Rapport mini-projet 2

Légender des images

Master 2 Data Science

Auteur :

Mohamed Dhmine

Kenza SKALLI FETTACHI

Moustapha NDIAYE

Encadrant :

M. Badih Ghattas

Année académique 2020–2021

Table des matières

1 Introduction

Le but de ce projet est de construire un modèle qui légende des images données en entrée. On nous a donné une centaine d'images et leurs descriptions. Dans un premier temps on a découpé le problème en trois parties. On a tout d'abord extrait quelques mots-clés de chaque légendes. On a ensuite utilisé un modèle qui associe à chaque image un ou plusieurs mots-clés. A partir de ces mots clés on a généré du texte, qu'on a évalué avec le score BLEU.Le score BLEU permet de mesurer la ressemblance entre les légendes réelles et les légendes générées. Dans un second temps on a directement utilisé un modèle qui décrit les images.

2 Base de données

Il nous a été donné un fichier et deux dossiers. Le premier dossier **Images** contient 255 images, le second dossier **Text_en_paragraphes** contient 77 fichiers qui décrivent chacun plusieurs de ces images, le dernier fichier **desc_OK.txt** contient aussi des descriptions mais beaucoup plus brèves de 198 des images. Ce sont des données géologiques. Le manque de données nous a poussé à récupérer la base de données Flickr8k_Dataset à partir du lien pour tester notre modèle.

3 Allocation de Dirichlet latente (LDA)

L'allocation de Dirichlet latente (latente Dirichlet allocation en anglais) est un modèle probabiliste qui permet d'identifier automatiquement des sujets présents dans un objet texte et d'extraire ainsi les motifs cachés. Il s'agit d'une approche non supervisée utilisée pour trouver et observer des groupes de mots (appelés «sujets ou concepts») dans un ensemble de documents de textes.

Le modèle LDA suppose que les documents sont produits à partir d'un mélange de sujets. Ces sujets génèrent ensuite des mots en fonction de leur distribution de probabilité. Étant donné un ensemble de données de documents, LDA tente de déterminer quels sujets créeraient ces documents.

Considérons une matrice (document-words DW) représentant le corpus (ensemble de document de textes) :

	W1	W2	···	Wn
D1	0	3	···	5
D2	1	4	···	2
D3	2	3	···	6
⋮	⋮	⋮	⋮	⋮
Dm	4	3	···	8

Cette matrice montre que le corpus est constitué de M documents répartis sur N vocabulaires (termes). La valeur de la cellule i,j donne le comptage de la fréquence du terme Wj dans le document Di.
La LDA consiste à convertir cette matrice en deux matrices binaires DT (document topic) et TW (topic words) respectivement :

	T1	T2	···	Wk
D1	0	1	···	1
D2	1	0	···	0
D3	1	1	···	1
⋮	⋮	⋮	⋮	⋮
Dm	1	0	···	0

	W1	W2	···	Wn
T1	0	0	···	1
T2	0	1	···	0
T3	1	0	···	1
⋮	⋮	⋮	⋮	⋮
Tk	1	1	···	0

La matrice DT représente la distribution des sujets ou concepts dans les documents et la matrice TW quand à elle représente la distribution des mots (vocabulaire) dans les sujets.
La LDA cherche à améliorer le modèle de sujet généré en calculant deux probabilités :
- $p_1 = \mathbb{P}(t/d)$ qui est la proportion de mots du document d affectéé au sujet t.
- $p_2 = \mathbb{P}(w/t)$ qui est la probabilité que le sujet t dans le corpus soit assigné au mot w.
On calcule alors le produit p_1 et p_2 qui correspond à la probabilité que le sujet t génère le mot w dans le document d.
Ce processus est répétés un grand nombre de fois jusqu'à ce que l'algorithme converge et on obtient le mélange de sujet présent dans chaque document en comptant chaque représentation d'un sujet (assigné aux mots du document). On obtient les mots associés à chaque sujet en comptant les mots qui y sont associés dans le corpus.

3.1 Prétraitement et nettoyage du texte

La modélisation de sujet textuels nécessite en grande partie une nettoyage en amont des données. C'est une étape cruciale pour obtenir des résultats satisfaisants et représentatifs. Ce prétraitement consiste entre autre à tokeniser les documents, c'est à dire découper chaque phrase du document en une liste de mots et supprimer les caractères inutiles tels que les ponctuations.
Un autre étape du prétraitement est aussi la normalisation des documents en passant par

la lemmatisation, c'est à dire de ne considérer que les racines des mots. Par exemple on peut avoir les mots "drawing" et "drawings" en même temps dans le texte. Etant donné que ces mots sont proches, les distinguer ne ferait qu'accroitre le surapprentissage et ne permettrait pas aux modèles d'exploiter pleinement les données d'apprentissage. Ensuite, il faut supprimer les mots vides, c'est à dire les mots qui apparaissent très fréquemment dans les documents notamment les articles définis et indéfinis.
Une fois ces étapes terminées, nous transformons les documents en une forme vectorisée en calculant la fréquence de chaque terme, constituant ainsi le corpus sur lequel nous allons entraîner le modèle LDA.

3.2 Mesure de cohérence des sujets

Afin de décider du nombre optimal de sujets à extraire à l'aide de LDA, on utilise le score de cohérence pour mesurer la qualité de l'extraction des sujets. Il est défini par :

$$\mathit{CoherenceScore} = \sum\nolimits_{i<j} \text{score}(W_i, W_j)$$

La cohérence mesure la distance relative entre les mots dans un sujet. On distingue deux type de scores :
- Score UCI définit par :

$$\text{Score\_uci} = \log \frac{\mathbb{P}(w_i, w_j)}{\mathbb{P}(w_i)\mathbb{P}(w_j)}$$

où $\mathbb{P}(w_i)$ représente la probabilité de voir w_i dans un document aléatoire, et $\mathbb{P}(w_i, w_j)$ la probabilité de voir les deux w_i et w_j se produisant dans un document aléatoire.
- Score UMass définit par :

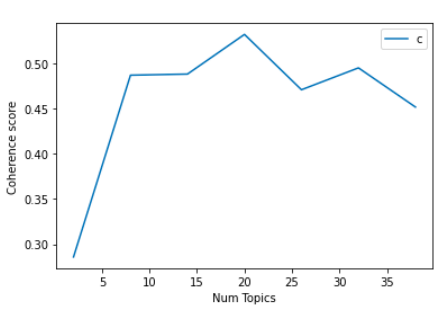
$$\text{Score\_UMass} = \log \frac{\mathbf{D}(w_i, w_j) + 1}{\mathbf{D}(w_i)}$$

où le terme $\mathbf{D}(w_i, w_j)$ du numérateur est le nombre de documents dans lesquels les mots w_i et w_j apparaissent ensemble. On ajoute 1 à ce terme puisque nous prenons le logarithme et nous devons éviter de prendre un logarithme de 0 lorsque les deux mots n'apparaissent jamais ensemble. Le dénominateur $\mathbf{D}(w_i)$ est le nombre de documents $\mathbf{D}$ où apparaît le terme w_i.
Le score est donc plus élevé si w_i et w_j apparaissent beaucoup ensemble dans les documents par rapport à la fréquence que w_i apparaît seul dans les documents. Cela a du sens en tant que mesure de la cohérence du sujet, car si deux mots d'un sujet vont vraiment ensemble, on s'attend à ce qu'ils apparaissent beaucoup ensemble. Le dénominateur ajuste simplement la fréquence des mots dans les documents, de sorte que des mots comme les articles (le, la , $\cdots$) n'obtiennent pas un score artificiellement élevé.

