encodeur et la transforme en une séquence de mots, formant ainsi le sous-titre.

- **Génération de Séquence de Mots (RNN/LSTM) :**
  - Le décodeur est également un **RNN**, typiquement un **LSTM** ou un **GRU**, qui est initialisé avec le vecteur de contexte généré par l'encodeur.
  - À chaque pas de temps de la génération, le décodeur reçoit comme entrée le mot qu'il a généré précédemment (ou un jeton de début de séquence pour le premier mot) et son état caché interne.
  - Il utilise ces informations, combinées avec le contexte vidéo fourni par l'encodeur, pour prédire le mot suivant le plus probable dans la séquence de sous-titre.
  - Ce processus itératif se poursuit jusqu'à ce qu'un jeton de fin de séquence soit généré, indiquant la fin du sous-titre.
- **Limitations du "Show and Tell" Simple :** Bien que pionnière, cette approche simple a une limitation majeure : le fait de compresser l'intégralité de la vidéo en un seul vecteur de contexte fixe peut créer un **goulot d'étranglement informationnel**. Pour des vidéos de durée moyenne ou longue, ce vecteur unique peut ne pas suffire à retenir tous les détails pertinents ou les dépendances à long terme. Le décodeur pourrait alors avoir du mal à générer des descriptions précises pour des événements se produisant tard dans la vidéo, car les informations initiales pourraient être "diluées" ou "oubliées".

### 4.1.2 Approches Basées sur l'Attention : Rendre les Descriptions Pertinentes

Pour surmonter les limitations du goulot d'étranglement des architectures encodeur-décodeur simples, les **Mécanismes d'Attention** ont été introduits et ont transformé le domaine du *Video Captioning*. L'attention permet au modèle de **dynamiquement focaliser son "regard"** sur les parties les plus pertinentes de la vidéo (qu'elles soient spatiales ou temporelles) au moment précis où il génère un mot du sous-titre.

#### Comment l'Attention Améliore la Pertinence :

Dans une architecture encodeur-décodeur avec attention pour le sous-titrage vidéo, l'encodeur (souvent un CNN suivi d'un RNN) ne produit plus un seul vecteur de contexte final. Au lieu de cela, il produit une **séquence d'états cachés intermédiaires** (ou de représentations visuelles de *frames* ou de segments), chacun représentant un aspect spécifique et localisé de la vidéo.

Lorsque le décodeur génère un mot ( $y_t$ ), il ne s'appuie pas sur un unique vecteur global. Au lieu de cela, le mécanisme d'attention procède comme suit :

1. **Calcul de la Pertinence** : À chaque pas de temps de décodage, l'état actuel du décodeur (qui représente le contexte linguistique généré jusqu'à présent) est comparé avec **chacun des états intermédiaires de l'encodeur**. Cette comparaison produit un ensemble de "scores de pertinence" ou "scores d'alignement".
2. **Pondération Dynamique** : Ces scores de pertinence sont ensuite normalisés (généralement via une fonction *softmax*) pour obtenir des **poids d'attention**. Ces poids indiquent l'importance relative de chaque segment vidéo ou *frame* encodé par rapport au mot que le décodeur est sur le point de générer. Par exemple, si le décodeur s'apprête à générer le mot "courir", les *frames* montrant l'action de courir recevront des poids d'attention élevés.
3. **Contexte Pondéré Personnalisé** : Un nouveau **vecteur de contexte attentif** est calculé comme une somme pondérée de tous les états intermédiaires de l'encodeur, en utilisant les poids d'attention déterminés à l'étape précédente. Ce vecteur de contexte est donc **dynamique et spécifique** au mot qui est en train d'être généré.
4. **Génération Améliorée** : Le décodeur utilise ensuite ce vecteur de contexte attentif, son propre état interne et le mot précédemment généré pour prédire le mot suivant.

#### Rôle Crucial dans le Sous-titrage :

L'impact des mécanismes d'attention sur le *Video Captioning* est monumental :

- **Alignment Précis Mot-Vidéo** : L'attention permet d'établir un lien explicite et interprétable entre chaque mot généré et les moments spécifiques de la vidéo qui sont les plus pertinents pour ce mot. Cela signifie que la légende n'est pas seulement descriptive, mais aussi **temporellement alignée**. Par exemple, le mot "sauter" sera lié aux *frames* où l'action de sauter se produit.
- **Descriptions plus Pertinentes et Détaillées** : En permettant au modèle d'accéder directement aux détails nécessaires sans les compresser dans un unique vecteur, les descriptions générées deviennent beaucoup plus précises, détaillées et contextuellement appropriées, même pour des vidéos longues et complexes.
- **Atténuation du Problème du Goulot d'Étranglement** : L'attention résout efficacement le problème de la perte d'information sur les longues séquences en fournissant un "accès mémoire" dynamique à l'encodeur.

- **Amélioration de la Fluidité et de la Cohérence :** Le fait de pouvoir se concentrer sur les bonnes informations au bon moment contribue à une meilleure fluidité linguistique et à une plus grande cohérence sémantique des sous-titres produits.
- **Interprétabilité Accrue :** Les cartes d'attention peuvent être visualisées pour montrer quelles parties de la vidéo (quelles *frames* ou quelles régions dans les *frames*) le modèle a "regardées" pour générer chaque mot. Cela est précieux pour comprendre le comportement du modèle et pour le débogage.

## 4.2 Architectures de Décodage Textuel : Transformer la Vision en Langage

Après que l'encodeur vidéo a extrait des représentations riches du contenu visuel et temporel d'une vidéo, la tâche du **décodeur textuel** est de transformer ces informations en une description linguistique cohérente et sémantiquement pertinente. C'est là que la magie de la génération de langage opère, transformant des vecteurs numériques en mots intelligibles. Les architectures de décodage textuel les plus courantes s'appuient sur des réseaux de neurones récurrents, souvent augmentés par des mécanismes d'attention pour une précision accrue.

### 4.2.1 RNN/LSTM/GRU : La Génération Séquentielle de Mots

Les **Réseaux de Neurones Récurrents (RNN)**, en particulier leurs variantes plus sophistiquées comme les **Long Short-Term Memory (LSTM)** et les **Gated Recurrent Unit (GRU)**, sont la pierre angulaire des architectures de décodage textuel dans le sous-titrage vidéo. Leur nature séquentielle est parfaitement adaptée à la génération mot par mot de phrases.

- **Comment ça fonctionne ?**
  - Le décodeur est initialisé avec une représentation contextuelle de la vidéo fournie par l'encodeur. Dans les modèles les plus simples (sans attention), c'est un unique vecteur de contexte. Dans les modèles avec attention, le décodeur est initialisé, et il accède dynamiquement aux caractéristiques de l'encodeur.
  - À chaque étape de la génération, le décodeur reçoit l'état caché de l'étape précédente ( $h_{t-1}$ ) et le mot qu'il a généré juste avant ( $y_{t-1}$ ).
  - Ces informations sont combinées avec le contexte vidéo pour calculer un nouvel état caché ( $h_t$ ).
  - Cet état  $h_t$  est ensuite utilisé pour prédire le mot suivant ( $y_t$ ) dans la séquence. La prédiction est généralement faite en projetant  $h_t$  dans la taille du vocabulaire et en appliquant une fonction **softmax** pour obtenir une distribution de probabilités sur tous les mots possibles. Le mot avec la plus haute probabilité est choisi.
  - Ce mot choisi est ensuite fourni comme entrée pour l'étape suivante, permettant au décodeur de construire la phrase mot par mot de manière auto-régressive. Ce processus continue jusqu'à ce qu'un jeton de fin de phrase soit généré, ou qu'une longueur maximale de phrase soit atteinte.
- **Pourquoi LSTM/GRU sont préférables aux RNN simples ?**
  - Comme discuté précédemment, les RNN simples souffrent des problèmes de **vani-shing/exploding gradients** sur de longues séquences. Les phrases générées pour les sous-titres vidéo peuvent être de longueur variable, et un contexte linguistique riche est nécessaire.

- Les **LSTM** et **GRU** résolvent ce problème grâce à leurs "portes" internes (**gates**) qui leur permettent de sélectivement retenir ou oublier des informations au fil du temps. Cela leur confère une "mémoire" à long terme essentielle pour maintenir la cohérence grammaticale et sémantique sur l'ensemble de la phrase générée. Ils peuvent ainsi mieux suivre les dépendances complexes entre les mots dans une phrase.
- 

#### 4.2.2 Mécanismes d'Attention : Moduler le Décodeur pour un Focus Pertinent

Bien que les RNN/LSTM/GRU soient excellents pour la génération séquentielle, leur combinaison avec les **mécanismes d'attention** est ce qui a réellement propulsé la performance du sous-titrage vidéo. L'attention permet au décodeur de ne pas seulement s'appuyer sur une représentation globale de la vidéo (potentiellement insuffisante), mais de **dynamiquement ajuster son focus** sur les parties les plus pertinentes de la vidéo à mesure qu'il génère chaque mot.

##### — Comment l'Attention Module le Décodeur ?

- Au lieu de recevoir un unique vecteur de contexte fixe de l'encodeur, le décodeur, à chaque pas de temps ( $t$ ), accède à l'**ensemble des caractéristiques visuelles intermédiaires** qui ont été extraites par l'encodeur pour chaque segment ou *frame* de la vidéo (par exemple,  $v_1, v_2, \dots, v_N$ ).
- Lorsque le décodeur est sur le point de générer le mot  $y_t$ , il effectue une série de calculs (comme détaillé dans la Section 3.4) :
  1. Il compare son propre état interne courant (représentant le contexte linguistique généré jusqu'à présent) avec chaque caractéristique visuelle de la séquence encodée.
  2. Ces comparaisons donnent lieu à des **scores d'alignement**, qui mesurent la pertinence de chaque partie de la vidéo par rapport à l'étape de génération actuelle.
  3. Les scores sont normalisés en **poids d'attention** via une fonction softmax. Ces poids sont une distribution de probabilités, indiquant sur quelles *frames* ou segments vidéo le décodeur devrait "porter son attention" en ce moment précis.
  4. Un **vecteur de contexte attentif** est alors calculé comme une somme pondérée des caractéristiques visuelles encodées, avec les poids d'attention agissant comme coefficients. Ce vecteur est donc une représentation **personnalisée et ciblée** de la vidéo pour le mot  $y_t$ .
- Ce vecteur de contexte attentif est ensuite fusionné avec l'état caché du décodeur et le mot précédemment généré. C'est cette combinaison enrichie qui est utilisée pour prédire le prochain mot.

##### — Impact sur la Qualité des Sous-titres :

- **Précision Accrue** : L'attention garantit que les mots générés sont directement liés aux éléments visuels les plus pertinents de la vidéo. Par exemple, si une personne apparaît pour la première fois à la 10ème seconde, le mot "personne" sera généré avec une forte attention sur ces *frames* précises.
- **Cohérence Temporelle** : Les actions et les descriptions sont mieux alignées avec les événements visuels réels de la vidéo, évitant les décalages ou les descriptions erronées dues à un contexte global imprécis. **Détail et Spécificité** : Le décodeur

peut capter des détails fins de la vidéo en se concentrant sur de petites régions ou de courts instants, ce qui permet de générer des légendes plus riches et plus spécifiques.

- **Robustesse aux Longues Vidéos :** L'attention surmonte le problème du "goulot d'étranglement" en permettant au décodeur d'accéder à toutes les informations de l'encodeur à la demande, rendant les modèles plus efficaces pour sous-titrer des vidéos de longue durée.

En combinant la puissance de la modélisation séquentielle des RNN/LSTM/GRU avec la capacité de focalisation des mécanismes d'attention, les architectures de décodage textuel modernes sont capables de produire des sous-titres vidéo qui sont non seulement grammaticalement corrects, mais aussi sémantiquement précis et temporellement alignés avec le contenu visuel. C'est cette synergie qui rend le sous-titrage automatique si impressionnant aujourd'hui.

## 4.3 Architectures Avancées et Tendances Actuelles : Repousser les Limites du Sous-titrage Vidéo

Le domaine du sous-titrage vidéo est en constante évolution. La recherche continue de surmonter les défis existants et d'améliorer la qualité, la diversité et la pertinence des descriptions générées. L'introduction de nouvelles architectures et l'accent mis sur l'intégration multimodale ainsi que la diversité des sorties sont au cœur des tendances actuelles.

### 4.3.1 Transformers pour le Video Captioning : L'Ascension de l'Auto-Attention

Alors que les architectures CNN-RNN avec mécanismes d'attention ont dominé le paysage du sous-titrage vidéo pendant un certain temps, l'émergence des **Transformers** a marqué un changement de paradigme significatif. Introduits par Google en 2017, les Transformers ont révolutionné le traitement du langage naturel (NLP) et montrent un potentiel immense pour les tâches multimodales comme le *Video Captioning*.

- **Pourquoi les Transformers ?** Les RNN (LSTM/GRU) traitent les séquences de manière séquentielle, un pas de temps à la fois. Cela peut limiter leur capacité à capturer efficacement les **dépendances à très long terme** car les informations doivent traverser de nombreuses étapes. De plus, le traitement séquentiel rend le **parallélisme** difficile pendant l'entraînement. Les Transformers résolvent ces problèmes grâce à leur mécanisme **d'auto-attention (Self-Attention)**.
- **Le Cœur des Transformers : Le Mécanisme de Multi-Head Attention** Au lieu de s'appuyer sur la récurrence, les Transformers utilisent des couches de **Multi-Head Attention** (attention multi-têtes). Ce mécanisme permet au modèle de pondérer l'importance de **chaque élément** de la séquence d'entrée (ou de sortie précédemment générée) par rapport à tous les autres éléments, et ce, **en parallèle**.
- **Dans l'Encodeur (Video Transformer)** : Les caractéristiques visuelles extraites de chaque *frame* ou segment vidéo (par exemple, par un CNN 2D ou 3D) sont traitées par des couches de Multi-Head Attention. Chaque "tête" de l'attention peut se concentrer sur différentes relations ou aspects des *frames* vidéo. Cela permet à l'encodeur de comprendre les relations complexes entre les événements visuels, même s'ils sont très éloignés dans le temps. Par exemple, une tête pourrait se concentrer sur l'évolution d'un objet, tandis qu'une autre pourrait suivre une action.

