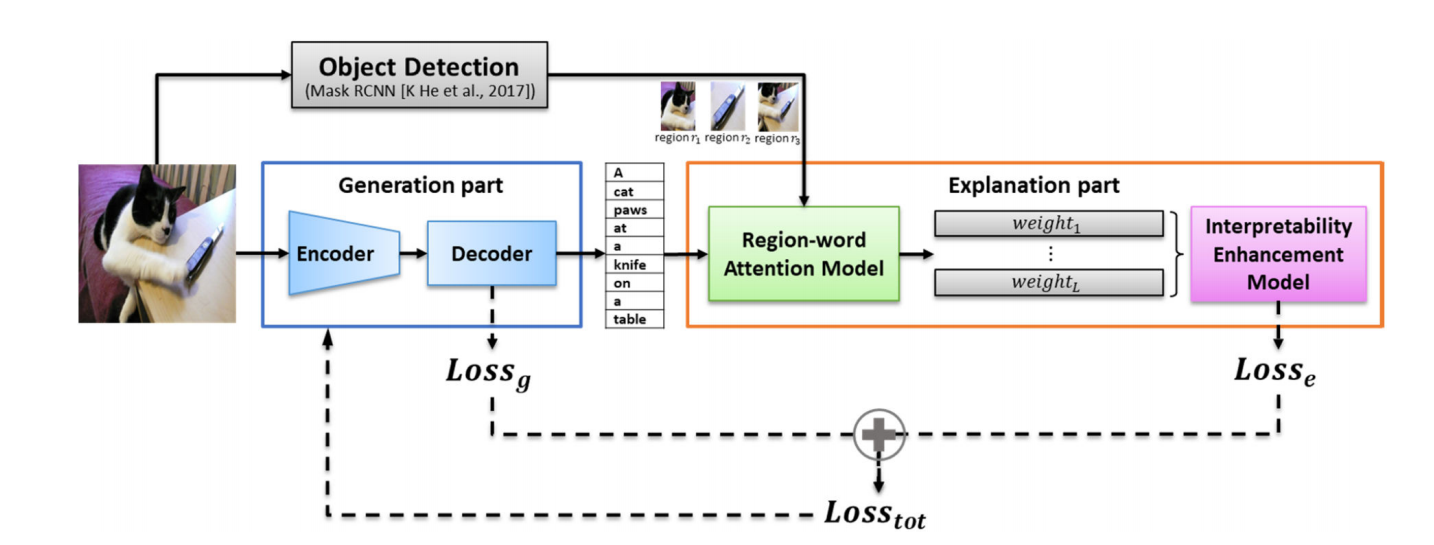
EXplainable AI (XAI) approach to image captioning

Différents types de modèles existent pour le sous-titrage d’image :

* Encoder-decoder model : Un encodeur extrait un vecteur de caractéristique depuis une image d’entrée via un CNN. Puis un décodeur génère une phrase en utilisant ce vecteur via un RNN.
* Object detection : Phrases plus détaillées en utilisant des parties spécifiques d’une image. Une couche de localisation va proposer des régions depuis une image d’entrée puis en extrait les caractéristiques. De celles-ci, un modèle langage RNN est entrainé et génère des légendes pour chaque région de l’image.
* Attention mechanism : L’encodeur divise une image donnée en une grille de régions et génère un ensemble de vecteurs de caractéristiques pour chaque région. Ensuite, ces vecteurs sont introduits dans un modèle d'attention, qui attribue des poids aux vecteurs de caractéristiques. Enfin, le décodeur convertit ces vecteurs de caractéristiques en vecteurs de contexte en multipliant les poids du modèle d’attention, puis génère une légende en utilisant les vecteurs de contexte

Modèle proposé :



Deux parties : génération et explication.

Partie génération : Génère une légende depuis l’image en utilisant une architecture encoder/decoder. Framework CNN-RNN.

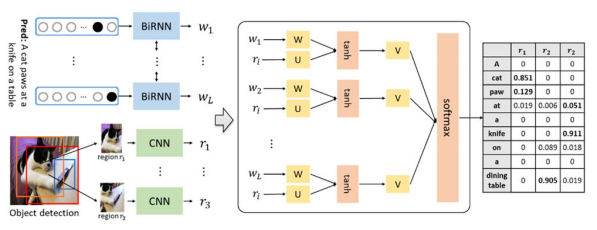
* Partie encoder : VGG-16 model et toutes les images convertis à la même taille pour extraire les image feature vector.
* Partie decoder : LSTM qui génère des mots en utilisant le vecteur de caractéristique de l’image et du word embedding.
* Utilisation d’un negative log likelihood loss function pour optimiser les paramètres entrainable du modèle.