3.3 Résultats

3.3.1 Nombre de sujets

Grace au score de cohérence nous pouvons définir le nombre de nécessiare pour construire le modèle.



Num Topics = 2 has Coherence Value of 0.2858
Num Topics = 8 has Coherence Value of 0.4871
Num Topics = 14 has Coherence Value of 0.4883
Num Topics = 20 has Coherence Value of 0.5322
Num Topics = 26 has Coherence Value of 0.471
Num Topics = 32 has Coherence Value of 0.4952
Num Topics = 38 has Coherence Value of 0.4518

FIGURE 1 – Score de cohérence pour obtenir le nombre de sujet nécessaire.

Ici, le modèle nous suggère de prendre 20 sujets puis qu'on obtient un score maximal lorsque le nombre de sujet est 20.

3.3.2 Extraction des mots clés

Nous avons appliqué le model LDA sur fichier de descriptions des images pour extraire les mots clés à travers les sujets. Pour mieux visualiser ces sujets, on utilise le package pyLDAvis qui est un outil graphique interactif.

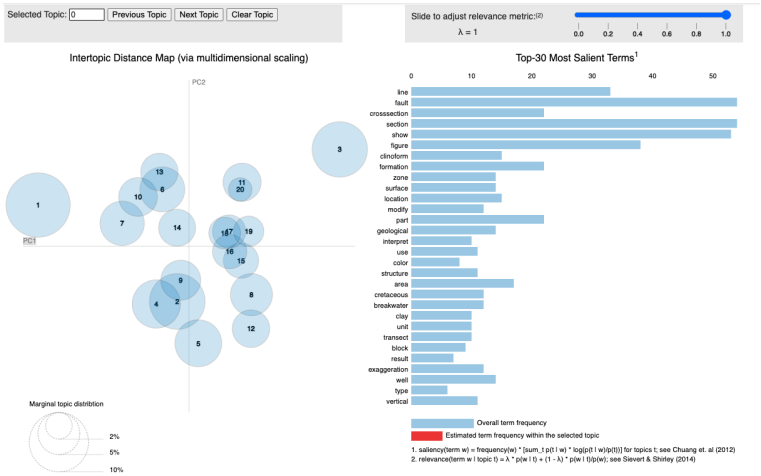


FIGURE 2 – Extraction de mots clés à travers les discriptions des images

Chaque bulle sur le graphique de gauche représente un sujet. Plus la bulle est grande, plus ce sujet est répandu. A droite, nous avons les mots clés les importants pour un sujet donné. Anisi, ces mots-clés seront utilisé pour essayer de générer du texte.

4 Classification multi-label

La classification multi-label est une extension de la classification traditionnelle dans laquelle les classes ne sont pas mutuellement exclusives, chaque individu pouvant appartenir à plusieurs classes simultanément. Ce type de classification est requis par un grand nombre d' applications actuelles telles que la classification d'images. L'ensemble de données que nous utiliserons dans cette partie de classification multi-label se compose de 255 images. Chaque image a sa propre label. Les labels sont définis à partir des données textuelles, et sont extraits au dessus. Pour ce faire, on va utiliser un réseaux de neurones convolutionnels qui a pour objectif de prédire les mots clés des images.
L'architecture CNN que nous utilisons est une version simplifiée de **VGGNet**.

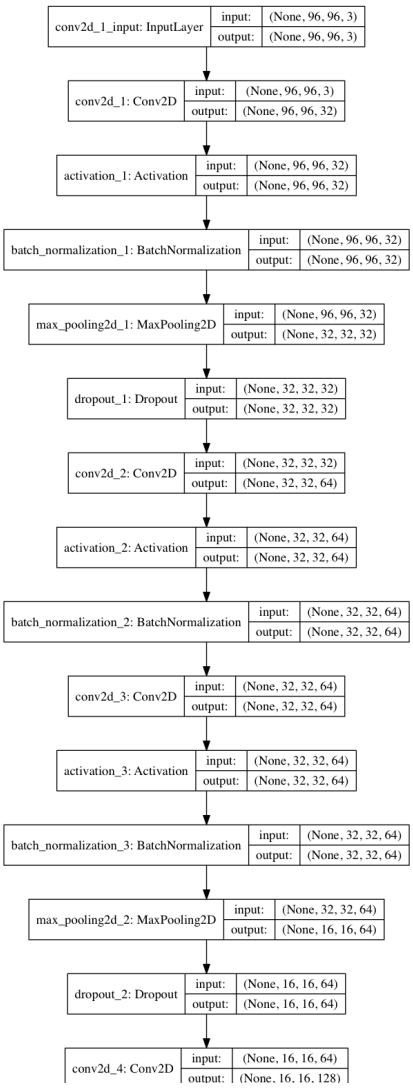


FIGURE 3 – Début du réseau de VGGNet simplifiée

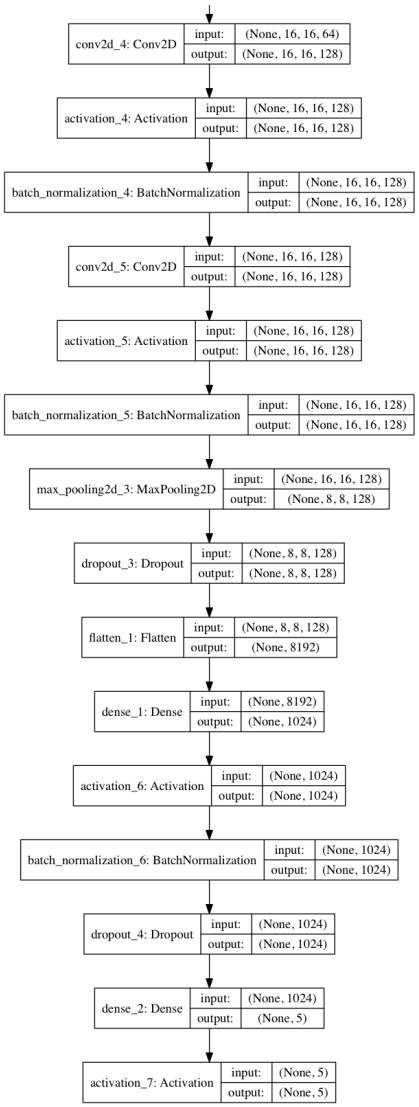


FIGURE 4 – Fin du réseau de VGGNet simplifiée

Les architectures de type VGGNet se caractérisent par :

1. En utilisant seulement 3 × 3 couches convolutives empilées les unes sur les autres à une profondeur croissante
2. Réduction de la taille du volume grâce au max pooling
3. Couches entièrement connectées à la fin du réseau avant softmax classifier

La couche de convolution a 32 filtres avec un 3 x 3 noyau. Nous utilisons *RELU* comme une fonction d'activation suivie de la normalisation des lots.
Notre *POOL* couche utilise un 3 x 3 *POOL* taille pour réduire rapidement les dimensions spatiales 96 x 96 à 32 x 32.