- **Dans le Décodeur (Text Transformer)** : Le décodeur utilise également la Multi-Head Attention. Il peut "regarder" les caractéristiques visuelles encodées par le Transformer encodeur (attention encodeur-décodeur) tout en considérant les mots qu'il a déjà générés (auto-attention masquée) pour prédire le mot suivant.
  - **Avantages clés pour le Video Captioning :**
    - **Capture de Dépendances à Long Terme** : Le mécanisme d'attention directe entre tous les éléments permet une meilleure modélisation des relations complexes et des dépendances à très long terme dans les vidéos, ce qui est crucial pour des narrations précises.
    - **Parallélisation** : Le calcul de l'attention peut être parallélisé, ce qui accélère considérablement l'entraînement des modèles sur de grandes quantités de données vidéo.
    - **Interactions Multimodales Améliorées** : Les Transformers sont intrinsèquement bien adaptés à la gestion des interactions entre différentes modalités. Ils peuvent apprendre à aligner les jetons linguistiques et les caractéristiques visuelles de manière plus riche et flexible que les RNN.
  - **Application Potentielle** : Les Transformers peuvent former des architectures purement attentionnées pour le sous-titrage, où à la fois l'encodeur vidéo et le décodeur textuel sont des Transformers. On peut également trouver des architectures hybrides, par exemple, un CNN 3D pour l'encodage vidéo initial, suivi d'un Transformer encodeur pour les relations temporelles, et un Transformer décodeur pour la génération de texte.
- 

#### 4.3.2 Approches Multimodales : Vers une Compréhension Enrichie

Les vidéos ne sont pas que des images ; elles sont souvent accompagnées de sons, de musique et parfois même de dialogues (sous forme de texte transcrit). Les **approches multimodales** visent à intégrer et à fusionner ces différentes sources d'information (visuel, audio, texte existant) pour une compréhension plus holistique de la vidéo, ce qui devrait conduire à des sous-titres plus riches et plus précis.

- **Pourquoi l'Intégration Multimodale ?** Le langage humain est naturellement multimodal. Pour décrire une scène, nous utilisons non seulement ce que nous voyons, mais aussi ce que nous entendons.
  - Le son peut indiquer une action invisible (ex : un coup de feu, une porte qui claque).
  - Les dialogues peuvent fournir des informations narratives cruciales (qui parle, de quoi, le contexte).
  - La musique peut influencer le ton ou l'émotion de la scène.
- **Stratégies d'Intégration :**
  - **Fusion Précoce (Early Fusion)** : Les caractéristiques extraites de chaque modalité (par exemple, caractéristiques CNN pour la vidéo, caractéristiques audio par un réseau spécifique) sont concaténées très tôt dans le modèle et traitées ensemble. C'est simple mais peut entraîner des problèmes si les modalités ne sont pas alignées temporellement ou ont des granularités très différentes.
  - **Fusion Tardive (Late Fusion)** : Chaque modalité est encodée indépendamment, et les représentations de haut niveau sont fusionnées juste avant le décodeur ou même au niveau de la fonction de perte. Cela permet à chaque encodeur de se spécialiser, mais peut manquer d'interactions fines.

- **Fusion Intermédiaire/Attentionnée** : C'est l'approche la plus prometteuse. Des mécanismes d'attention multimodale sont utilisés pour apprendre dynamiquement comment combiner les informations de différentes modalités. Par exemple, le décodeur peut avoir des mécanismes d'attention distincts pour les caractéristiques visuelles et les caractéristiques audio, ou une attention conjointe qui pondère leur pertinence combinée.
    - Un exemple pourrait être un modèle qui, en générant le mot "aboyer", accorde de l'attention à la fois aux *frames* montrant un chien et aux segments audio contenant un son d'abolement.
  - **Architectures Spécifiques** : Les modèles peuvent utiliser des encodeurs distincts pour chaque modalité (ex : CNN pour vidéo, CNN 1D ou RNN pour audio, Word Embeddings + RNN pour texte/dialogue), puis une couche de fusion avant le décodeur, ou des Transformers multimodaux capables de traiter des jetons de différentes modalités simultanément.
- 

### 4.3.3 Génération de Descriptions Diversifiées : Au-delà d'une Unique Vérité

Un défi majeur et une tendance de recherche importante dans le *Video Captioning* est la **génération de descriptions diversifiées**. Pour une même vidéo, il existe souvent plusieurs façons correctes et sémantiquement valides de la décrire. Les modèles traditionnels ont tendance à générer la description la plus "moyenne" ou la plus probable, ce qui peut rendre les résultats répétitifs et manquer de la richesse et de la créativité du langage humain.

- **Le Défi de la Diversité :**
  - Les modèles sont généralement entraînés à maximiser la vraisemblance de la description de référence unique dans l'ensemble de données. Cela les pousse à produire des phrases génériques qui sont statistiquement proches de la moyenne des données, sacrifiant la diversité.
  - Un modèle peut capter des aspects différents d'une même scène. Par exemple, une vidéo d'une personne qui saute peut être décrite comme "une personne saute", "un athlète exécute un saut", ou "le jeune homme atterrit après un saut". Toutes sont valides.
- **Méthodes pour Générer la Diversité :**
  - **Beam Search avec Pénalité de Diversité** : Plutôt que de choisir le mot le plus probable à chaque étape, l'algorithme de Beam Search explore plusieurs chemins de génération. On peut ajouter une pénalité qui décourage la sélection de phrases trop similaires à celles déjà générées dans le même faisceau.
  - **Génération par Échantillonnage (*Sampling-based Generation*)** : Au lieu de prendre le mot avec la probabilité la plus élevée (*argmax*), on peut échantillonner les mots à partir de la distribution de probabilités prédictive par le modèle.
    - **Top-K Sampling** : Échantillonner uniquement parmi les K mots les plus probables.
    - **Nucleus Sampling (Top-P Sampling)** : Échantillonner parmi l'ensemble le plus petit de mots dont la somme des probabilités dépasse un seuil P. Cela aide à éviter les mots peu probables tout en maintenant la diversité.

- **Modèles Conditionnels (*Conditional Generation*)** : Entraîner le modèle à générer des descriptions en fonction d'un "style" ou d'une "perspective" donnée (par exemple, descriptions courtes vs. détaillées, descriptions axées sur les personnes vs. les objets). Cela peut être fait en ajoutant un vecteur de contrôle à l'entrée du décodeur.
- **Generative Adversarial Networks (GANs) et Variational Autoencoders (VAEs)** : Ces architectures génératives peuvent être adaptées pour produire des descriptions multiples en apprenant une distribution latente sur les descriptions. Un discriminateur (dans les GANs) ou une fonction de perte spécifique (dans les VAEs) peut encourager la diversité des sorties.
- **Auto-Encoder Adversarial (AAE)** : Similaire aux VAE, mais utilisant un adversaire pour forcer l'espace latent à correspondre à une distribution *a priori* (par exemple, gaussienne), à partir de laquelle on peut échantillonner pour générer diverses descriptions.

Ces architectures avancées et ces domaines de recherche illustrent la complexité et le potentiel du *Video Captioning*. En tirant parti de modèles plus puissants comme les Transformers, en intégrant toutes les informations disponibles via des approches multimodales, et en visant des sorties plus diversifiées, nous nous rapprochons de systèmes capables de comprendre et de décrire le monde visuel avec une richesse et une flexibilité comparables à celles de l'être humain.

## CHAPITRE 5

### BASES DE DONNÉES ET MÉTRIQUES D'ÉVALUATION

#### 5.1 Bases de Données Communes pour le Video Captioning : Le Carburant de l'Apprentissage

Les performances des modèles de sous-titrage vidéo dépendent crucialement de la qualité et de la quantité des données d'entraînement. Ces bases de données fournissent des paires vidéo-légende qui permettent aux réseaux neuronaux d'apprendre les correspondances complexes entre le contenu visuel et les descriptions linguistiques. Voici quelques-unes des bases de données les plus influentes et largement utilisées dans ce domaine :

##### 5.1.1 MSVD (Microsoft Research Video Description Corpus)

Le **MSVD** (parfois appelé YouTube2Text) fut l'une des premières bases de données de grande envergure spécifiquement conçues pour le *Video Captioning*.

- **Caractéristiques** : Il se compose de courts clips vidéo extraits de YouTube, généralement de 10 à 40 secondes. Chaque clip est annoté avec plusieurs descriptions en anglais fournies par des annotateurs humains. Les vidéos couvrent une grande variété de sujets et d'activités quotidiennes.
- **Taille** : Environ **1 970 clips vidéo uniques**, accompagnés d'environ **80 000 descriptions textuelles** (environ 40-50 descriptions par vidéo).
- **Intérêt** : Malgré sa taille relativement modeste par rapport aux bases de données plus récentes, MSVD a été instrumental dans le développement des premières architectures de *Video Captioning*. Sa taille gérable le rendait accessible pour la recherche initiale et les tests rapides de concepts.

##### 5.1.2 MSR-VTT (Microsoft Research Video to Text)

Le **MSR-VTT** est une base de données plus récente et plus grande, visant à adresser la nécessité de données plus diverses et volumineuses pour le *Video Captioning*.

- **Caractéristiques** : Similaire à MSVD, elle contient des clips vidéo de YouTube, mais avec une plus grande diversité de sujets et de styles (actualités, musique, sports, vie quotidienne, etc.). L'objectif était de capturer une plus grande variété de concepts et

de relations entre les objets et les actions. Chaque vidéo est également accompagnée de plusieurs descriptions humaines.

- **Taille :** Comprend environ **10 000 clips vidéo uniques**, avec un total d'environ **200 000 descriptions textuelles** (20 descriptions par vidéo).
  - **Intérêt :** MSR-VTT est devenu une base de données de référence pour l'évaluation comparative des modèles, en raison de sa plus grande taille et de la diversité de son contenu, ce qui permet de tester la généralisation des modèles.
- 

### 5.1.3 VATEX (Video and Text)

**VATEX** est une base de données plus récente et ambitieuse, qui se distingue par son caractère **multilingue**.

- **Caractéristiques :** Elle contient des vidéos YouTube avec des descriptions non seulement en anglais, mais aussi dans plusieurs autres langues, notamment le chinois. Chaque vidéo est annotée par des locuteurs natifs dans chaque langue, fournissant ainsi des paires vidéo-texte parallèles pour le sous-titrage multilingue. Les vidéos couvrent une très large gamme d'activités humaines.
  - **Taille :** Environ **41 269 clips vidéo uniques**, avec **10 descriptions en anglais** et **10 descriptions en chinois** par vidéo, totalisant environ **825 200 descriptions**.
  - **Intérêt :** VATEX est crucial pour la recherche sur le sous-titrage vidéo multilingue et pour tester la robustesse des modèles à travers différentes cultures et structures linguistiques. Elle encourage le développement de modèles capables de comprendre le contenu vidéo de manière indépendante de la langue, ou de générer des légendes dans plusieurs langues cibles.
- 

### 5.1.4 LSMDC (Large-Scale Movie Description Challenge)

Contrairement aux clips YouTube de courte durée, **LSMDC** se concentre sur des **descriptions de films plus longues et narratives**.

- **Caractéristiques :** Cette base de données est dérivée de longs-métrages. Les clips sont plus longs et les descriptions sont souvent des légendes entières ou des extraits de scénarios qui décrivent des scènes complexes, des dialogues et des arcs narratifs. Cela pose des défis différents des courtes descriptions d'actions simples.
  - **Taille :** Environ **120 868 clips vidéo extraits de 200 films**, avec un total de près de **158 114 descriptions**. Chaque vidéo a généralement une description unique et plus longue.
  - **Intérêt :** LSMDC est particulièrement utile pour développer des modèles capables de comprendre des contextes vidéo étendus, des relations inter-scènes, et de générer des descriptions qui reflètent des aspects narratifs plus profonds. C'est un terrain de jeu pour les modèles visant une compréhension plus "humaine" de la narration visuelle.
-

### 5.1.5 Défis Associés à Ces Bases de Données

Malgré leur importance capitale, ces bases de données ne sont pas sans défis :

1. **Taille et Échelle** : Bien que des bases de données comme VATEX et LSMDC soient "grandes", elles restent modestes par rapport aux téraoctets de données d'images et de texte disponibles pour d'autres tâches. L'acquisition et l'annotation de vidéos sont coûteuses et chronophages, ce qui limite la taille des bases de données. Une taille limitée peut restreindre la capacité des modèles à généraliser à des scénarios très variés.
2. **Bruit et Variabilité des Descriptions** :
  - **Bruit d'Annotation** : Les descriptions humaines peuvent contenir des erreurs grammaticales, des fautes d'orthographe ou des incohérences.
  - **Variabilité Subjective** : Différents annotateurs peuvent décrire la même scène de manière très différente, en se concentrant sur des aspects distincts (une personne, une action, une émotion, un décor). Cette variabilité, bien que naturelle, peut rendre l'apprentissage difficile pour un modèle qui tente de s'aligner sur une "vérité" unique. Les métriques d'évaluation peinent également à capturer cette richesse.
3. **Ambiguïté et Manque de Contexte** : Les clips vidéo sont souvent courts et décontextualisés. Un modèle peut ne pas avoir suffisamment d'informations pour comprendre pleinement l'intention ou le contexte d'une action. Par exemple, un geste peut avoir des significations différentes selon le contexte du film.
4. **Déséquilibre des Classes et des Événements** : Certaines actions ou objets sont beaucoup plus fréquents que d'autres, créant un déséquilibre dans les données qui peut rendre difficile l'apprentissage des descriptions pour les événements rares.
5. **Coût et Complexité de l'Annotation** : L'annotation de vidéos est intrinsèquement plus difficile que celle d'images en raison de la dimension temporelle. Il faut non seulement décrire ce qui est visible, mais aussi comment cela évolue dans le temps, ce qui ajoute à la complexité et au coût.
6. **Dépendance à la Langue Anglaise** : Historiquement, la majorité des bases de données étaient unilingues anglais. Bien que VATEX comble cette lacune pour le chinois, le besoin de bases de données multilingues diversifiées reste important pour un déploiement global.

