Recherche croisée entre images et textes

But du projet

L'objectif de ce projet est d'exploiter le deep learning pour mettre en oeuvre une recherche croisée entre des images et des textes (textes simples comme des titres pour commencer). A partir d'une image donnée en entrée, il doit être possible de retrouver les textes les plus probables correspondant à cette image. Inversement, étant donné un texte descriptif, il faut être capable d'obtenir les images les plus représentatives de ce texte. Ce problème de recherche croisée à de nombreuses applications à la fois dans l'e-commerce, les services mais aussi dans des projets industriels.

Les jeux de données que nous allons utilisés sont constitué d'images et de textes les décrivants (captions) comme dans l'exemple suivant :

```
#0    Une femme en jeans et une femme en short caressent un bébé kangourou .
#1    Deux filles caressent un petit kangourou dans un champ herbeux.
#2    Deux femmes se penchent pour toucher un petit kangourou .
#3    Deux femmes caressent un bébé kangourou dans le parc.
#4    Deux femmes caressent un kangourou dans un parc.
```

Le but premier du projet est d'implémenter un modèle ayant les fonctions suivantes :

- Déterminer les textes décrivant le mieux une image donnée.
- Rechercher dans le jeu de données les images proches d'un texte donné.
- Rechercher les images proches d'un couple (texte,image) donné.

Pour ce faire nous allons utiliser des réseaux de neurones capables d'abstraire les données d'entrée et de les projeter dans le même espace vectoriel.

Ce projet étant orienté vers la recherche, le cahier des charges n'est pas fixe et est susceptible d'évoluer selon mon avancée.

Il sera par exemple possible d'intégrer la notion de région dans le traitement de l'image.
Il est également souhaité d'intégrer le modèle obtenu à une application mobile existante.

Travail effectué

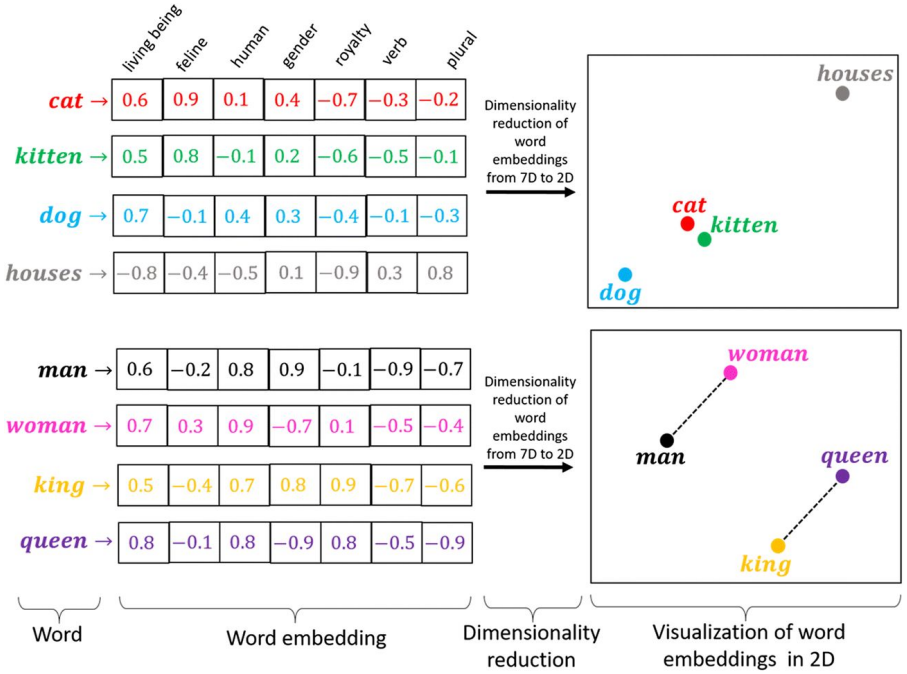
Représenter des textes et des images

Afin d'effectuer de la recherche croisée sur du textes et des images il est nécessaire d'homogénéiser les données. Pour cela nous allons utiliser deux modèles pré-entraînés.

Pour les images nous utiliserons le modèle Resnet50 dont l'architecture détaillée est donnée en bibliographie[2]. Ce réseau de neurones transforme des images (224*224 pixels) en vecteur d'entiers de taille 1000. Ce vecteur d'entier contient les notions abstraites présentes dans l'image et permet d'effectuer des traitements mathématiques.