4.1 Résultats

Nous avons formé le réseau pendant 60 itérations, le nombre total d'époques pour lesquelles nous formerons notre réseau (c'est-à-dire combien de fois notre réseau «voit» chaque exemple de formation et en apprend des modèles), réalisant :

— Exactitude de la classification de 96,70% sur l'ensemble d'entraînement ;
— 94,82% de précision de classification sur l'ensemble de test.

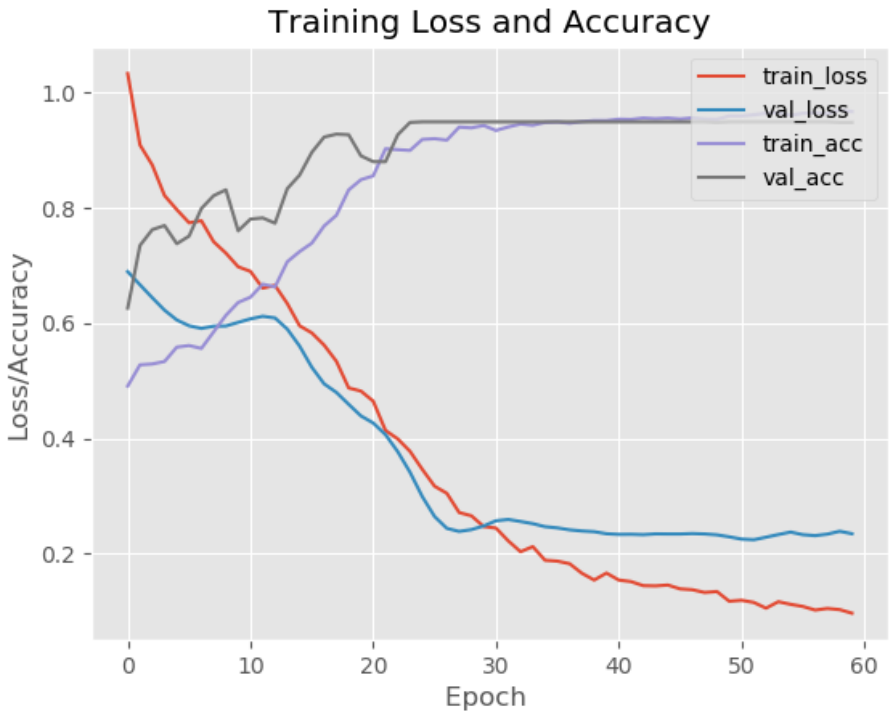


FIGURE 5 – Le graphique d'entraînement

Quelques exemples des resultats obtenus :

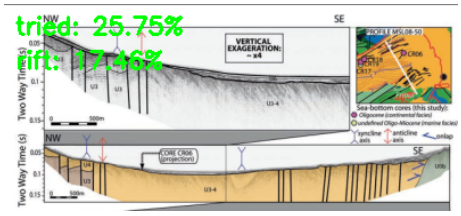


FIGURE 6 – L'exemple 1

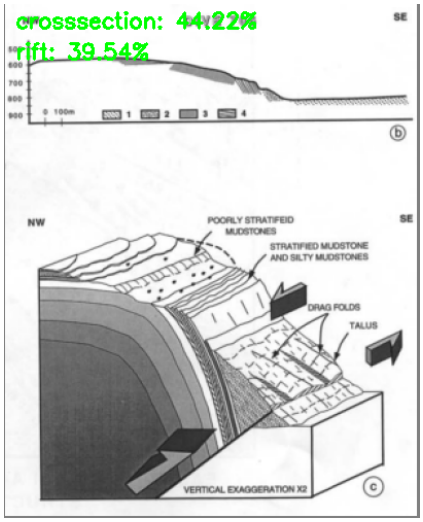


FIGURE 7 – L'exemple 2

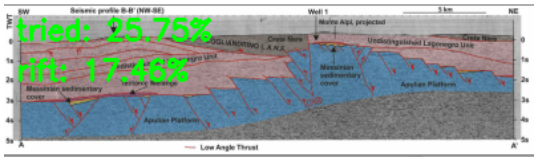


FIGURE 8 – l'exemple 3

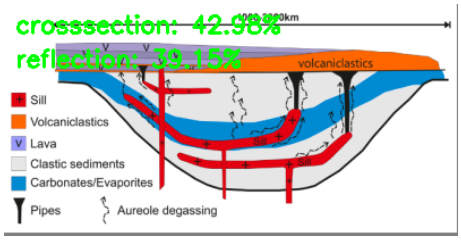


FIGURE 9 – L'exemple 4

Le nombre d'images est insuffisant pour entraîner le reseau de neurones. Tout comme un réseau de neurones classique, il ne peut pas prédire les classes sur lesquelles il n'a jamais été entraîné, le réseau de neurones ne peut pas prédire plusieurs étiquettes de classe pour des combinaisons qu'il n'a jamais vu. La raison de ce comportement est due aux activations des neurones à l'intérieur du réseau.

5 Génération de texte à partir de mots-clés

5.1 Le modèle sequence-to-sequence

On a repris l'architecture encodeur-décodeur du modèle sequence-to-sequence [3]. Ce modèle est très adapté à la traduction automatique. En effet, une séquence de mots, ou de charactères, est donnée en entrée, et une séquence de mots, ou de charactères, est attendue en sortie. Par contre, il est moins adapté à notre problème. La probabilité d'apparition du second mot-clé ne dépend pas du premier. Les mots-clés n'ont pas de relation d'ordre. Je l'ai malgré tout appliqué à notre question, en considérant donc qu'il y a une relation d'ordre entre les mots-clés.

L'encodeur et le décodeur sont des LSTM ; des RNN plus avancés.