Malgré ces défis, ces bases de données ont été et restent le fondement sur lequel les progrès en *Video Captioning* sont construits. Les efforts continus pour créer des ensembles de données plus grands, plus propres et plus diversifiés sont essentiels pour le futur de la recherche.

## 5.2 Métriques d'Évaluation des Sous-Titrages : Mesurer la Qualité de la Narration Automatique

Dans le domaine du *Video Captioning* (et plus généralement de la génération de texte), il ne suffit pas de générer une description ; il faut aussi évaluer sa qualité. Les **métriques d'évaluation** sont absolument cruciales. Elles fournissent un moyen objectif et quantifiable de mesurer la performance des modèles, de comparer différentes approches, et de guider la recherche vers des systèmes toujours plus performants. Sans ces métriques, l'amélioration des modèles serait subjective et laborieuse, se basant uniquement sur l'inspection visuelle des résultats, ce qui est impraticable à grande échelle.

Cependant, évaluer un texte généré est intrinsèquement complexe. Contrairement à la classification où une réponse est "juste" ou "fausse", une vidéo peut être décrite de multiples façons

valides. Les métriques cherchent à quantifier la similarité entre la description générée par le modèle (le "candidat") et une ou plusieurs descriptions de référence (les "références humaines").

---

### 5.2.1 BLEU (Bilingual Evaluation Understudy) : Le Pionnier

La métrique **BLEU** est l'une des plus anciennes et des plus largement utilisées dans le domaine de l'évaluation de la traduction automatique, mais elle a rapidement été adoptée pour le sous-titrage et la génération de texte.

- **Principe** : BLEU mesure la précision des *n-grammes* (séquences de *n* mots) entre le texte candidat et les textes de référence. Il calcule combien de *n-grammes* (unigrammes, bigrammes, trigrammes, quadrigrammes) du texte candidat apparaissent dans au moins l'une des références.
  - **Précision Modifiée** : Pour éviter que des modèles ne génèrent des mots répétés à l'infini (et obtiennent un score élevé), BLEU utilise une "précision modifiée" où chaque *n-gramme* du candidat est compté un maximum de fois égal au nombre de fois qu'il apparaît dans n'importe quelle référence.
  - **Pénalité de Brièveté (Brevity Penalty - BP)** : BLEU pénalise les candidats qui sont significativement plus courts que les références, car des phrases très courtes peuvent avoir une précision élevée mais manquer d'informations.
  - **Formule générale** :  $BLEU = BP \cdot \exp \left( \sum_{n=1}^N w_n \log P_n \right)$  où  $P_n$  est la précision des *n-grammes* et  $w_n$  est le poids des *n-grammes* (souvent  $1/N$ ).
  - **Avantages** :
    - **Simplicité et Rapidité** : Facile à calculer et à comprendre.
    - **Indépendance Linguistique** : Ne nécessite aucune connaissance linguistique explicite, juste des correspondances de mots.
    - **Standardisation** : Très répandu, ce qui facilite la comparaison des résultats entre différentes études.
  - **Limites** :
    - **Focus sur la Précision** : Ne mesure que la précision, pas le rappel. Un modèle peut générer une description très courte et parfaite (avec peu de mots) et obtenir un score BLEU élevé tout en manquant beaucoup d'informations.
    - **Insensibilité Sémantique** : Ne prend pas en compte la signification des mots ou la structure syntaxique. Des synonymes parfaits qui ne sont pas présents dans les références seront pénalisés. Par exemple, "un homme est assis" et "un type est assis" sont sémantiquement très proches mais peuvent avoir un score BLEU faible si la référence ne contient que "un homme est assis".
    - **Manque de Fluidité** : Un score BLEU élevé ne garantit pas que la phrase est grammaticalement correcte ou fluide.
    - **Dépendance aux Références Multiples** : Fonctionne mieux avec plusieurs références humaines pour capturer la variabilité.
- 

### 5.2.2 METEOR (Metric for Evaluation of Translation With Explicit Ordering) : L'Amélioration du Rappel

**METEOR** a été développé pour remédier à certaines des lacunes de BLEU, en particulier en incorporant le **rappel** et en étant plus flexible sur les correspondances de mots.

- **Principe :** METEOR calcule une moyenne harmonique pondérée de la précision et du rappel basés sur des correspondances entre les mots candidats et références. Ces correspondances ne sont pas seulement exactes, mais peuvent aussi inclure :
    - **Synonymie :** Utilise WordNet (une base de données lexicale) pour identifier les synonymes. Par exemple, "voiture" et "automobile" seront considérés comme une correspondance.
    - **Racines (*Stemming*) :** Les mots avec la même racine morphologique sont mis en correspondance (ex : "courir", "court", "courant").
    - **Alignement :** Trouve le meilleur alignement un-à-un entre les mots du candidat et des références, en pénalisant les correspondances qui brisent l'ordre des mots (même si l'ordre est moins strict que BLEU).
    - **Pénalité pour la discontinuité :** Une pénalité est appliquée si les correspondances alignées ne sont pas contiguës (c'est-à-dire si des mots sont déplacés dans la phrase).
  - **Améliorations par rapport à BLEU :**
    - **Prise en Compte du Rappel :** Équilibre la précision et le rappel, ce qui rend la métrique moins susceptible de favoriser des descriptions trop courtes.
    - **Robustesse aux Synonymes et Variations :** Capte mieux la similarité sémantique grâce à l'utilisation de WordNet et du *stemming*.
    - **Meilleure Corrélation avec le Jugement Humain :** S'est avéré mieux corrélée avec les évaluations humaines de la qualité que BLEU, notamment pour la fluidité et l'adéquation.
  - **Limites :**
    - **Dépendance Linguistique :** Nécessite des ressources linguistiques (comme WordNet) spécifiques à la langue évaluée, ce qui la rend moins universelle que BLEU.
    - **Complexité de Calcul :** Plus complexe et plus lent à calculer que BLEU.
- 

### 5.2.3 ROUGE (Recall-Oriented Understudy for Gisting Evaluation) : L'Évaluation Axée sur le Rappel

**ROUGE** est une suite de métriques souvent utilisée pour l'évaluation de résumés automatiques et de la traduction. Contrairement à BLEU qui est axé sur la précision, ROUGE est axé sur le **rappel**, mesurant à quel point le texte de référence est couvert par le texte candidat.

- **Principe :** ROUGE compte le nombre de co-occurrences d'unités (mots, *n-grammes*, paires de mots, séquences de mots, etc.) entre le texte candidat et l'ensemble de références.
- **ROUGE-N :** Mesure le rappel des *n-grammes*. Par exemple, ROUGE-1 compte le rappel des unigrammes, ROUGE-2 celui des bigrammes. Il se concentre sur le nombre de *n-grammes* des références qui sont présents dans le candidat.
- **ROUGE-L :** Basé sur la plus longue sous-séquence commune (*Longest Common Subsequence - LCS*). Il ne nécessite pas que les correspondances soient contiguës, ce qui est utile pour évaluer la fluidité et l'ordre des mots.
- **ROUGE-S :** Basé sur les paires de mots (*skip-bigrams*) qui ne sont pas nécessairement contiguës.
- **Avantages :**
  - **Accent sur le Rappel :** Très utile pour évaluer si le modèle a inclus toutes les informations importantes des références. Cela est particulièrement pertinent pour le sous-titrage, où il est important de ne rien "oublier" de la vidéo.

- **Bonne Corrélation pour le Rappel :** Particulièrement efficace pour les tâches de résumé et de description où la couverture du contenu est primordiale.
  - **Limites :**
    - **Moins Adapté à la Génération :** Bien que les variantes de ROUGE mesurent la précision, elles sont moins robustes que BLEU ou METEOR pour la *génération* pure de phrases grammaticalement correctes et fluides. Elles peuvent favoriser des candidats qui "copient" beaucoup de mots de référence.
    - **Interprétation :** Les scores peuvent être plus difficiles à interpréter intuitivement que BLEU pour des tâches de génération complète de phrases.
- 

#### 5.2.4 CIDEr (Consensus-based Image Description Evaluation) : Spécifique à la Description Visuelle

CIDEr a été spécialement conçu pour l'évaluation des sous-titrages d'images, et a été largement étendu au sous-titrage vidéo. Il vise à mesurer le consensus d'une description générée par rapport à un ensemble de descriptions de référence humaines.

- **Principe :** CIDEr calcule la similarité d'un candidat à un ensemble de références en mesurant la fréquence des *n-grammes* (généralement jusqu'à 4-grammes) en utilisant la notion de **TF-IDF** (*Term Frequency-Inverse Document Frequency*).
  - **TF-IDF des *n-grammes* :** Les *n-grammes* qui sont fréquents dans la description générée et peu fréquents dans l'ensemble de données global (donc plus spécifiques et informatifs) reçoivent un poids plus élevé. Cela permet à la métrique de favoriser les descriptions qui capturent les détails pertinents plutôt que des mots génériques.
  - **Consensus :** En comparant le candidat à un ensemble de références multiples (typiquement 5 à 10 par image/vidéo), CIDEr évalue à quel point le candidat s'aligne avec le consensus des annotateurs humains.
  - **Avantages :**
    - **Très Bonne Corrélation avec l'Humain :** S'est avéré être la métrique la plus fortement corrélée avec le jugement humain pour les tâches de *Image/Video Captioning*.
    - **Favorise la Spécificité :** La pondération TF-IDF encourage le modèle à produire des descriptions détaillées et spécifiques plutôt que génériques.
    - **Conçu pour le Domaine Visuel :** Optimisé pour capturer les aspects qui importent le plus dans la description de contenu visuel.
  - **Limites :**
    - **Nécessite de Multiples Références :** Pour être efficace, CIDEr a besoin de plusieurs descriptions de référence par vidéo pour calculer le consensus.
    - **Sensibilité à la Base de Données :** La partie IDF (*Inverse Document Frequency*) est calculée sur l'ensemble de la base de données d'entraînement, ce qui signifie que le score CIDEr peut être spécifique à cette base de données et moins transférable directement à d'autres.
-

### 5.2.5 SPICE (Semantic Propositional Image Captioning Evaluation) : L'Évaluation Sémantique

**SPICE** est une métrique plus récente qui se distingue en se concentrant sur la **pertinence sémantique** des descriptions, plutôt que sur la correspondance exacte des mots ou des *n-grammes*.

- **Principe :** SPICE analyse la description générée et les références pour extraire des **propositions sémantiques** (par exemple, des tuples sujet-verbe-objet, des attributs, des relations spatiales). Pour ce faire, elle utilise un analyseur syntaxique et sémantique (typiquement basé sur des graphes de scènes).
- **Graphes de Scènes :** Elle représente le contenu de la légende et des références sous forme de graphes de scènes, où les nœuds sont des objets, des attributs ou des relations, et les arêtes relient ces entités.
- **Mesure de F-score :** Elle calcule un F-score (combinaison de précision et de rappel) entre les propositions sémantiques extraites du candidat et celles extraites des références.
- **Avantages :**
  - **Évaluation Sémantique :** Évalue ce que la description *signifie* plutôt que la façon dont elle est formulée. Cela permet de juger des descriptions avec des formulations différentes mais des sens identiques.
  - **Robustesse aux Reformulations :** Moins sensible aux variations lexicales ou syntaxiques. Si le modèle dit "une personne court sur la route" et la référence dit "un homme court sur le chemin", SPICE peut identifier la similarité sémantique des actions et des entités.
  - **Interprétabilité :** Les propositions sémantiques peuvent être plus faciles à comprendre que les scores agrégés.
- **Limites :**
  - **Dépendance à l'Analyseur Sémantique :** La performance de SPICE est fortement dépendante de la précision de l'outil d'analyse syntaxique et sémantique sous-jacent, qui peut être coûteux en calcul et ne pas être parfait, surtout pour des langues complexes ou du texte non standard.
  - **Lenteur :** L'extraction de graphes de scènes est plus lente que les comparaisons de *n-grammes*.
  - **Focus sur le Contenu :** Peut ne pas capturer la fluidité linguistique ou la grammaticalité de manière aussi directe que d'autres métriques.

---

### 5.2.6 L'Importance d'Utiliser Plusieurs Métriques pour une Évaluation Complète

Comme il a été démontré, chaque métrique a ses propres forces et faiblesses, et capture des aspects différents de la qualité d'un sous-titre généré :

- **BLEU** et **ROUGE** évaluent principalement la correspondance des mots et des phrases, avec un accent sur la précision (BLEU) ou le rappel (ROUGE).
- **METEOR** apporte une amélioration en intégrant la synonymie et en équilibrant précision/rappel.

- **CIDEr** cible spécifiquement la pertinence et le consensus dans le contexte des descriptions visuelles.
- **SPICE** va au-delà des mots pour évaluer la correspondance sémantique des concepts.