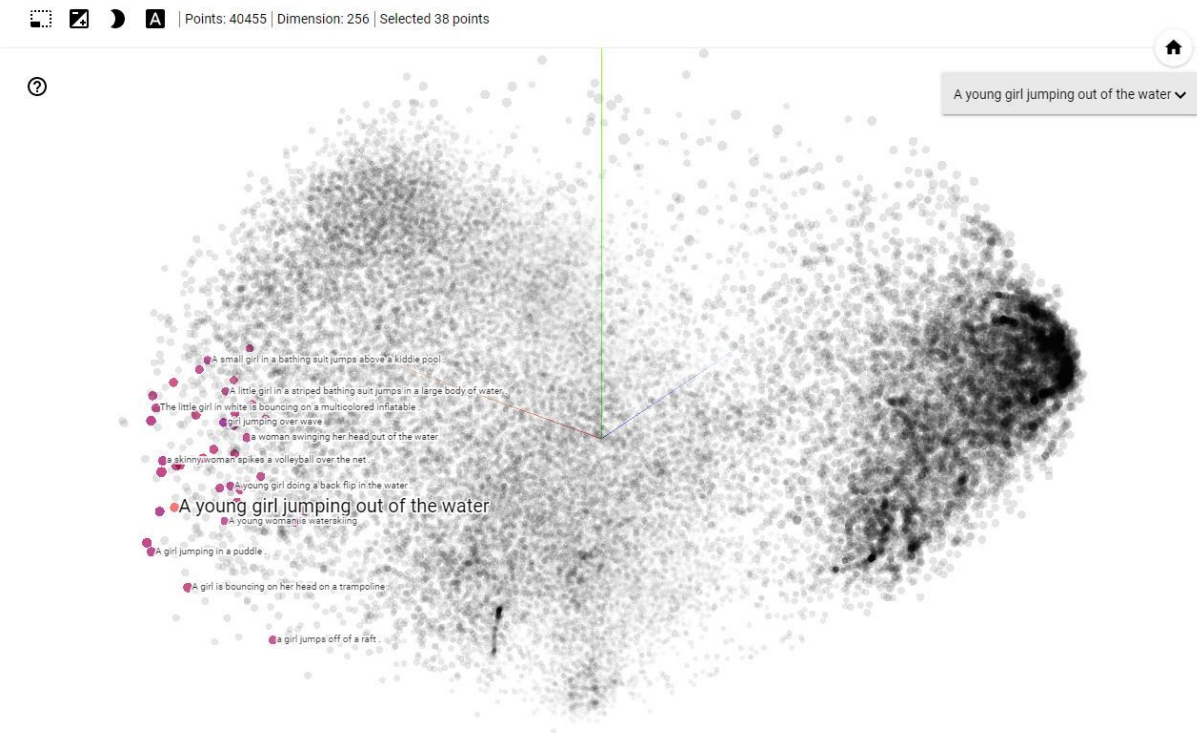
Pour les textes nous utiliserons un encodage par 'word embedding'. Avec les textes en français nous utiliserons les vecteurs de mot de l'algorithme word2vec développé par Google, avec les textes anglais nous utiliserons les vecteurs GloVe developpé par Twitter . Cet algorithme convertit une chaîne de caractères (phrase) en une matrice de dimension nombre de mots * 100 dans laquelle chaque ligne correspond à un mot et chaque colonne correspond à une dimension, comme dans l'exemple suivant :

Exemple de word embedding



Il est intéressant de noter que dans la représentation ‘word embedding’ la translation linéaire entre ‘man’ et ‘woman’ est identique à la translation entre ‘king’ et ‘queen’. De même pour la translation France->Paris et Allemagne->Berlin.

Il est possible de projeter ces données dans un espace vectoriel afin d’évaluer la cohérence de notre représentation, en regardant si les points les plus proches font partie du même champ lexical par exemple. Nous avons utilisé tensorflow[6] pour obtenir cette projection de 40455 points (en utilisant une ACP) :



Nous avons donc ici nos deux types de données (image et texte) sous forme de vecteur d’entier.