5.1.1 Brève explication des RNN et des LSTM

Les réseaux de neurones récurrents traditionnels (RNN) sont très efficaces pour traiter des données séquentielles, comme les phrases, et de prédire à partir d'une séquence de mots le mot suivant. Mais il rencontre des difficultés lorsqu'il est face à de longues phrases : problème d'explosion et de disparition du gradient. Pour calculer l'erreur de prédiction, il a besoin de tous les « hidden state » ou en français les « états cachés ». L'état caché h_k d'une étape k, c'est-à-dire, du mot numéro k comporte l'information des mots précédents.

Les LSTM ont été introduit pour surmonter ces difficultés. En plus de l'état caché, les LSTM ont des « cell state ». Ce sont ces cellules qui vont permettre de retenir seulement l'information essentielle.

5.1.2 Encodeur

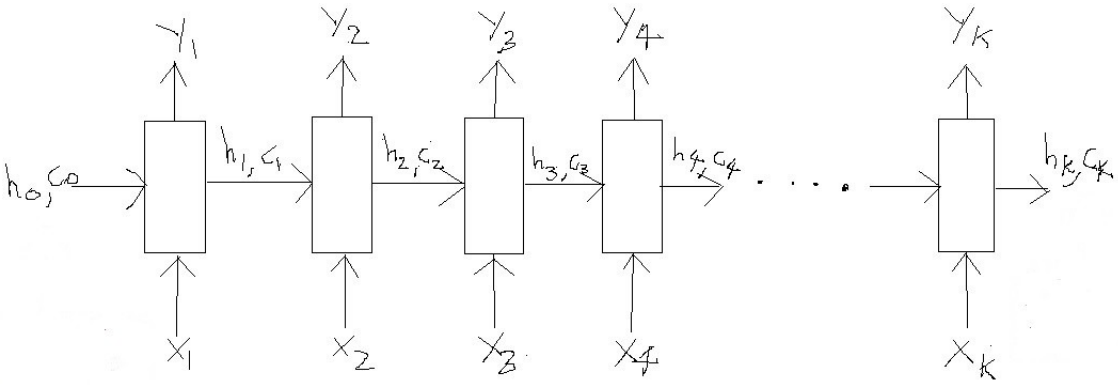


FIGURE 10 – LSTM for language modeling.
Pris d'ici

L'encodeur va résumer la séquence d'entrée , c'est-à-dire les mots clés (qui ne sont en fait pas des séquences), dans ce qui est appelé un vecteur d'états : la concaténation de l'état« hidden state » et de « cell state ».
Oublions notre problème qui est de générer des phrases à partir de mots-clés, et focalisons-nous sur l'exemple de la traduction automatique. L'encodeur va représenter la phrase à traduire dans une autre dimension. Prenons comme phrase à traduire en anglais « La mer est bleue ». Si l'entrée est donnée comme une séquence de mots, le LSTM de l'encodeur va la lire en quatre étapes : $X_1 = \text{La}$, $X_2 = \text{mer}$, $X_3 = \text{est}$, $X_4 = \text{bleue}$.
Si elle est donnée comme une chaine de charactères, le LSTM va la lire en 13 étapes.

Les X_i sur la figure 1 sont donc les entrées à l'étape i.

Les mots ne sont évidemment pas présentés tels quels au LSTM. Ils sont préalablement encodés et ensuite représentés dans une autre dimension par une méthode de "word embedding".

Dans la figure 1 ci dessus, h_i et c_i sont respectivement les "hidden state" et les "cell state" à l'étape i. Ils retiennent ce que le LSTM à appris jusqu'à maintenant. h_2 et c_2 permettent de retenir que le LSTM a lu les mots "la" et "mer" à l'étape 2. h_0 et c_0 sont en général initialisés aléatoirement.

A partir de ces états et de X_i le modèle peut prédire la sortie Y_i à l'étape i. Y_i est un vecteur qui contient la distribution de probabilité de chaque mot du vocabulaire. Le vocabulaire étant tous les mots français présents dans les phrases que l'encodeur reçoit. Ils ne sont pas récupérés.

5.1.3 Décodeur : la phase d'entrainement

Si on reprend l'exemple qui est de traduire la phrase "la mer est bleue", le décodeur va apprendre à renvoyer la phrase "the sea is blue ".

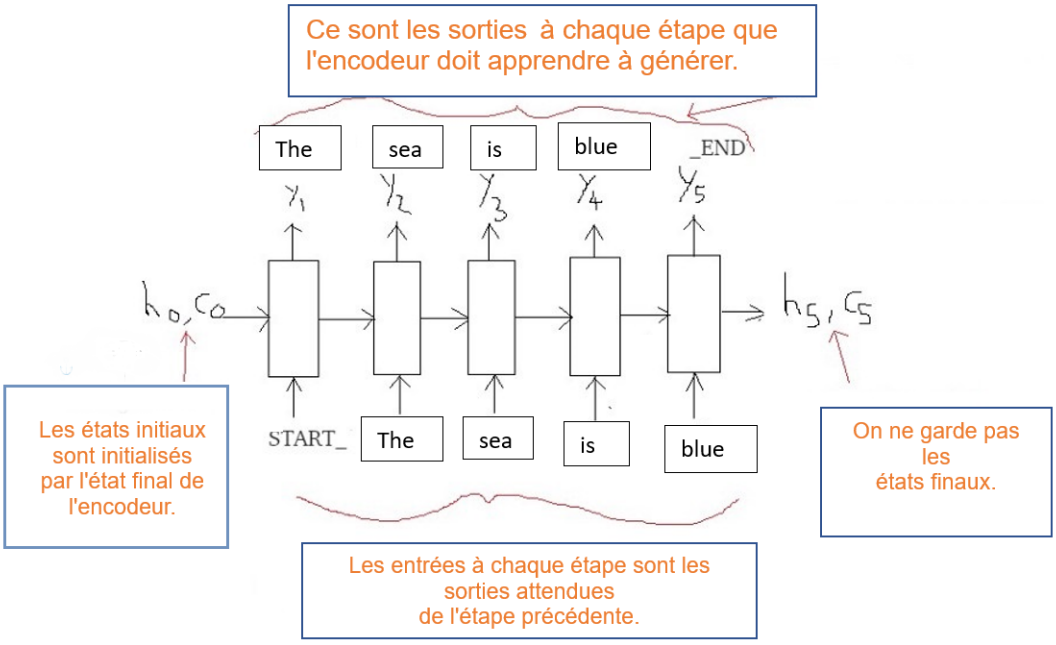


FIGURE 11 – LSTM du décodeur.
Pris d'ici et modifié

La première donnée que le décodeur reçoit est le "mot" "_START". Une fois qu'une phrase est terminée, le décodeur doit apprendre à générer le "mot" "_END". Ce mot "_END" servira aussi lors de la génération des phrases. La séquence que le décodeur doit donc renvoyer est "START_ the sea is blue _END".
Comme expliqué sur la figure 2, les états initiaux h_0 et c_0 sont les états finaux de l'encodeur.