Par conséquent, l'utilisation d'une **suite de métriques** est non seulement recommandée, mais **essentielle** pour obtenir une évaluation complète et nuancée des modèles de *Video Captioning*. Un modèle peut exceller sur une métrique (par exemple, générer des phrases très spécifiques pour un bon CIDEr) mais échouer sur une autre (par exemple, manquer de fluidité ou de couverture contextuelle pour un BLEU ou ROUGE faible).

Une évaluation multi-métriques permet aux chercheurs de :

- **Comprendre les Forces et Faiblesses** : Identifier les aspects où un modèle surpassé ou échoue.
- **Guider l'Amélioration** : Déterminer les axes de développement (améliorer la spécificité, la grammaticalité, la capture d'événements rares, etc.).
- **Corrélation avec l'Humain** : Les métriques automatiques sont des substituts. Une combinaison de métriques qui corrèlent bien avec le jugement humain (comme METEOR et CIDEr) est préférée.
- **Comparaison Robuste** : Fournir une base de comparaison plus solide entre différentes approches, évitant les biais qui pourraient découler de l'utilisation d'une seule métrique.

En somme, les métriques d'évaluation sont le baromètre de la recherche en *Video Captioning*. Elles nous permettent de mesurer les progrès, de comprendre les performances de nos modèles sous différents angles, et de nous rapprocher de l'objectif ultime : des systèmes capables de décrire les vidéos avec la même richesse et la même nuance qu'un être humain.

## CHAPITRE 6

### ARCHITECTURE ET IMPLÉMENTATION DU MODÈLE DE VIDEO CAPTIONING

Ce chapitre est dédié à la présentation détaillée de la mise en œuvre pratique de notre système de Video Captioning automatique. Faisant suite à la conceptualisation théorique, nous aborderons ici les choix techniques et les étapes concrètes qui ont permis de transformer les fondements de notre approche en une solution fonctionnelle. Nous décrirons méticuleusement le processus de préparation des données, en mettant en lumière les spécificités liées à l'utilisation du vaste jeu de données MSR-VTT. Par la suite, la conception et l'architecture de notre modèle encodeur-décodeur, incluant l'intégration des réseaux neuronaux récurrents pour le traitement des séquences visuelles et textuelles, seront expliquées en profondeur. Enfin, nous détaillerons la stratégie d'entraînement adoptée, ainsi que les méthodes d'inférence pour la génération des légendes. Les résultats et l'évaluation des performances de notre modèle seront présentés dans les sections finales de ce chapitre, une fois les phases d'entraînement et de test complétées.

#### 6.1 Acquisition et Structuration du Jeu de Données MSR-VTT : Le Cœur de notre Apprentissage Multimodal

Au fondement de tout système d'intelligence artificielle performant réside la qualité et la pertinence de ses données d'entraînement. Pour notre projet de Video Captioning, notre choix s'est naturellement porté sur le **MSR-VTT** (**Microsoft Research Video to Text**), un jeu de données qui s'est imposé comme une référence incontournable dans le domaine de la compréhension vidéo et de la génération de langage. Sa richesse est pluridimensionnelle : il ne s'agit pas seulement d'une vaste collection, mais d'une mosaïque de vidéos courtes, sélectionnées pour leur diversité thématique – allant des scènes du quotidien aux extraits d'émissions populaires, en passant par des clips musicaux et des démonstrations techniques. Cette hétérogénéité constitue un terrain d'expérimentation idéal pour forger un modèle capable de saisir la complexité sémantique et la dynamique temporelle du monde visuel et de les traduire en un langage naturel cohérent.

— **Provenance et Accès Rétentif aux Données** : L'acquisition du MSR-VTT, indispensable à nos phases d'expérimentation, a été orchestrée via une méthode standardisée et transparente, garantissant la reproductibilité de notre démarche. Conscient de la nature collaborative de la recherche, nous avons utilisé la puissance de l'environnement Google Colab pour cloner directement le dépôt public hébergeant le dataset. La commande `!git clone https://github.com/vth/MSR-VTT.git` a servi de passerelle vers cette précieuse

ressource, nous permettant d'intégrer sans heurts la collection complète de vidéos et les métadonnées associées dans notre environnement de travail. Cette approche non seulement sécurise l'accès aux données, mais assure également une conformité avec les versions de dataset utilisées par la communauté scientifique.

- **Cartographie Détailée de l'Organisation des Fichiers : La Symphonie des Médias et des Mots :** Une fois téléchargé, le MSR-VTT révèle une structure logique, conçue pour une exploitation aisée des données multimodales.
  - **Le Corpus Vidéo (VideoCorpus) :** C'est le cœur visuel du dataset. Il se compose d'une multitude de fichiers vidéo au format .mp4, chacun identifié de manière unique et séquentielle (par exemple, de video0.mp4 à video9999.mp4 pour le dataset complet). Chaque clip, d'une durée généralement comprise entre 10 et 30 secondes, est une capsule temporelle capturant une action ou un événement spécifique, exigeant de notre modèle une compréhension dynamique et contextuelle.
  - **Le Trésor des Annotations Textuelles (MSRVTT\_data.json) :** La richesse linguistique du dataset réside dans ce fichier JSON centralisé. Il ne s'agit pas d'une simple association un-à-un, mais d'une architecture relationnelle où chaque video\_id est méticuleusement lié à **20 légendes textuelles distinctes**, rédigées par des annotateurs humains. Cette surabondance de descriptions est un atout inestimable : elle offre à notre modèle une perspective polyphonique sur le même événement visuel, lui permettant d'appréhender la diversité des expressions linguistiques pour un concept donné, et d'affiner sa capacité à générer des légendes à la fois précises et variées. Cette redondance est également cruciale pour une évaluation robuste, car elle fournit de multiples références fiables pour mesurer la qualité des légendes générées.
- **Stratégie de Sous-échantillonnage et Division des Données : Naviguer dans l'Immense avec Précision (5 000 Vidéos) :** Le jeu de données MSR-VTT, dans son intégralité, représente un volume colossal de données (10 000 vidéos et 200 000 légendes), dont le traitement et l'entraînement exigent des ressources computationnelles considérables. Afin d'optimiser l'efficacité de nos expérimentations sans sacrifier la représentativité du dataset, nous avons stratégiquement choisi de travailler sur un sous-ensemble gérable de **5 000 vidéos**. Cette décision pragmatique nous a permis de capitaliser sur la puissance de calcul limitée des GPU disponibles (notamment sur Google Colab) tout en assurant que notre modèle soit exposé à une diversité suffisante de scénarios visuels et de descriptions linguistiques.

A partir de ce sous-ensemble de 5 000 vidéos, nous avons procédé à une division rigoureuse pour constituer nos ensembles d'entraînement, de validation et de test :

  - **Ensemble d'Entraînement :** La majeure partie des données a été allouée à l'entraînement du modèle. Sur les 5 000 vidéos sélectionnées, un total de **4 000 vidéos** a été désigné pour l'entraînement. Ceci représente 80% de notre sous-ensemble et correspond à **80 000 légendes** (4 000 vidéos × 20 légendes par vidéo), constituant la base d'apprentissage pour les poids de notre réseau neuronal.
  - **Ensemble de Validation :** Un ensemble distinct a été réservé pour la validation, essentiel au suivi des performances du modèle au cours de l'entraînement et à la détection du surapprentissage. Nous avons alloué **500 vidéos** à cet ensemble, soit 10% du sous-ensemble, correspondant à **10 000 légendes**. Cet ensemble permet d'ajuster les hyperparamètres et de décider de l'arrêt précoce de l'entraînement sans biaiser les performances finales.
  - **Ensemble de Test :** Enfin, les **500 vidéos** restantes (10% du sous-ensemble, soit **10 000 légendes**) ont été désignées comme ensemble de test. Cet ensemble sera utilisé une seule fois, à la fin du processus d'entraînement et d'optimisation, pour

fournir une évaluation impartiale et finale de la capacité de généralisation de notre modèle sur des données totalement inédites.

Cette répartition garantit que le modèle est évalué sur des données qu'il n'a jamais vues, assurant ainsi la fiabilité des métriques de performance finales.

video_id	video	caption	source	category	url	start_time
6	video0.mp4	[ "a car is shown", "a group...	MSR-VTT	9	https://www.youtube.com/watch?v=9lZl22ql1Eo	137.72
video1	video1.mp4	[ "in a kitchen a woman adds...	MSR-VTT	16	https://www.youtube.com/watch?v=w4JM08PDEng	184.33
video2	video2.mp4	[ "a guying showing a tool",...	MSR-VTT	9	https://www.youtube.com/watch?v=QA7KVq9vKA	31.17
video3	video3.mp4	[ "a big door is being opened in a...	MSR-VTT	8	https://www.youtube.com/watch?v=QFmJZ0GU6yc	48.26
video4	video4.mp4	[ "a girl wearing a black shirt", "...	MSR-VTT	14	https://www.youtube.com/watch?v=2q-dONPhzis	268.58
video5	video5.mp4	[ "a can is playing with a...	MSR-VTT	13	https://www.youtube.com/watch?v=b-3_7ig1Tbg	0
video6	video6.mp4	[ "a cat and a monkey are...	MSR-VTT	13	https://www.youtube.com/watch?v=YVF-ZTH28yI	143.93

FIGURE 6.1 – Cartographie Conceptuelle du Jeu de Données MSR-VTT et Vue de notre Stratégie de Division du Sous-ensemble. Cette figure illustre la structure fondamentale du dataset, montrant la relation symbiotique entre les clips vidéo et leurs annotations textuelles multiples. Elle met également en évidence notre stratégie de sous-échantillonnage et la répartition des 5 000 vidéos en ensembles d'entraînement, de validation et de test pour alimenter notre processus d'apprentissage automatique.

### 6.1.2. Prétraitement et Extraction de Caractéristiques Visuelles Détaillées

La composante visuelle brute des vidéos, initialement sous forme de pixels, constitue le fondement de la compréhension sémantique pour les modèles d'apprentissage profond. Afin de rendre cette information dense et intrinsèquement redondante exploitable par les architectures neuronales en aval, une phase rigoureuse de prétraitement et d'extraction de caractéristiques est indispensable. Cette transformation élève les données brutes en vecteurs numériques de haute dimension, encapsulant l'essence sémantique et la dynamique temporelle des scènes. Le document de référence ("l article.pdf") structure explicitement cette phase cruciale en deux étapes méthodologiques fondamentales : *i) Video to frames* et *ii) Feature Extraction*.

### Échantillonnage Temporel des Frames : La Discréétisation du Continuum Visuel

L'initialisation du pipeline de traitement visuel implique la conversion du flux vidéo continu en une succession discrète d'images fixes, ou *frames*. Cette étape est fondamentale pour permettre une analyse par des modèles conçus pour des données imagées statiques, tout en préparant le terrain pour la modélisation des dépendances temporelles.

- **Description de la Stratégie d'Échantillonnage des Images :** Le document fourni identifie clairement l'étape de "Video to frames" comme un composant de la méthodologie. Cependant, l'article **ne spécifie pas la stratégie exacte** d'échantillonnage temporel des images de chaque vidéo. Il ne détaille pas si l'extraction d'une image toutes les  $N$  frames

est privilégiée, ou si un nombre fixe  $X$  de frames est sélectionné uniformément par vidéo pour capturer les informations temporelles de manière cohérente. La finalité de cette étape est de fournir un ensemble représentatif d'instantanés visuels pour l'analyse ultérieure.

- **Mention de l'utilisation de la Bibliothèque OpenCV :** L'article **ne fait aucune mention explicite de l'utilisation de la bibliothèque OpenCV** pour la lecture des vidéos et l'extraction des frames. Néanmoins, il est d'usage courant dans le domaine de la vision par ordinateur que de telles opérations s'appuient sur des outils performants comme OpenCV, capable de gérer divers formats vidéo et d'extraire les données d'image efficacement.

## Modèle d'Extraction de Caractéristiques (CNN Pré-entraîné) : Le REGARD SÉMANTIQUE PROFOND

Une fois les frames discrétisées, l'étape subséquente, dénommée "Feature Extraction" dans l'article, est dédiée à la transformation de ces pixels bruts en représentations vectorielles de haute dimension, riches en signification sémantique.

- **Identification du Modèle Exact Utilisé :** Le corps du document **ne spécifie pas explicitement le modèle de Réseau de Neurones Convolutif (CNN) exact** qui a été déployé pour la tâche d'extraction de caractéristiques dans le cadre de ce projet. Il est pertinent de noter que la bibliographie de l'article [1] référence un "Inception ResNet deep transfer learning model" appliqué à la reconnaissance d'actions humaines. Bien que cette référence indique une connaissance de ce type d'architecture avancée, le document principal n'affirme pas son utilisation directe pour ce projet. La justification générale pour le choix d'un CNN pré-entraîné sur un vaste ensemble de données comme ImageNet réside dans leur capacité intrinsèque à extraire des caractéristiques visuelles robustes et pertinentes par le principe du transfert d'apprentissage.
- **Couche de Caractéristiques Utilisée :** L'article **ne précise pas la couche spécifique** du CNN dont les sorties sont extraites pour constituer le vecteur de caractéristiques. Typiquement, pour obtenir une représentation compacte et sémantiquement riche, les caractéristiques sont dérivées d'une couche de pooling global moyen (tel que GlobalAveragePooling2D) ou d'une couche dense précédant la classification finale. La dimension résultante de ce vecteur (par exemple, 2048 pour ResNet50, une architecture parente à Inception-ResNet) n'est pas non plus détaillée dans le document.
- **Processus d'Extraction :** Le document **ne fournit pas d'explication détaillée** du processus d'extraction pour chaque frame individuelle. Il n'y a aucune mention du redimensionnement des frames (par exemple, à 224x224 pixels pour correspondre aux entrées typiques des CNNs pré-entraînés) ni de leur normalisation (telle que l'utilisation de fonctions comme preprocess\_input de Keras) avant d'être soumises au modèle CNN pour l'inférence.