Partie explication : Génère une matrice de poids pour chaque région de l’image et chaque mot de la légende. Cette partie permet d’indiquer à la génération les objets importants à considérer lors de la génération de légende et d’explications.

* Pendant l’entrainement : génère une image-sentence relevance loss qui chiffre si la légende générée représente bien les objets de l’image d’entrée. Les objets sont extraits par un algorithme de détection d’objet. Plus la génération est entrainée, plus le model arrive à générer des légendes considérant les objets de l’image.
* Pendant le testing : génère une matrice de poids pour les différentes régions extraites de l’image d’entrée et pour les mots générés par la partie génération. Chaque poids représente la cohérence dans la pair objet-mot.

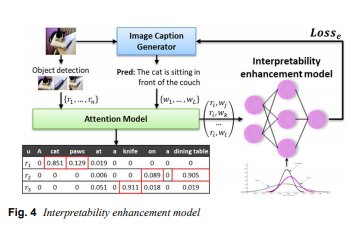
Ces deux parties génèrent deux Loss Values qui affectent les paramètres entrainables de la génération afin que celle-ci considère les informations de chaque région.

La partie explication est composée de deux éléments :

* Region-word attention model : aider la génération en considérant les informations de chaque région. 

La partie explication est pre-entrainée avant la génération. Pour cela, la sous-partie attention est entrainée en premier. Pendant la phase d’entrainement, les entrées du Region-word attention sont toutes les régions extraites des images et toutes les legendes GT des images. (Optimizer des paramètres : Mean-squarred loss function)

* Interpretability enhancement model : Génère l’image-sentence relevance loss.



# Protocole de mise en place contexte de voiture autonome

* 1. **Récupérer un Dataset d’image caméras de voitures.**
  2. **Labelliser le dataset avec des légendes (Entre 1 et 5 légendes)**

**2.1 Pre-processing des données pour maximiser les performances.**

2.1.1 Pour les légendes :

Tout en minuscule.

Pas de phrases de plus de 15 mots.

Filtre des mots trop récurrents comme the this etc...

2.1.2 Pour les images :

Il faut d’abord construire un dataset des régions d’image. Via un détecteur d’objet, garder les régions avec une confiance de plus de 85%. (Detector model : Mask RCNN)

Rejeter les régions de moins de 50x50 pixels.

Puis pre-processing des images entières.

Altération de la taille de toutes les images à une taille fixé (256x256 ?).

Rejet des images avec aucune région de confiance ~85%.

**3.1 Création de la partie Génération.**

3.1.1 Encoder : VGG-16 model pour extraire un vecteur de caractéristique.

3.1.2 Decoder : LSTM (RNN) en utilisant le vecteur de caractéristique et du word embedding.

3.1.3 Optimizer : Negative log likelihood.

**3.2 Création de la partie Explication.**

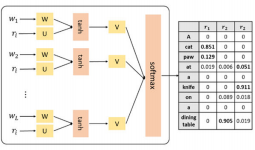
3.2.1 Region-word attention model :

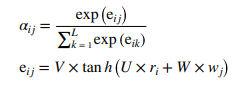
Première entrée : régions, sous-images extraites des images originales.

Deuxième entrée : mots, généré par la génération pendant l’entrainement. (Cela correspond à un « deuxieme entrainement » car le modèle est au préalable pré-entrainé)

Structure du modèle : Feed-forward neural network.

Sortie du modèle : Matrice de poids. Chaque colonne représente un vecteur de poids pour chaque région.

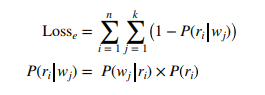




Pré-Entrainement du modèle : Toutes les régions extraites de l’image et tous les mots du GTC. (Ground truth captions). Optimisation par un mean-squared loss function. Les poids de la matrice de sortie sont utilisés comme les predicted value. Les similarités entre le label de chaque région et un mot k des N mots d’entrée est utilisée comme les truths values. Pour obtenir cette valeur de similarité, on utilise un pre-entrained word-embedding, construit avec un dictionnaire des mots et les labels des catégories.

3.2.2 Interpretability enhancement model :

Première entrée : La matrice de poids du Region-word attention model. Il récupère les pairs région-mots avec les plus grandes valeurs.

 Sortie : image-légende relevance loss grâce à la Bayersian inference.

P(ri|wj) : Posterior probability. Comment chaque région et mot sont liés dans la distribution de données.

P(wj|ri) : Likelihood. Calculé statistiquement en utilisant le dataset d’entrainement.

P(ri) : prior probability. Calculé statistiquement en utilisant le dataset d’entrainement.

Entrainement : Cross-entropy loss function avec truth value : produit du likelihood avec la prior probability et predicted value : la posterior probability du modèle.

**3.3 Evalutation du modèle par BLEU et METEOR (algorithm for sentence evaluation)**