A l'étape $i+1$ l'entrée est la sortie que l'encodeur doit prédire à l'étape i, mais ce ne n'est pas ce qu'il a prédit.

5.1.4 Decodeur : phase d'inférence

Le décodeur va générer les mots un part un. Le LSTM sera appelé plusieurs fois dans une boucle, jusqu'à ce qu'il génère le "mot" "_END". A ce moment-là, la boucle prendra fin.
Les données d'entrées encodées dans le vecteur d'état, sont fournies au décodeur : ils sont résumés dans l'état final de l'encodeur, et sont l'état initial du décodeur. La première entrée du décodeur est l'élément "START_". A partir de cette entrée et des états initiaux, le décodeur va prédire le mot suivant Y_1 et va calculer les états h_1 et c_1. A la seconde étape de la boucle, le décodeur va générer le mot Y_2 à partir du mot Y_1 généré précédement et passé en entrée ; et à partir des états h_1 et c_1 calculés à l'étape une et passé comme état initiaux à l'encodeur de l'étape deux. La boucle prend fin lorsque le mot "_END" est généré.

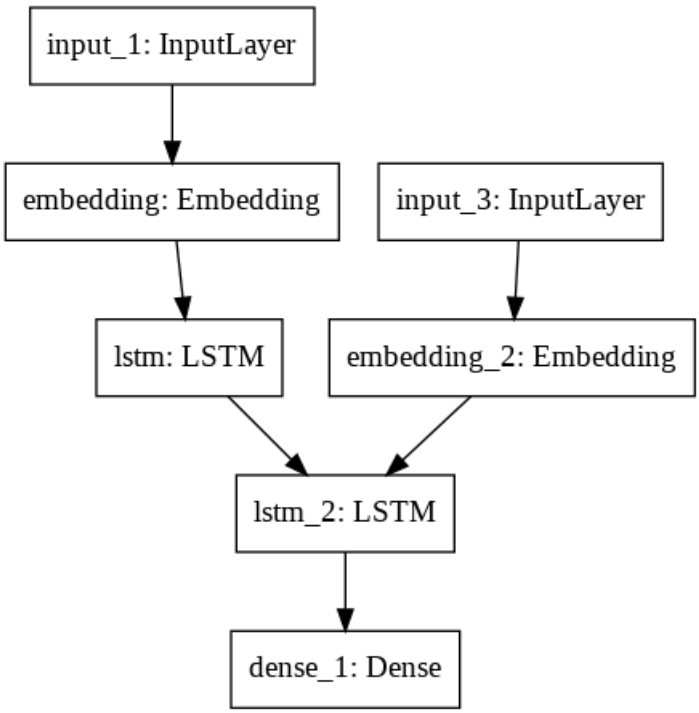


FIGURE 12 – Modèle : encodeur-decodeur, dessiné par keras.utils.plot_model

5.1.5 Word Embedding

Avant d'être donnés aux couches LSTM, les mots sont transformés dans un espace de dimension choisi, par des méthodes de word embedding. On peut ajouter directement une couche de embedding dans le modèle comme sur la figure 12, ou bien le faire à part, par exemple avec la méthode Word2Vec de la bibliothèque gensim.

Dans ce nouvel espace, les mots similaires sont proches, et on obtient de bonnes propriétés, tel que king - man = woman (un exemple fréquement donné).

5.2 Traitement des données

5.2.0.1 séquence de mots On a traité de la même façon les déscriptions géologiques du fichier desc_OK.txt et les descriptions de la base Flickr8k_Dataset. Dans un premier temps j'ai choisi au hasard des mots clés de chaque descriptions. Le nombre de mots-clés dépend de la taille de la description :

— un mot si l'image est décrite par un seul mot.

— la moitié du nombre de mots si elle est décrite par moins de dix mots.

— cinq mots si elle est décrite par plus de dix mots.

A partir de ces mots-clés on obtient un dictionnaire d'entrée. Chaque mot du dictionnaire est désigné par un nombre entier compris entre 1 et le nombre total de mot-clés. La matrice que reçoit l'encodeur est de taille 198x5 où 5 est le nombre maximale de mots-clés pour une description donnée.

Par exemple, on a récupéré qu'un mot-clé de la description 188. Le mot-clé est 'uninterpreted', seul un élément de encoder_input_data prend une valeur différente de 0.

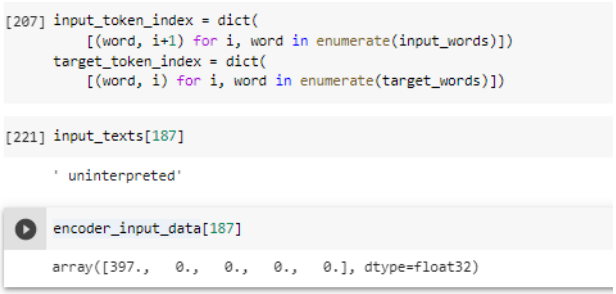


FIGURE 13 – Données en entrées de l'encodeur

On traite de la même façon les données cibles : on récupère tous les mots présents dans chacune des descriptions, et chacun de ces mots sera désigné par un entier. Ces mots forment un dictionnaire des cibles. Alors que l'encodeur reçoit qu'une matrice encoder_input_data, le décodeur en reçoit deux : decoder_input_data et decoder_output_data.

decoder_input_data est définie de la même façon que encoder_inut_data. Elle est de dimension 198x166 où 166 correspond à la taille de la plus longue description.

decoder_output_data est un tenseur de taille 198x166x1851 qui contient les distributions de probabilités sur les mots y compris _END mais (_START est exclu) pour chaque séquence et à chaque pas de la séquence. Par exemple decoder_output_data[0,1,2] contient la probabilité que le second mot de la première phrase soit le numéro deux du dictionnaire des cibles. Ces distributions de probabilités sont connues (égales à zéro ou un), étant donnée qu'on a les descriptions.

A partir de la figure 12, on peut voir que les données en entrée sont transformées par la couche Embedding de Keras. encoder_input_data est transformée en un tenseur de taille (198x6x100). Chaque mot clé de chaque descripptions est représenté par un vecteur de dimension 100. J'ai fait la même chose avec la méthode Word2Vec de la bibliothèque gensim (au lieu d'utiliser la couche Embedding de Keras).

5.2.0.2 Séquence de charactères On a procédé de la même façon, lorsque l'encodeur et le décodeur lisent les séquences comme des chaines de charactères. Chaque charactère des entrées (les mots clés) sera numéroté, de même pour les charactères des cibles.