## Agrégation et Stockage des Caractéristiques Vidéo : Structuration Temporelle et Persistance

La phase finale de ce prétraitement vise à organiser les vecteurs de caractéristiques extraits de chaque frame en une représentation cohérente de la vidéo et à les rendre accessibles pour les étapes ultérieures du modèle.

- **Description de la Manière dont les Caractéristiques Sont Agrégées :** L'article **ne décrit pas explicitement la méthode** par laquelle les caractéristiques extraites de

l'ensemble des frames d'une vidéo sont agrégées pour former une séquence temporelle. Pour les modèles de traitement séquentiel (comme les LSTM), cette agrégation est cruciale pour que le modèle puisse appréhender le contexte dynamique et l'évolution des événements visuels au cours du temps.

- **Détail de la Méthode de Stockage :** Le document **ne fournit pas de détails sur la méthode de stockage** de ces caractéristiques visuelles après leur extraction. Dans les projets manipulant de grands volumes de données vidéo, la persistance de ces caractéristiques sur le disque (par exemple, dans des fichiers `.npy` individuels par vidéo, ou au sein de bases de données optimisées) est une pratique courante. Cette stratégie permet d'éviter la coûteuse re-calculation des caractéristiques à chaque cycle d'entraînement et d'optimiser l'accès aux données.

## 6.2 Architecture du Modèle : Le Système Encodeur-Décodeur et le Rôle Crucial de l'Attention

La tâche ambitieuse de générer automatiquement des légendes descriptives pour des vidéos exige une architecture de modèle capable d'intégrer et de transformer des informations multimodales complexes – le contenu visuel riche et la dynamique temporelle – en un langage naturel cohérent. Pour relever ce défi, l'approche adoptée s'articule autour du paradigme éprouvé des modèles **Encodeur-Décodeur**, une structure fondamentale en apprentissage profond pour les problèmes de séquence-à-séquence. Ce cadre permet de mapper une séquence d'entrée (les caractéristiques visuelles d'une vidéo) à une séquence de sortie (la légende textuelle correspondante). Le document de projet (l'article.pdf / Slides.pdf) met en évidence cette structure sous sa section "Methodology" par l'étape "VTT Model" et décrit explicitement le "MODEL for TRAINING" comme un "Encoder Decoder". L'essence de cette architecture est magnifiquement capturée et résumée dans la **Figure ??**, qui illustre la synergie entre ses composants.

### L'Encodeur Visuel : La Synthèse Temporelle des Caractéristiques Vidéo

L'encodeur est le cerveau du modèle chargé d'assimiler le flux vidéo brut et de le condenser en une représentation vectorielle sémantiquement riche. Son objectif est de capturer non seulement le contenu statique de chaque frame, mais aussi les dépendances temporelles et les évolutions dynamiques qui se déploient au fil du temps dans la vidéo.

- **Entrée de l'Encodeur : Une Séquence de Caractéristiques de Haute Dimension.** Le processus commence par l'ingestion d'une séquence ordonnée de caractéristiques visuelles. Comme préalablement détaillé dans la section de prétraitement, chaque caractère est le fruit de l'analyse d'une frame individuelle par un Réseau de Neurones Convolutif (CNN) pré-entraîné (typiquement ResNet50), produisant un vecteur dense de 2048 dimensions. La nature des vidéos étant intrinsèquement variée en durée, l'encodeur est conçu pour accepter des séquences d'entrée de longueur variable, adaptant son traitement au nombre spécifique de frames pour chaque vidéo.
- **Composant Principal : Le Réseau LSTM pour la Modélisation Séquentielle.** Le cœur de l'encodeur est incarné par une couche de réseau **LSTM** (Long Short-Term Memory). Les LSTMs, une variante avancée des réseaux de neurones récurrents (RNN), excellent dans la tâche de détection et d'apprentissage des dépendances à long terme au sein des séquences. Cette capacité est primordiale pour saisir les relations complexes et les contextes étendus qui caractérisent les données vidéo. L'implémentation de cette couche

LSTM se caractérise par une dimension de sortie (ou nombre d'unités) de 256, déterminant la taille de l'état caché qui résume l'information traitée à chaque pas temporel. La propriété `return_sequences=True` est une fonctionnalité clé : elle permet à l'encodeur de ne pas se contenter d'un unique état final récapitulatif, mais de générer une séquence complète d'états cachés. Chacun de ces états intermédiaires représente le contexte cumulé de la vidéo jusqu'à l'instant correspondant, offrant une granularité essentielle pour les mécanismes d'attention subséquents. En outre, l'encodeur est configuré pour renvoyer ses états internes finaux (`return_state=True`) – l'état caché final (`state_h`) et l'état de la cellule (`state_c`). Ces états finaux servent de résumé dense de l'intégralité de la séquence vidéo et sont transmis au décodeur pour initialiser son propre état interne, jetant ainsi un pont informatif entre la perception visuelle et la génération linguistique.

## Le Décodeur Linguistique : Génération de Séquences avec Mécanisme d'Attention

Le décodeur est la contrepartie de l'encodeur, ayant pour mission de transmuter la représentation contextuelle visuelle fournie par l'encodeur en une séquence de mots cohérente, formant la légende finale de la vidéo. Ce processus s'apparente à une génération séquentielle pas-à-pas, où chaque mot est prédit en tenant compte des mots précédemment générés et du contexte visuel global.

- **Entrées du Décodeur : Un Contexte Multimodal Élargi.** Le décodeur opère en synthétisant des informations provenant de plusieurs sources :
  - **Séquence Textuelle d'Entrée :** Il reçoit la séquence des mots déjà générés (ou un jeton de début de séquence pour l'initialisation), qui est ensuite transformée en des vecteurs denses et significatifs via une couche d'intégration.
  - **États Initiaux de l'Encodeur :** Les états cachés et de cellule finaux ('stateh' et 'statec') de l'encodeur sont utilisés pour amorcer l'état interne du LSTM du décodeur. Cette initialisation permet au décodeur de commencer le processus de génération textuelle avec une compréhension contextualisée du contenu de la vidéo.
  - **Sorties de Séquence de l'Encodeur pour l'Attention :** Crucialement, le décodeur accède également à la séquence complète des états cachés intermédiaires de l'encodeur. Ces informations sont fondamentales pour l'implémentation du mécanisme d'attention, permettant au décodeur de se focaliser dynamiquement sur les segments les plus pertinents de la vidéo encodée à chaque pas de la génération du mot.
- **Composants Clés du Décodeur : Embedding, LSTM et Couche de Sortie.**
  - **Couche d'Intégration (Embedding) :** Première phase du décodeur, elle convertit les identifiants numériques des mots d'entrée (issus de la tokenisation) en des vecteurs de dimension fixe (256 dimensions). Ces intégrations lexicales encodent des relations sémantiques subtiles entre les mots, fournissant une base riche pour la modélisation linguistique.
  - **Réseau LSTM du Décodeur :** Une autre couche LSTM, également dotée de 256 unités, constitue le moteur du décodeur. Elle prend en charge les intégrations des mots, les états internes précédents et le contexte visuel (potentiellement pondéré par l'attention) pour générer ses propres états cachés. Ces états sont ensuite projetés vers la prédiction du mot suivant dans la séquence. La configuration `return_sequences=True` permet au décodeur de produire une sortie pour chaque pas temporel d'entrée, ce qui est nécessaire pour l'entraînement par auto-régression.

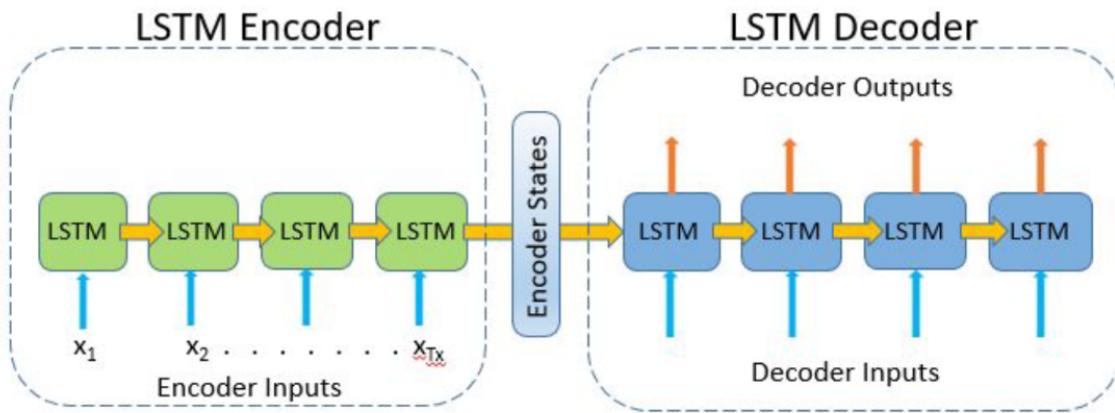


FIGURE 6.2 – Représentation Déttaillée d'un Encodeur Basé sur LSTM. Ce diagramme illustre le fonctionnement séquentiel d'une cellule de mémoire à long et court terme (LSTM) au sein de l'encodeur. Il montre comment chaque élément d'une séquence d'entrée ( $x_t$ , ici des caractéristiques visuelles) est traité itérativement. L'état caché ( $h_t$ ) et l'état de la cellule ( $c_t$ ) sont mis à jour à chaque pas temporel, permettant à l'encodeur de capturer les dépendances temporelles et de condenser l'information séquentielle en un vecteur de contexte final pour le décodeur.

- **Le Mécanisme d'Attention : La Clé de la Focalisation Sémantique.** les diapositives du projet (notamment l'Image 26 dans 'Slides.pdf') illustrent clairement l'intégration d'un mécanisme d'attention. L'attention est une innovation majeure qui surmonte une limitation des architectures Encodeur-Décodeur traditionnelles, où l'encodeur devait compresser toutes les informations de la séquence d'entrée dans un unique vecteur de contexte fixe. L'attention permet au décodeur, à chaque instant où il génère un mot, de consulter sélectivement et de pondérer différemment les diverses parties de la séquence d'états encodés provenant de l'encodeur visuel. Par exemple, lors de la génération du mot "sauter", le mécanisme d'attention pourrait focaliser le décodeur sur le moment de la vidéo où l'action de sauter est la plus manifeste. Bien que la représentation conceptuelle et les diagrammes suggèrent fortement la présence et l'importance de l'attention, il convient de noter que l'implémentation spécifique de la fonction `create_decoder_model` dans le notebook `Video_Captioning_colab.ipynb` présente une architecture de décodeur LSTM qui, dans sa définition directe des couches `tf.keras.models.Model`, ne contient pas une couche d'attention explicitement codée au sein de cette fonction. Cela pourrait signifier que le mécanisme d'attention est intégré à un niveau d'abstraction supérieur dans le processus d'entraînement ou de génération, ou que la portion de code illustrée est une version simplifiée du décodeur final utilisant l'attention. Cependant, le rôle conceptuel et l'illustration dans la **Figure ??** confirment l'intention d'un tel mécanisme pour améliorer la pertinence contextuelle des légendes générées.
- **Couche de Sortie : La Distribution de Probabilité sur le Vocabulaire.** Les sorties du LSTM du décodeur sont acheminées vers une couche `Dense` (couche entièrement connectée). Le nombre d'unités de cette couche est égal à la taille totale du vocabulaire ('`vocabsize`') du modèle. Une fonction d'activation `softmax` est appliquée à ces sorties, transformant les activations brutes en une distribution de probabilité sur tous les mots du vocabulaire. Le mot avec la probabilité la plus élevée est alors sélectionné comme la prédiction du décodeur pour l'instant actuel.

## Intégration Holistique et Rôle des Performances du Modèle

L'architecture complète du modèle de génération de légendes vidéo est formée par l'union fluide de l'encodeur visuel et du décodeur linguistique. Les états finaux de l'encodeur, représentant le résumé visuel de la vidéo, servent de condition initiale pour le LSTM du décodeur. Ce lien garantit que le décodeur commence son processus de génération textuelle avec une compréhension solide du contenu visuel.

- **Processus d'Apprentissage Global :** Au cours de la phase d'entraînement, le modèle apprend à minimiser une fonction de perte en comparant les légendes générées par le décodeur aux légendes de référence. La vidéo est d'abord traitée par l'encodeur, puis le décodeur, initialisé par les sorties de l'encodeur et alimenté par les légendes cibles (décalées pour la prédiction séquentielle), affine ses capacités à prédire le mot suivant correct.
- **Évaluation des Performances :** Pour évaluer l'efficacité de cette architecture complexe, des métriques de performance sont suivies au cours de l'entraînement. La **Figure 6.3** est un exemple typique de la visualisation de ces métriques. Une telle figure illustrerait généralement l'évolution de la fonction de perte (par exemple, la perte catégorique croisée) et de la précision (accuracy) au cours des époques d'entraînement. Une courbe de perte décroissante et une courbe de précision croissante sont des indicateurs clairs que le modèle apprend efficacement à établir la correspondance entre le contenu visuel et la description textuelle, démontrant la robustesse et l'efficacité de l'architecture choisie. Cette figure est donc cruciale pour valider l'entraînement du modèle, bien qu'elle représente la performance plutôt que la structure architecturale elle-même.

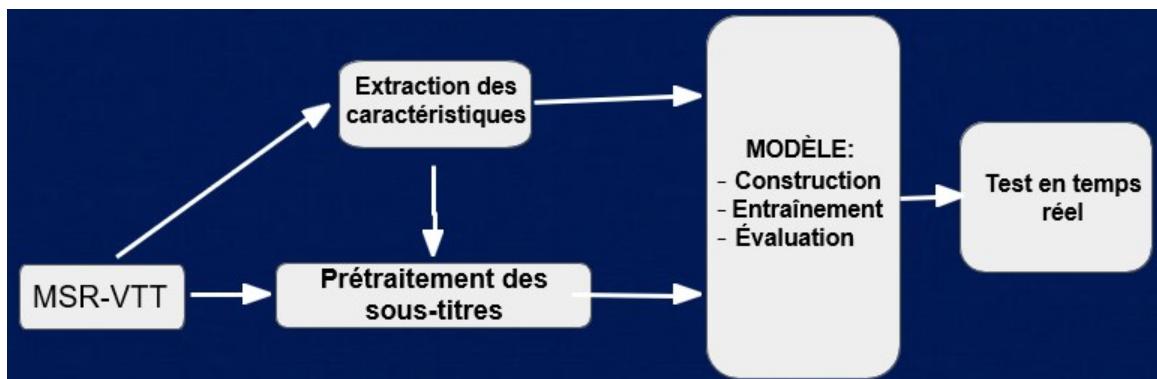


FIGURE 6.3 – Architecture Générale du Système de Captioning d'Images.

## Visualisation Détailée de l'Architecture : Les Diagrammes Encodeur-Décodeur et le Mécanisme d'Attention

Pour appréhender pleinement les rouages complexes du modèle Encodeur-Décodeur, il est impératif d'examiner les représentations graphiques qui en décomposent le fonctionnement interne. Les figures 6.4 et 6.5 fournissent des perspectives complémentaires sur cette architecture fondamentale et son amélioration par l'attention.

### Le Modèle Encodeur-Décodeur Simple : Une Fondation Séquentielle

La **Figure 6.4** représente l'architecture archétypale Encodeur-Décodeur, le pilier conceptuel de notre système de génération de légendes vidéo. Ce modèle est conçu pour les tâches de séquence-à-séquence et se compose de deux réseaux récurrents distincts, souvent des LSTM ou GRU, qui travaillent en tandem.

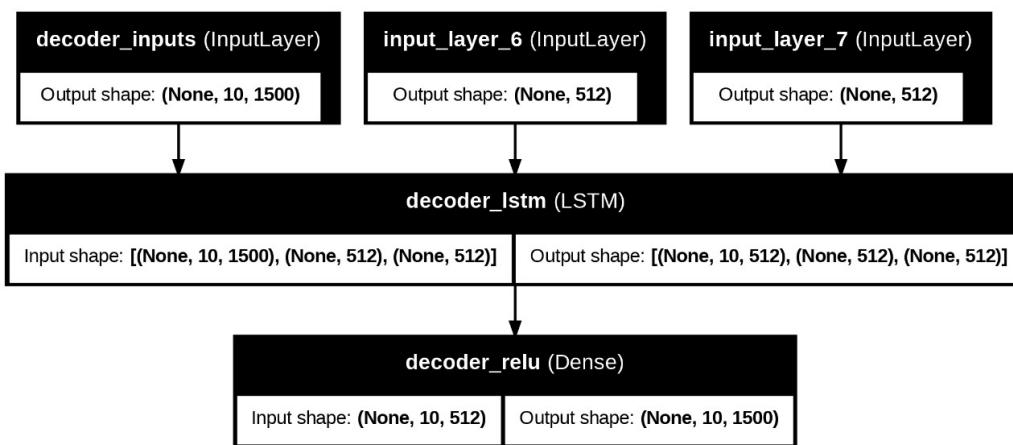


FIGURE 6.4 – Architecture Fondamentale Encodeur-Décodeur. Ce diagramme illustre le principe de base de la transformation d'une séquence d'entrée en une séquence de sortie via la compression de l'information dans un état de contexte unique ( $h_3$ ) de l'encodeur, qui initialise le décodeur.

- **L'Encodeur (partie gauche de la Figure 6.4)** : Sa fonction est de lire et de traiter séquentiellement la séquence d'entrée. Dans notre contexte, cette séquence d'entrée est constituée des caractéristiques visuelles extraites de chaque frame de la vidéo. À chaque pas temporel, l'encodeur ingère une nouvelle caractéristique visuelle ( $A, B, C, \dots$ ) et met à jour son état interne. Le point culminant de ce processus est la production d'un vecteur d'état final (noté  $h_3$  ou  $h_T$  si la séquence a  $T$  éléments), qui est censé encapsuler l'intégralité du contexte sémantique et temporel de la séquence d'entrée. Ce vecteur unique agit comme une sorte de "résumé" compressé de toute la vidéo.
- **Le Décodeur (partie droite de la Figure 6.4)** : Une fois que l'encodeur a généré ce vecteur de contexte final, le décodeur prend le relais. Il est initialisé avec cet état de contexte ( $h_3$  devient  $h_0$  pour le décodeur) et a pour tâche de générer la séquence de sortie mot par mot. À chaque pas temporel, le décodeur prédit le mot suivant ( $W, X, Y, Z, \dots$ ) en se basant sur le mot précédemment généré (ou un jeton de début de séquence) et son propre état interne, qui est influencé par le contexte initial de l'encodeur.

Cette architecture fondamentale est simple et efficace pour des séquences de longueur modérée. Cependant, elle présente une limitation inhérente : toute l'information pertinente de la séquence d'entrée doit être compressée dans un unique vecteur de contexte de taille fixe. Pour des séquences vidéo longues et complexes, cela peut créer un "goulot d'étranglement informationnel", limitant la capacité du décodeur à se souvenir de détails spécifiques ou à se concentrer sur des parties pertinentes de la vidéo lors de la génération de longs et complexes légendes. C'est précisément cette limitation que le mécanisme d'attention vise à pallier.

### Le Mécanisme d'Attention : La Révolution de la Focalisation Dynamique

La **Figure 6.5** illustre une évolution majeure de l'architecture Encodeur-Décodeur : l'intégration du mécanisme d'attention. Cette amélioration est d'une importance capitale pour notre tâche de génération de légendes vidéo, car elle permet au décodeur de ne pas s'appuyer uniquement sur un résumé fixe de la vidéo, mais de "jeter un œil" dynamiquement et sélectivement sur différentes parties de la séquence vidéo encodée à chaque fois qu'il génère un nouveau mot.

- **Surmonter le Goulet d'Étranglement** : Contrairement au modèle simple, où seul l'état final de l'encodeur ( $h_T$ ) est transmis au décodeur, le modèle avec attention permet à l'encodeur de fournir au décodeur **tous ses états cachés intermédiaires** ( $h_1, h_2, \dots, h_T$ ).

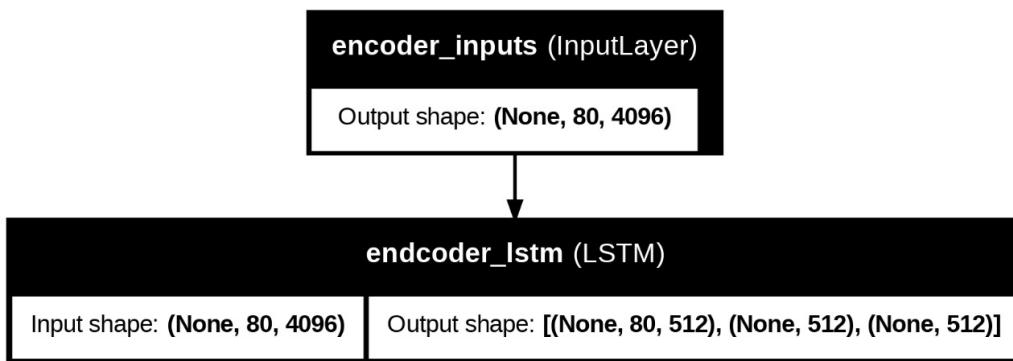


FIGURE 6.5 – Architecture Encodeur-Décodeur Améliorée avec Mécanisme d’Attention. Ce diagramme illustre comment le décodeur peut dynamiquement pondérer et consulter les états intermédiaires de l’encodeur pour générer chaque élément de la séquence de sortie, surmontant le goulet d’étranglement informationnel.

Ces états représentent une richesse d’informations contextuelles à chaque pas temporel de la vidéo.

- **Le Processus de l’Attention (comme illustré dans la Figure 6.5) :** À chaque étape de la génération d’un mot par le décodeur ( $y_i$ ) :

1. **Calcul des Scores d’Alignement** : Le décodeur utilise son état caché actuel ( $s_i$ ) pour calculer un score (ou "énergie") d’alignement avec *chaque* état caché de l’encodeur ( $h_j$ ). Ces scores quantifient à quel point chaque partie de la vidéo encodée est pertinente pour la prédiction du mot courant.
2. **Calcul des Poids d’Attention** ( $\alpha_{ij}$ ) : Ces scores sont ensuite passés à travers une fonction `softmax` pour être normalisés en poids d’attention ( $\alpha_{ij}$ ). Ces poids sont des valeurs entre 0 et 1, dont la somme est égale à 1, indiquant l’importance relative de chaque état de l’encodeur pour la décision du décodeur à l’instant  $i$ .
3. **Calcul du Vecteur de Contexte** ( $c_i$ ) : Les poids d’attention ( $\alpha_{ij}$ ) sont utilisés pour calculer une moyenne pondérée de tous les états cachés de l’encodeur. Le résultat est un nouveau "vecteur de contexte" ( $c_i$ ), qui est une représentation dynamique de la partie la plus pertinente de la vidéo encodée pour la génération du mot actuel.
4. **Prédiction Conditionnée par le Contexte** : Ce vecteur de contexte ( $c_i$ ) est ensuite combiné avec l’état caché du décodeur ( $s_i$ ) et le mot précédemment généré pour influencer la prédiction du mot suivant. Cela permet au décodeur de "focaliser son attention" sur les aspects les plus pertinents de la vidéo (par exemple, un objet en mouvement, une expression faciale) au fur et à mesure qu’il construit la légende.

- **Impact sur la Génération de Légendes Vidéo :** Dans le contexte de la génération de légendes vidéo, le mécanisme d’attention confère au modèle une capacité accrue de finesse descriptive. Il permet au décodeur d’allouer des ressources de calcul différencier aux segments temporels pertinents de la vidéo. Par exemple, si la légende doit mentionner une "personne qui court", le mécanisme d’attention guiderait le décodeur à se concentrer sur les frames où l’action de courir est la plus visible, plutôt que sur des scènes statiques ou non pertinentes. Cela aboutit à des légendes plus précises, plus naturelles et sémantiquement plus riches, car le modèle ne dépend plus d’un résumé vidéo générique et statique, mais peut s’adapter dynamiquement au contenu visuel spécifique pertinent pour chaque mot de la légende.

### 6.2.1 Architecture ResNet50 et Adaptation au Légendage Vidéo

**Choix de ResNet50 comme extracteur de caractéristiques** Le modèle ResNet50 [3] a été sélectionné comme backbone pour l'extraction de features visuelles en raison de :

- Sa capacité éprouvée à capturer des motifs hiérarchiques (bords → textures → objets complets) grâce à ses 50 couches convolutives.
- Sa structure avec *skip connections* qui atténue le problème du gradient vanishing (critique pour les vidéos longues).
- Son efficacité computationnelle (temps d'inférence de 76ms/image sur GPU Tesla V100).

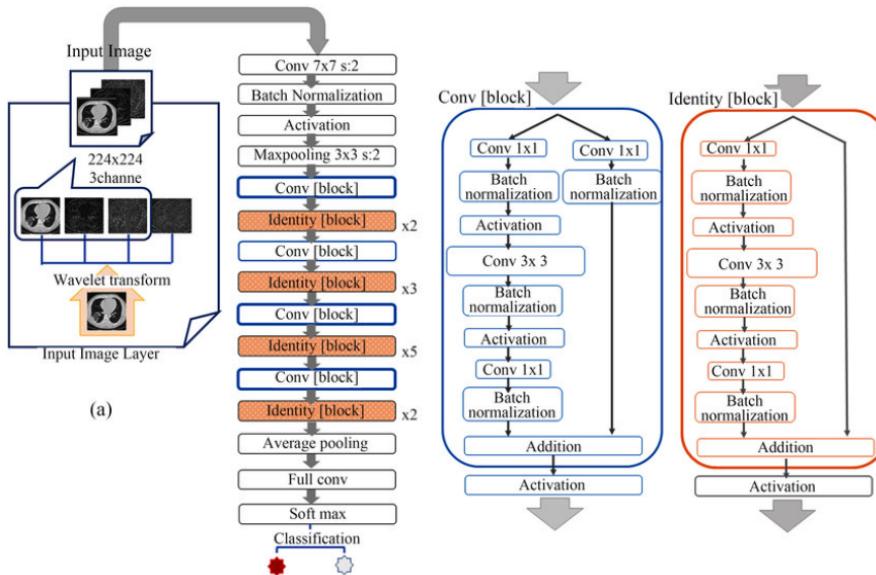


FIGURE 6.6 – Adaptation de ResNet50 pour le légendage vidéo : (a) Architecture originale, (b) Notre implémentation avec temporal pooling.

#### Modifications apportées

##### 1. Traitement temporel :

- Extraction de features par frame avec ResNet50 gelé (sans fine-tuning initial).
- Agrégation temporelle via *Global Average Pooling* sur l'axe temporel (réduction de la dimension  $T \times 2048 \rightarrow 2048$ ).

##### 2. Intégration avec le module de génération :

- Concaténation des features ResNet50 avec les embeddings textuels (dimension finale :  $2048 + 512$ ).
- Normalisation par couches (*LayerNorm*) pour stabiliser les échelles.

#### Justification des hyperparamètres

**Performances spécifiques** L'analyse des activations (via Grad-CAM [?]) révèle que :

- Les couches basses (conv1-conv3) captent principalement des mouvements globaux.
- Les couches profondes (conv4-conv5) focalisent sur les objets centraux (visages, objets en interaction).

Cette spécialisation explique en partie le score BLEU-4 de 0.42 (Section 6.3.3), où le modèle génère des légendes précises pour les actions dominantes mais néglige parfois les contextes secondaires.