5.3 Mesure de la qualité des phrases générées

5.3.1 Score BLEU

Pour évaluer la qualité des phrases générées, on a utilisé le score BLEU. En effet, ce score permet entre autre d'évaluer la qualité d'un texte traduit de façon mécanique d'une langue à l'autre. Il compare les phrases consécutives de la traduction automatique avec les phrases consécutives qu'il trouve dans la traduction de référence, et il compte le nombre de correspondances de manière pondérée. Ces correspondances sont indépendantes de la position. Un degré de concordance plus élevé indique un degré de similitude plus élevé avec la traduction de référence, et un score plus élevé. L'intelligibilité et l'exactitude grammaticale ne sont pas prises en compte.Il a plusieurs défauts, mais il est déjà implémenté dans la bibliothèque nltk et est facile à utiliser. On l'a utilisé pour mesurer la ressemblance entre la phrase de référence et celle générée.

5.4 Résultats

5.4.1 Données géologiques

```
for seq_index in [0,50,60,70,80,90,100,105, 120,129,150,197]:
    input_seq = encoder_input_data[seq_index: seq_index + 1]
    decoded_sentence = decode_sequence(input_seq)
    print('-')
    print('Input sentence:', input_texts[seq_index: seq_index + 1])
    print('Decoded sentence:', decoded_sentence)
```

```
-
Input sentence: [' lautaret slopes meije of ns']
Decoded sentence:  the the the the the the the the the the the the the the the
-
Input sentence: [' tortonian clay omrane under fig']
Decoded sentence:  the the the the the the the the the the the the the the the
-
Input sentence: [' aussonnelle the section through valley']
Decoded sentence:  the the the the the the the the the the the the the the the
-
Input sentence: [' of system the section associated']
Decoded sentence:  the the the the the the the the the the the the the the the
-
Input sentence: [' rear north cumberland and basementinvolved']
Decoded sentence:  the the the the the the the the the the the the the the the
-
Input sentence: [' location of borehole otway otway']
Decoded sentence:  the the the the the the the the the the the the the the the
-
Input sentence: [' regional regional crosssection']
Decoded sentence:  the the the the the the the the the the the the the the the
-
Input sentence: [' surfaces magallanesaustral of and pasos']
Decoded sentence:  the the the the the the the the the the the the the the the
-
Input sentence: [' curvedslit of breakwater at']
Decoded sentence:  the the the the the the the the the the the the the the the
-
Input sentence: [' the fault young the retreat']
Decoded sentence:  the the the the the the the the the the the the the the the
-
Input sentence: [' total function to cross long']
Decoded sentence:  the the the the the the the the the the the the the the the
-
Input sentence: [' lines acquired and to acquired']
Decoded sentence:  the the the the the the the the the the the the the the the
```

FIGURE 14 – Phrases générées à partir de plusieurs mots-clés choisis au hasard (des données d'entrainement), en considérant que les phrases sont des séquences de mots

```
-
Input sentence:  bans in across de de
Decoded sentence: section crosssection of the serthern and sertion and the sertical section section and the serte the sertict of the serted and sermine the senteral sertion and the serted and sintine the serted and sinthern the section sec
-
Input sentence:  geological cross
Decoded sentence: section crosssection of the sed and and the sertical section sertical section of the serthern and sertion and the serted and sintine the serted and sinthern the section section sertical sertige the sertion and the sertion
-
Input sentence:  and crosscutting available based horizons
Decoded sentence: section crosssection of the serthern and sertion and the sertical section section and the serte the sertict of the serted and sermine the senteral sertion and the serted and sintine the serted and sinthern the section sec
-
Input sentence:  is of active in evolutionary
Decoded sentence: section crosssection of the serthern and sertion and the sertical section section and the serte the sertict of the serted and sermine the senteral sertion and the serted and sintine the serted and sinthern the section sec
-
Input sentence:  observed with stratified bedded thin
Decoded sentence: section crosssection of the sed and and the sertical section sertical section of the serthern and sertion and the serted and sintine the serted and sinthern the section section sertical sertige the sertion and the sertion
-
Input sentence:  datagreen phylite system ductile quartzite
Decoded sentence: section crosssection of the serthern and sertion and the sertical section section and the serte the sertict of the serted and sermine the senteral sertion and the serted and sintine the serted and sinthern the section sec
-
Input sentence:  to undifferenciated basement retaceous precretaceous
Decoded sentence: section crosssection of the serthern and sertion and the sertical section section and the serte the sertict of the serted and sermine the senteral sertion and the serted and sintine the serted and sinthern the section sec
-
Input sentence:  postrift to aptianalbian messinian cretaceous
Decoded sentence: section crosssection of the sed and and the sertical section sertical section of the serthern and sertion and the serted and sintine the serted and sinthern the section section sertical sertige the sertion and the sertion
-
Input sentence:  to lower aptianalbian midburdigalian uppercretaceous
Decoded sentence: section crosssection of the serthern and sertion and the sertical section section and the serte the sertict of the serted and sermine the senteral sertion and the serted and sintine the serted and sinthern the section sec
-
Input sentence:  early burdigalian to early plioquaternary
Decoded sentence: section crosssection of the serthern and sertion and the sertical section section and the serte the sertict of the serted and sermine the senteral sertion and the serted and sintine the serted and sinthern the section sec
-
Input sentence:  crosssection to by burdigalian interpretation
Decoded sentence: section crosssection of the sed and and the sertical section sertical section of the serthern and sertion and the serted and sintine the serted and sinthern the section section sertical sertige the sertion and the sertion
-
Input sentence:  and of deposits the in
Decoded sentence: section crosssection of the serthern and sertion and the sertical section section and the serte the sertict of the serted and sermine the senteral sertion and the serted and sintine the serted and sinthern the section sec
-
Input sentence:  the across figure saintevictoire also
Decoded sentence: section crosssection of the serthern and sertion and the sertical section section and the serte the sertict of the serted and sermine the senteral sertion and the serted and sintine the serted and sinthern the section sec
-
Input sentence:  shading section reflections beneath indicates
Decoded sentence: section crosssection of the sed and and the sertical section sertical section of the serthern and sertion and the serted and sintine the serted and sinthern the section section sertical sertige the sertion and the sertion
-
```

FIGURE 15 – Phrases générées à partir de plusieurs mots-clés choisis au hasard (des données d'entrainement), en considérant que les phrases sont des chaines de charactères

Il n'est pas étonnant que l'on obtienne de meilleurs résultats lorsque l'on considère que les phrases sont des chaines de caractères. Les textes sont courts et peu nombreux, et le vocabulaire est plutôt riche. Le ratio "nombre total de mots/nombre de mots différents"

est bas, et le réseau de neurones aura vu trop rarement chaque mot distinct pour savoir les prédires.