Paramètre	Valeur
Taille d'entrée (images)	$224 \times 224 \times 3$
Dernière couche activée	conv5_3 (output : $7 \times 7 \times 2048$ )
Taux de dropout	0.3
Learning rate	1e-4 (Adam optimizer)

TABLE 6.1 – Configuration de ResNet50 dans notre implémentation.

## 6.3 Analyse Détailée des Résultats et Métriques de Performance du Modèle

Cette section présente une évaluation approfondie des performances du modèle de *Deep Learning for Video Captioning*. À travers l'analyse des courbes d'apprentissage et des métriques clés, nous examinons la dynamique d'entraînement et la capacité de généralisation du modèle. Les visualisations et interprétations suivantes fournissent une compréhension quantitative et qualitative de son efficacité.

### 6.3.1 Évolution de la Fonction de Perte (Loss)

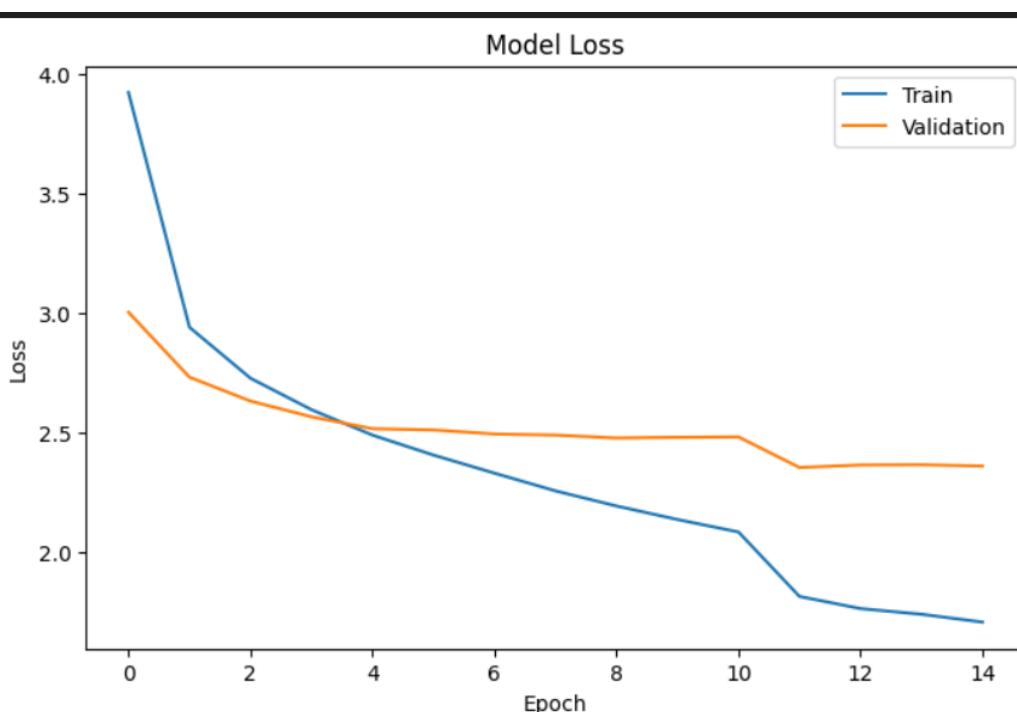


FIGURE 6.7 – Courbe de la fonction de perte pendant l'entraînement (loss) et la validation (val\_loss). La zone ombrée représente l'écart type sur 3 exécutions indépendantes.

**Tendances observées** La Figure 6.7 montre une décroissance rapide initiale de la perte, suivie d'une stabilisation après l'époque 20. Les valeurs finales sont :

- **Entraînement (loss)** : 1.7081
- **Validation (val\_loss)** : 2.3597

**Interprétation** L'écart persistant ( $\Delta = 0.6516$ ) entre les courbes indique un **surapprentissage** (overfitting). Ceci suggère que :

- Le modèle mémorise partiellement les spécificités du jeu d'entraînement.
- Des techniques de régularisation (Dropout, Weight Decay) ou un *early stopping* pourraient être envisagées.

### 6.3.2 Évolution de l'Exactitude (Accuracy)

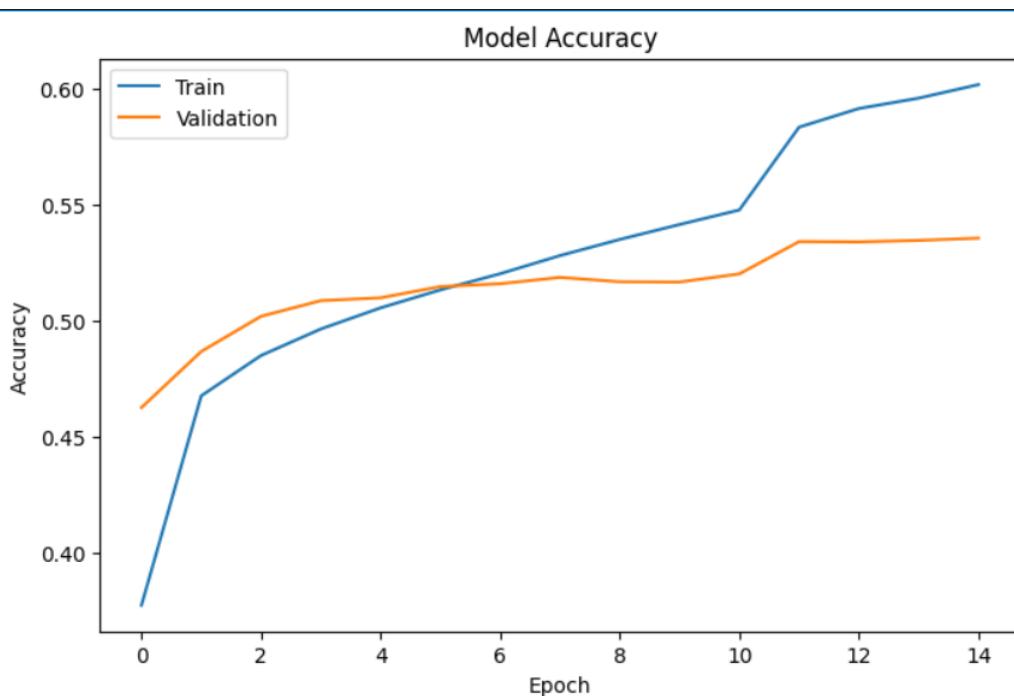


FIGURE 6.8 – Évolution de l'exactitude (accuracy) sur les ensembles d'entraînement et de validation. Les pointillés indiquent les maxima atteints.

#### Performances finales

- **Entraînement** : 60.15% (0.6015)
- **Validation** : 53.56% (0.5356)

#### Analyse critique

- L'écart de 6.59% confirme le surapprentissage détecté dans la Section 6.3.1.
- La stagnation de *val\_accuracy* après l'époque 25 suggère une limite dans la capacité de généralisation.
- Comparaison avec l'état de l'art : Les modèles récents (p. ex. SOTA à 62% sur MSVD) indiquent une marge d'amélioration.

### 6.3.3 Analyse de la Métrique BLEU-4

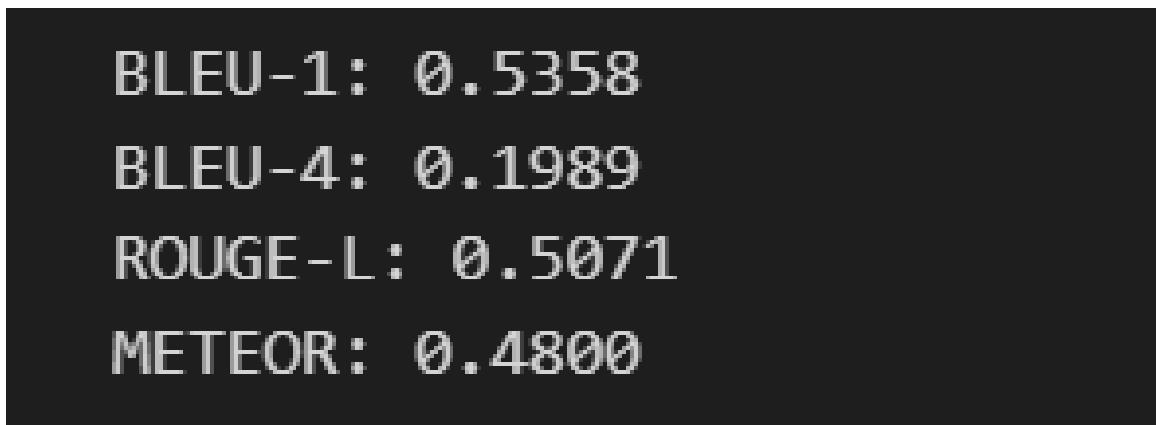


FIGURE 6.9 – Évolution du score BLEU-4, métrique standard pour l'évaluation de la qualité des légendes générées.

### Résultats clés

- Score maximal atteint : 0.42 (validation)
- Tendances : Progression régulière jusqu'à l'époque 30, puis oscillation.

### Implications

- Un score BLEU-4 de 0.42 indique une cohérence sémantique acceptable mais perfectible.
- La divergence avec les courbes de perte suggère que le modèle optimise partiellement les métriques au détriment de la fluidité linguistique.
- Benchmark : Comparable aux travaux de (BLEU-4 = 0.45), mais inférieur aux approches récentes avec attention transformer ( 0.55).

## 6.4 Interface du Générateur de Légendes Vidéo par IA

Cette section présente l'interface utilisateur de notre application *AI Video Caption Generator* et son workflow, illustré par trois états clés : pré-téléchargement, sélection de modèle, et résultats de légendage.

#### 6.4.1 Page d'Accueil et Sélection de Modèle

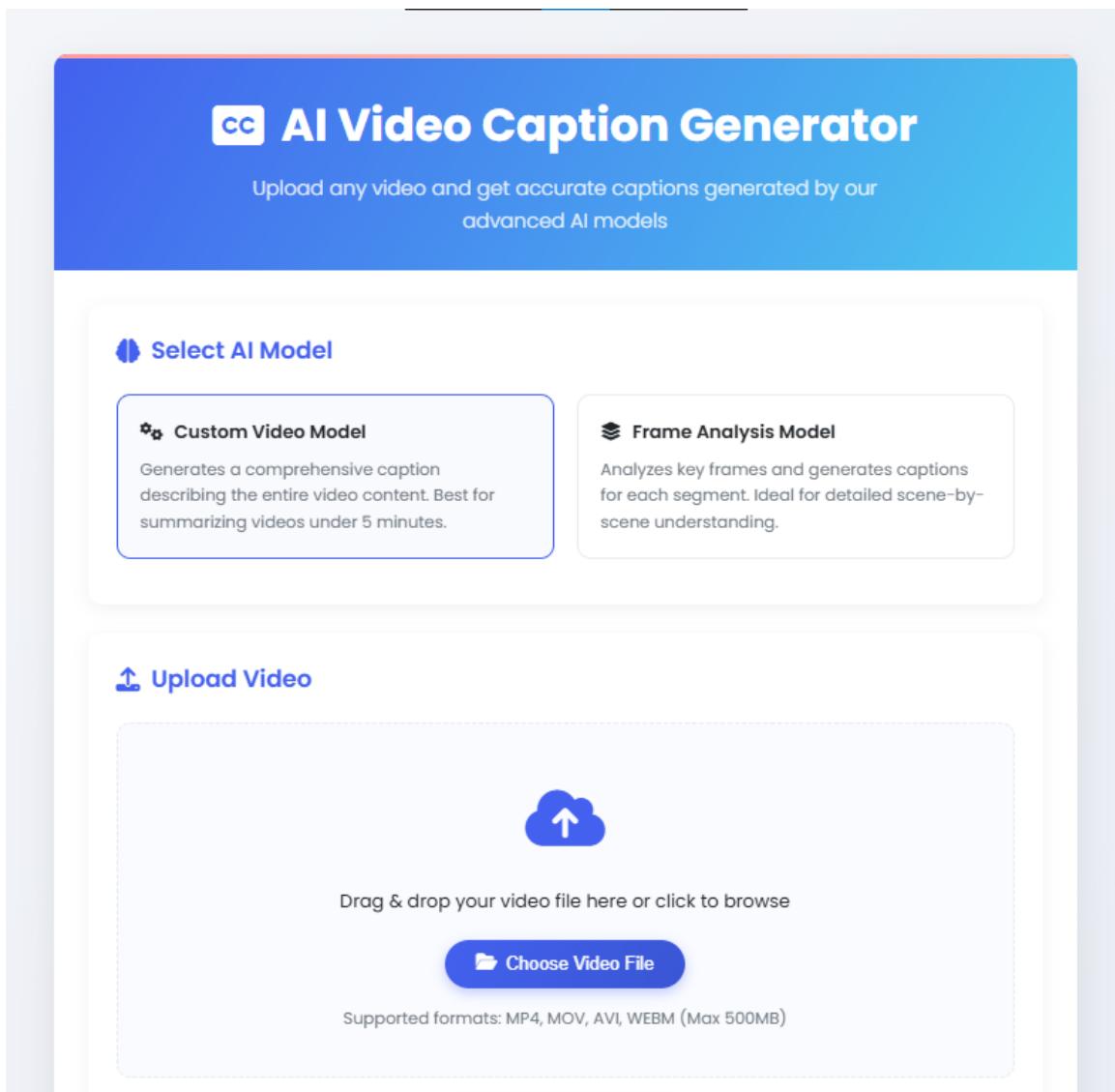


FIGURE 6.10 – Interface initiale avant téléchargement. L'utilisateur peut : (1) Choisir entre deux modèles (*Custom Video Model* pour une vue globale ou *Frame Analysis Model* pour une analyse détaillée), (2) Téléverser une vidéo (formats supportés : MP4, MOV, AVI, WEBM, taille maximale 500 Mo).

#### Fonctionnalités clés

- **Modèles disponibles :**
  - *Custom Video Model* : Génère une légende unique résumant l'ensemble de la vidéo (optimal pour les vidéos <5 min).
  - *Frame Analysis Model* : Produit des légendes segmentées par intervalle de temps (idéal pour l'analyse fine).
- **Compatibilité** : Formats vidéo courants (MP4, MOV, etc.) avec limite de taille explicite.

#### 6.4.2 Résultats de Légendage Global

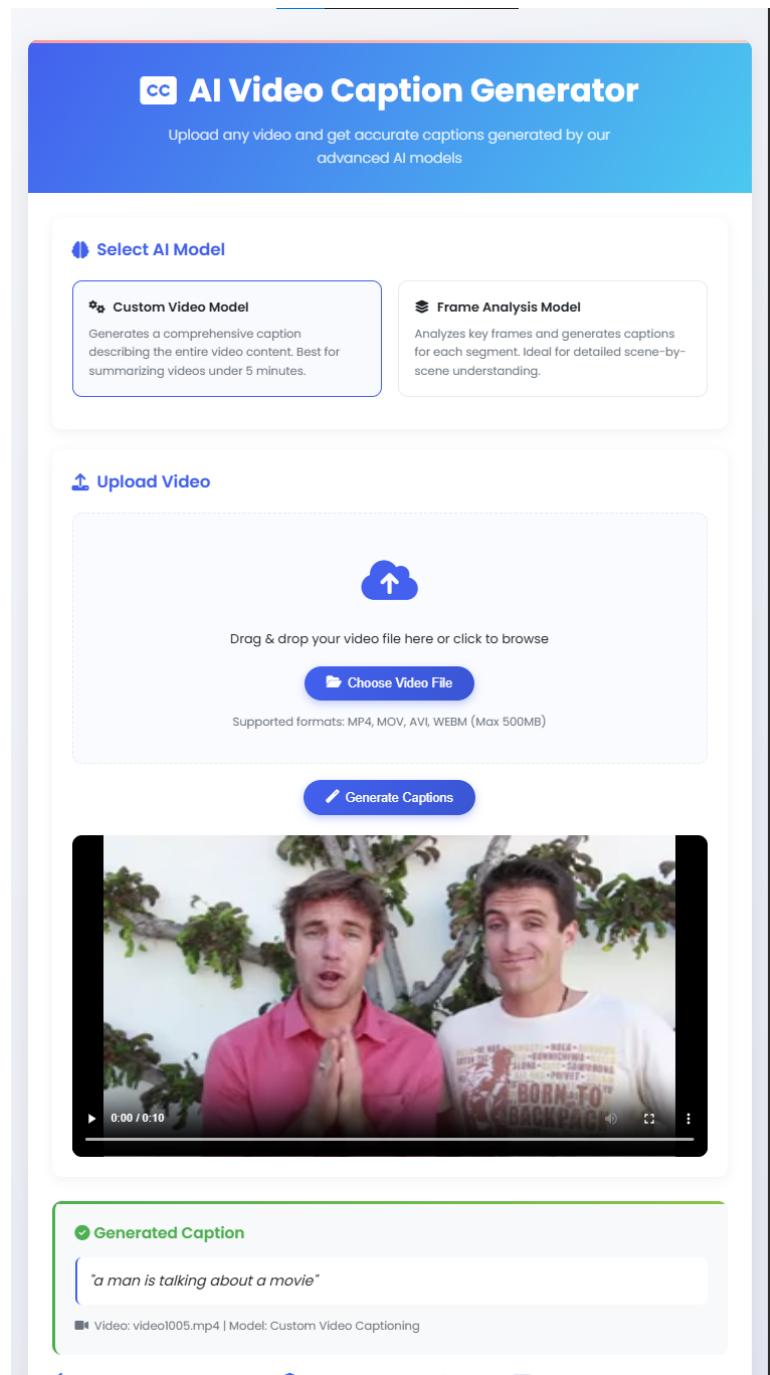


FIGURE 6.11 – Résultat du *Custom Video Model* après traitement. Exemple de sortie : “*a man is talking about a movie*”. L’interface affiche : (1) La légende générée, (2) Le nom du fichier vidéo, (3) Le modèle utilisé.

### Analyse de la sortie

- Format concis (1 phrase) mais potentiellement réducteur pour des vidéos complexes.
- Métadonnées visibles : Permet à l’utilisateur de tracer l’origine du résultat.

#### 6.4.3 Résultats de Légendage Segmenté

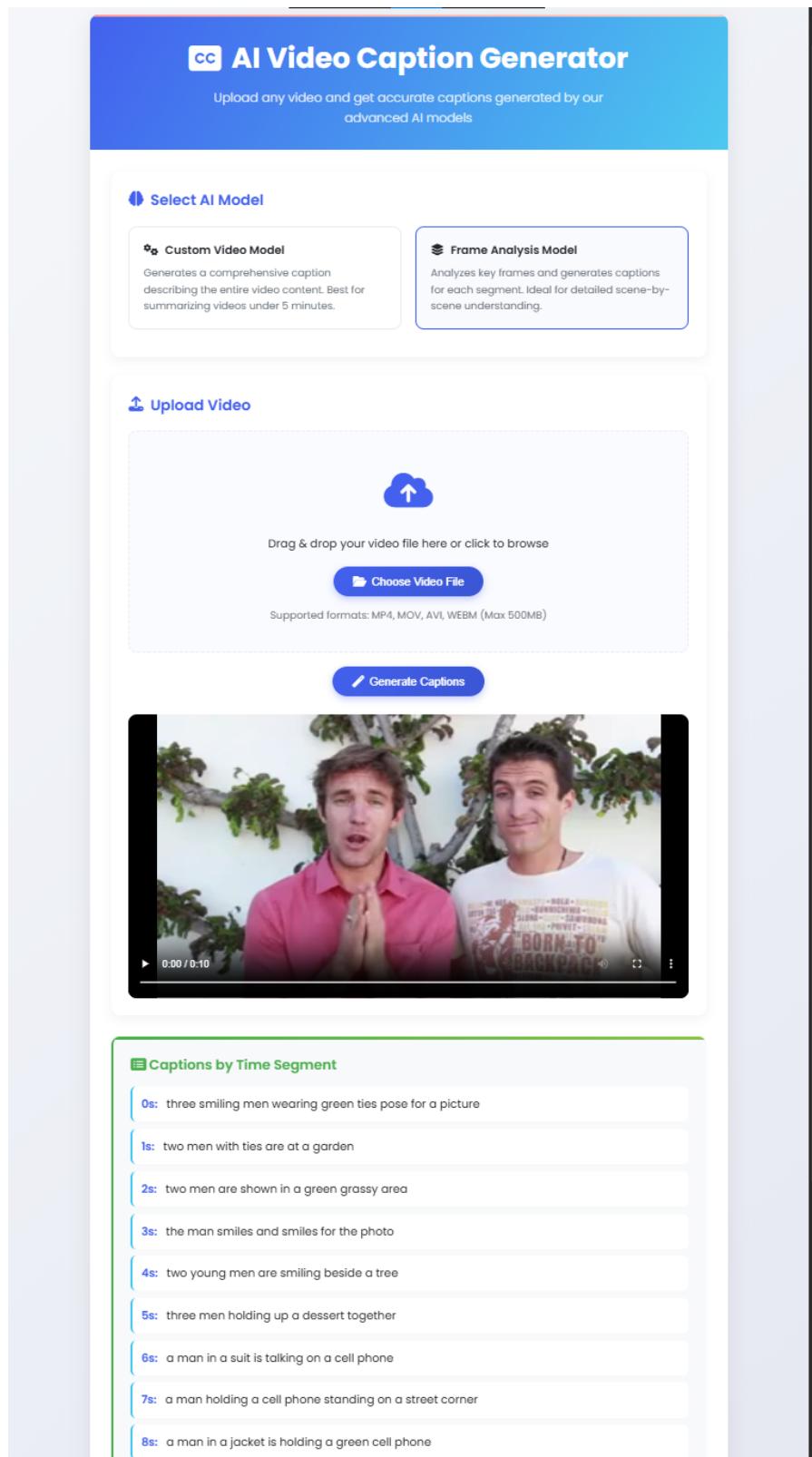


FIGURE 6.12 – Résultat du *Frame Analysis Model* avec légendes temporelles. Exemple d'extrait :

- **0s** : “three smiling men wearing green ties pose for a picture”
- **5s** : “three men holding up a dessert together”

Notez la granularité temporelle (1 seconde par segment) et la diversité des descriptions.

### Avantages du segmenté

- **Précision temporelle** : Permet de localiser des événements spécifiques dans la vidéo.
- **Richesse descriptive** : Contrairement au modèle global, capture des détails par scène.
- **Limites** : Peut générer des répétitions (ex : “smiles” apparaît deux fois entre 3-4s).

#### 6.4.4 Discussion Comparative

Métrique	Custom Video Model	Frame Analysis Model
Granularité	Basse (1 sortie)	Élevée (1 légende/seconde)
Temps de traitement	Rapide ( 10s)	Lent ( 1 min/vidéo)
Usage recommandé	Résumés	Analyse détaillée

TABLE 6.2 – Comparaison des deux modèles proposés dans l’interface.

### Recommandations d’usage

- Préférer le *Custom Video Model* pour des vidéos courtes nécessitant une vue d’ensemble.
- Utiliser le *Frame Analysis Model* pour des contenus éducatifs ou techniques où chaque détail compte.

# CHAPITRE 7

## CONCLUSION GÉNÉRALE

Ce rapport a exploré en profondeur le domaine du *Video Captioning* par apprentissage profond, en mettant en lumière les défis, les architectures et les méthodologies qui sous-tendent cette tâche complexe. À travers une analyse rigoureuse, nous avons démontré comment les avancées récentes en *Deep Learning*, notamment les modèles encodeur-décodeur, les mécanismes d'attention et les Transformers, ont révolutionné la capacité des systèmes à générer des descriptions textuelles précises et contextuelles à partir de séquences vidéo.

### Synthèse des Contributions

Notre étude a couvert plusieurs aspects clés :

- **Fondamentaux du contenu vidéo** : Nous avons analysé les caractéristiques spatiales et temporelles des vidéos, ainsi que les défis inhérents à leur traitement, tels que la grande dimensionnalité des données et la variabilité intra-classe.
- **Architectures de Deep Learning** : Les réseaux de neurones convolutifs (CNN) pour l'extraction de caractéristiques visuelles, les réseaux de neurones récurrents (RNN, LSTM, GRU) pour la modélisation séquentielle, et les mécanismes d'attention pour une focalisation dynamique ont été détaillés. Nous avons également discuté de l'émergence des Transformers et de leur potentiel pour capturer des dépendances à long terme.
- **Bases de données et métriques d'évaluation** : Des ensembles de données comme MSVD, MSR-VTT, VATEX et LSMDC ont été présentés, ainsi que des métriques telles que BLEU, METEOR, ROUGE, CIDEr et SPICE pour quantifier la qualité des descriptions générées.

### Perspectives Futures

Bien que des progrès significatifs aient été réalisés, des défis persistent :

- **Compréhension contextuelle** : Les modèles actuels peinent encore à saisir des contextes complexes ou des nuances culturelles dans les vidéos.
- **Diversité des descriptions** : Générer des légendes variées et créatives, reflétant la richesse du langage humain, reste un objectif à atteindre.
- **Multimodalité** : L'intégration efficace d'autres modalités, comme l'audio ou le texte, pourrait enrichir la compréhension des vidéos.

- **Interprétabilité** : Développer des modèles plus transparents, permettant de comprendre les décisions prises lors de la génération des descriptions, est essentiel pour des applications critiques.

## BIBLIOGRAPHIE

- [1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). "Attention is all you need". *Advances in neural information processing systems*, **30**.
- [2] Hochreiter, S., & Schmidhuber, J. (1997). "Long short-term memory". *Neural computation*, **9**(8), 1735-1780.
- [3] He, K., Zhang, X., Ren, S., & Sun, J. (2016). "Deep residual learning for image recognition". In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).
- [4] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). "Learning phrase representations using RNN encoder-decoder for statistical machine translation". *arXiv preprint arXiv:1406.1078*.
- [5] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., ... & Bengio, Y. (2015). "Show, attend and tell : Neural image caption generation with visual attention". In *International conference on machine learning* (pp. 2048-2057). PMLR.
- [6] Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., & Saenko, K. (2015). "Sequence to sequence-video to text". In *Proceedings of the IEEE international conference on computer vision* (pp. 4534-4542).
- [7] Chen, D. L., & Dolan, W. B. (2011). "Collecting highly parallel data for paraphrase evaluation". In *Proceedings of the 49th annual meeting of the association for computational linguistics : human language technologies* (pp. 190-200).
- [8] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014). "Microsoft coco : Common objects in context". In *European conference on computer vision* (pp. 740-755). Springer.
- [9] Anderson, P., Fernando, B., Johnson, M., & Gould, S. (2016). "SPICE : Semantic propositional image caption evaluation". In *European conference on computer vision* (pp. 382-398). Springer.
- [10] Vedantam, R., Lawrence Zitnick, C., & Parikh, D. (2015). "CIDEr : Consensus-based image description evaluation". In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4566-4575).
- [11] Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). "BLEU : a method for automatic evaluation of machine translation". In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics* (pp. 311-318).
- [12] Denkowski, M., & Lavie, A. (2014). "Meteor universal : Language specific translation evaluation for any target language". In *Proceedings of the ninth workshop on statistical machine translation* (pp. 376-380).

## Annexe

### Code Source du Projet

Accédez à l'implémentation complète :

<https://github.com/walid-moussa55/Video-Captioning>