5.4.2 Flickr8k_Dataset

Pour vérifier si les mauvais résultats sont dus à un choix de modèle peu adapté à notre problème ou à un manque de données, j'ai testé l'encodeur-décodeur sur la base de données Flickr8k_Dataset.

```
Input sentence:  in playing are
Decoded sentence: black and white dog is running along the sidewalk

expected sentence:      the children are playing in the water

score: 0.727427152512826
----------------------------
Input sentence:  blond in blue appears wait
Decoded sentence: brown and white dog is running through the snow with the sky in the snow

expected sentence:      blond woman in blue shirt appears to wait for ride

score: 0.6732743971900496
----------------------------
Input sentence:  train looking at on
Decoded sentence: boy in blue shirt and blue shirt and blue shirt and blue shirt and blue shirt and blue shirt and blue shirt and
expected sentence:      little girl looking at brochure on train rides

score: 0.5421921872325952
----------------------------
Input sentence:  dog dog after is running
Decoded sentence: brown dog is running along the sidewalk above large market

expected sentence:      brown dog is running after black dog on rocky shore

score: 0.7431990115298822
----------------------------
Input sentence:  descends of end off
Decoded sentence: brown and white dog is running through the snow

expected sentence:      boy descends off the end of high diving board

score: 0.7348889200874658
----------------------------
Input sentence:  his window taking stands
Decoded sentence: boy in blue shirt and blue shirt and blue shirt and blue shirt and blue shirt and blue shirt and blue shirt and
expected sentence:      guy stands by window taking his overshirt off

score: 0.5421921872325952
----------------------------
```

FIGURE 16 – Phrases générées à partir de plusieurs mots-clés choisis au hasard (des données d'entrainement), en considérant que les phrases sont des chaines de charactères

```
expected sentence:      smiling boy in white shirt and blue jeans in front of rock wall with man in overalls behind him

score: 0.8250731153691881
----------------------------
Input sentence: leaps
Decoded sentence: dog leaps over water

expected sentence:      black dog leaps over log

score: 0.8694417438899827
----------------------------
Input sentence: grey
Decoded sentence: grey dog running on the grass near the water

expected sentence:      grey dog is leaping over fallen tree

score: 0.7331329005620809
----------------------------
Input sentence: leaps
Decoded sentence: dog leaps over water

expected sentence:      large black dog leaps fallen log

score: 0.8507331335123524
----------------------------
Input sentence: grey
Decoded sentence: grey dog running on the grass near the water

expected sentence:      mottled black and grey dog in blue collar jumping over fallen tree

score: 0.7331329005620809
----------------------------
Input sentence: jumped
Decoded sentence: boy jumps off of steep snow covered hill

expected sentence:      the black dog jumped the tree stump

score: 0.7903765226488096
----------------------------
Input sentence: white
Decoded sentence: black and white dog is running through the snow

expected sentence:      brown and white dog is running through the snow

score: 0.7598356856515925
----------------------------
Input sentence: dog
Decoded sentence: brown dog is running through the snow

expected sentence:      dog is running in the snow

score: 0.7790872728179328
----------------------------
Input sentence: through
Decoded sentence: black dog running through the grass with toy in its mouth

expected sentence:      dog running through snow

       - ----------------
```

FIGURE 17 – Phrases générées à partir d'un seul mot-clé choisi au hasard (des données d'entrainement), en considérant que les phrases sont des chaines de charactères

Sur la figure ci-dessus, on peut voir que les phrases générées ont plus de sens que celles sur la figure 16. Le modèle est le même (un modèle séquence to séquence avec l'architecture encodeur-décodeur), en considérant toujours que les séquences sont des charactères, mais entrainé sur cinq fois plus de données. Ces données correspondent toujours aux données Flickr8k_Dataset. Dans cette base de données, chaque image est légendée par cinq descriptions. La figure 16 montre les phrases générées par le modèle entrainé sur la première description de chaque image. Les phrases sur la figure 17 sont générées par le modèle entrainé sur l'ensemble des descriptions. Cependant, je pense que les meilleurs résultats de la figure 17 sont aussi dus au fait que ce que reçoit l'encodeur est bien une séquence : une séquence de un mot. Je vous invite à lire l'article [2]. Les auteurs de cet article ont construit un GAN qui génère des phrases à partir d'un mot-clé et deux autres mots similaires. La partie générateur correspond au modèle sequence-to-sequence. Dans

le cas où on ne considère qu'un mot clé, et sur la base de données Flickr8k_Dataset, nos phrases ont du sens mais ne sont pas satisfaisants.

Table 3. The BLEU Score of two models

Method	BLEU-2	BLEU-3	BLEU-4
MLE	0.7958	0.6120	0.4359
Our model	**0.8182**	**0.6255**	**0.4378**

FIGURE 18 – Sources [2]

Table 4. Example of the generated text according to the input words.

Input word	Generated text
People/ **men**/women	- Two **men** playing a video game in a park bench with a book to it
Restroom/urinals/ **toilet**	- A **toilet** and sink next to each other by the small window in the bathroom
Birthday/ **cake**/chocolate	- A **cake** with a side of a cake lit candles while sliced leaves on a cake
Fruits/ vegetables/stir	- Two **fruits**, one of them on a plate with a knife and fork
Cheese/pizza/ **broccoli**	- **Broccoli**, tomato, cucumber, onion and meat in a plastic container on top of an office
Doubles/apartment/ **streets**	- Two red **streets** sign stand on a wall, a lamp, has a lamp base
Racket/ball/**tennis**	- A **tennis** player wearing a blue outfit stretches up high with his **racket** on a tennis court
Business/ **balcony**/towels	- A **balcony** with chairs and a table is seen through a glass door

FIGURE 19 – Sources [2]

Je n'ai pas pu obtenir de résultats sur des mots-clé préalablement choisis. Les mot-clé ont été choisi aléatoirement.

5.5 D'autres approches

Le modèle sequence-to-sequence peut fonctionner si on ne lui donne qu'un seul mot-clé. On peut donc, après avoir représenté les mots dans une autre dimension par une méthode de word embedding, les sommer. On obtiendra ainsi, à partir des différents mots-clés, un unique mot.

```
expected sentence:      the white and brown dog is running over the surface of the snow

score: 0.6990549173171025
----------------------------
Input sentence:   pictures man blue displaying
Decoded sentence: man in blue standing in the snow and white standing on the street

expected sentence:      man in hat is displaying pictures next to skier in blue hat

score: 0.7124038313502759
----------------------------
Input sentence:   past man man displaying
Decoded sentence: man is standing in the snow and white standing on the street

expected sentence:      man skis past another man displaying paintings in the snow

score: 0.7041908148225626
----------------------------
Input sentence:   looking person pictures wearing
Decoded sentence: man in blue standing in the snow and white standing on the street

expected sentence:      person wearing skis looking at framed pictures set up in the snow

score: 0.7124038313502759
----------------------------
Input sentence:   framed pictures snow
Decoded sentence: the boy playing in the snow of the street

expected sentence:      skier looks at framed pictures in the snow next to trees

score: 0.7730551756939454
----------------------------
Input sentence:   man looking skis
Decoded sentence: man in blue standing in the snow of the street

expected sentence:      man on skis looking at artwork for sale in the snow

score: 0.7638455118619067
----------------------------
Input sentence:   one climbing
Decoded sentence: two man is standing on the street of the street

expected sentence:      collage of one person climbing cliff

score: 0.7071067811865476
----------------------------
Input sentence:   rock rock climbing
Decoded sentence: person in the snow and white standing on the street

expected sentence:      group of people are rock climbing on rock climbing wall

score: 0.6781836812426999
----------------------------
Input sentence:   group rock people
Decoded sentence: three people are standing in the snow of the street

expected sentence:      group of people climbing rock while one man belays

score: 0.7328616209964707
----------------------------
```

FIGURE 20 – Phrases générées à partir de plusieurs mots-clés qui ont été sommé après avoir été transformé par la méthode Word2Vec.

Sur la figure 20, les mots ont été transformés par la fonction word2vec de gensim, dans un espace de taille 50. Ensuite ils ont été sommés. Les phrases sont correctes gramatocalements (avec beaucoup d'indulgence), mais trop éloignées de celles qu'on attend. On peut deviner que le modèle necessite plus de données :comme sur les figures précédentes, on retrouve souvent les mêmes "pattern" tels que "on the street", "on the snow". Le choix des mots-clés semble déterminant aussi.

Une autre idée serait d'encoder les mots clés différement, sans utiliser un réseau de neurones récurrents. C'est le LSTM qui n'est pas adapté, et non pas l'idée d'une architecture encodeur-décodeur.

6 Model de captionning

On a récupéré un modèle [1] lien qui légende directement les images. On peut voir ce modèle aussi comme un "descripteur-decodeur.

— Un réseau de neurones convolutifs (CNN) extrait l'information essentielle et la plus représentative des images.

— Un LSTM "résume" les descriptions.

— La représentation des descriptions par le LSTM et les caractéristiques extraites par le CNN sont passées au décodeur, qui est deux couches de réseau dense.

L'extraction des features des images se fait en dehors du modèle : lors de l'entrainement du modèle, cette partie n'est pas entrainée. Ainsi, les caractéristiques ne sont pas extraites dans l'optique d'obtenir des légendes . De mauvaises légendes peuvent être causées par une mauvaise représentation des images. Cependant, cette approche n'est constitué que de deux parties, la notre trois. Dans notre approche, le choix de l'ensemble de la base de données et l'attribution des mots-clés à chaque image n'est pas ajusté à partir du fait que les phrases générées sont proches de la phrase de référence ou non.

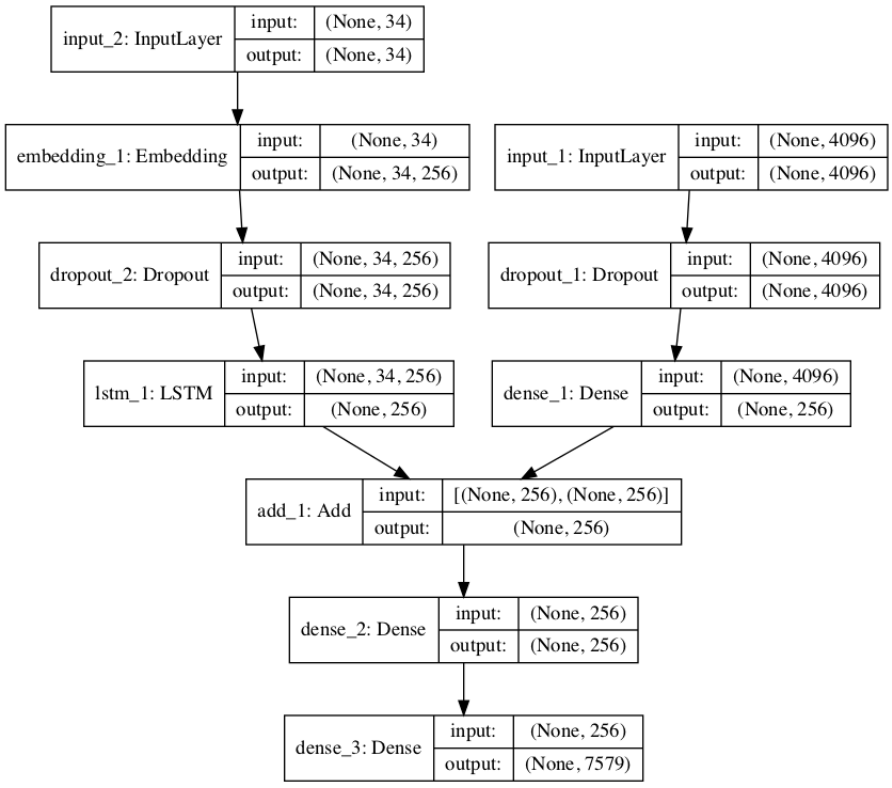


FIGURE 21 – Captioning model

```
startseq unit onlap fig and the beausset syncline ends against the calanques coastline and the calanques coastline and the marseillais canyon and west of the cass
```

FIGURE 22 – Exemple de phrase généré par le modèle captioning

Testé sur notre base de données les résultats 22sont médiocres : il n'y pas assez d'images et de légendes. On peut augmenter les données et chercher un LSTM pré entrainé.

7 Conclusion

Couper le problème en trois partie nous a permis de mieux appréhender le sujet. Mais cette approche à un défaut : la similarité des phrases générées va dépendre du mot-clé utilisés. Le choix des mots-clés qui doivent résumer les descriptions ne s'ajuste pas lors de l'entrainement du modèle qui génère les phrases. Or on peut mesurer la justesse du choix des mots-clés en mesurant la qualité des phrases générées. Il serait interessant de relier directement ces deux parties.

La classification multi-classe et le générateur de phrases ont besoin de davantages de données. Dans ce cas là, sur la base de données Flickr8_datasete, le modèle sequence-to-sequence donne des résultats plutôt corrects si on n'utilise qu'un mot-clé et si on les compare aux résultats de [2]. Mais un mot-clé n'est pas suffisant pour obtenir les légendes demandées : on peut esperer obtenir des phrases grammaticalement correctes mais pas nécessairement proches de celles souhaitées. Dans la littérature, peu d'articles traite de ce sujet : générer des phrases à partir de mots clés.

On peut augmenter la taille des données et utiliser des réseaux de neurones entrainés, pour obtenir de meilleurs résultats.

Références

[1] Jason Brownlee. **How to Develop a Deep Learning Photo Caption Generator from Scratch**. 2019.

[2] Dongju Park et Chang Wook Ahn. **LSTM Encoder-Decoder with Adversarial Network for Text Generation from Keyword**. 2018.

[3] Harshall Lamba. **Word Level English to Marathi Neural Machine Translation using Encoder-Decoder Model**. 2019.