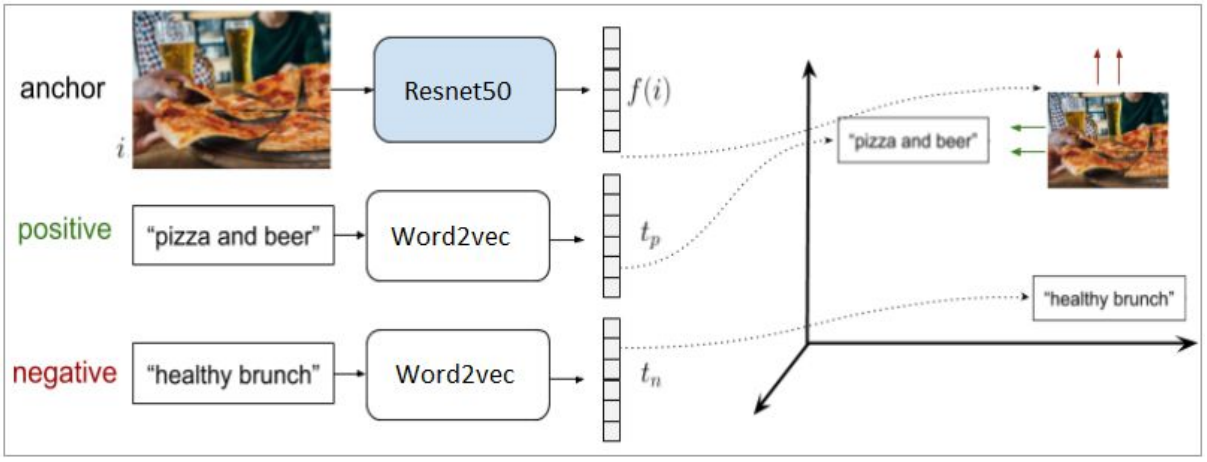
Fonction de perte

Une fois les données homogénéisées, nous souhaitons les projeter dans un même espace vectoriel.

Pour cela nous allons définir des paires positives (positive pair) et des paires négatives (negative pair) [3] .
Une paire positive correspond à une image et un texte la décrivant.
Une paire négative correspond à une image et un texte ne la décrivant pas (ou étant sémantiquement éloigné).

Le réseau prendra donc en entrée ce triplet (image, texte positif, texte négatif) et va chercher une représentation dans laquelle les paires positives seront proches, tandis que les paires négatives seront éloignées.

Exemple de projection de triplet



Pour trouver cette représentation, le réseau va chercher à minimiser la fonction de perte suivante :

$$L(i, t_p, t_n) = max(0, m + ||f(i) - t_p|| - ||f(i) - t_n||)$$

Avec :

- i l'image
- $f(i)$ sa représentation par Resnet50
- t_p, t_n les embeddings des textes positifs et négatifs
- *m* une marge choisie.

On a donc :

$\|f(i) - t_p\|$ la distance entre l'image et son texte positif (dans l'espace vectoriel).
$\|f(i) - t_n\|$ la distance entre l'image et son texte négatif

Voici les 3 situations de cette fonction de perte :

- Triplet simple : $\|f(i) - t_n\| > \|f(i) - t_p\| + m$
  Dans ce cas l'élément t_n est déjà assez éloigné de l'image (plus grande que la distance positive + m) la fonction de perte L vaut donc 0 et les poids ne sont pas mis à jour.

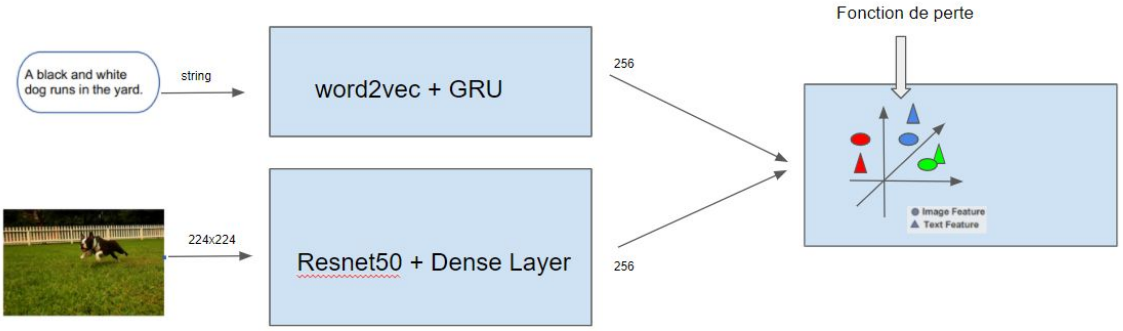
- Triplet dur : $\|f(i) - t_n\|) < \|f(i) - t_p\|$
  Le texte négatif est plus proche de l'image que le texte positif, la perte est donc positive et plus grande que m.

- Triplet mixte : $\|f(i) - t_p\| < \|f(i) - t_n\| < \|f(i) - t_p\| + m$
  Le texte négatif est plus éloigné de l'image que le texte positif mais la distance n'est pas plus grande que m, la perte est donc positive et plus faible que m.

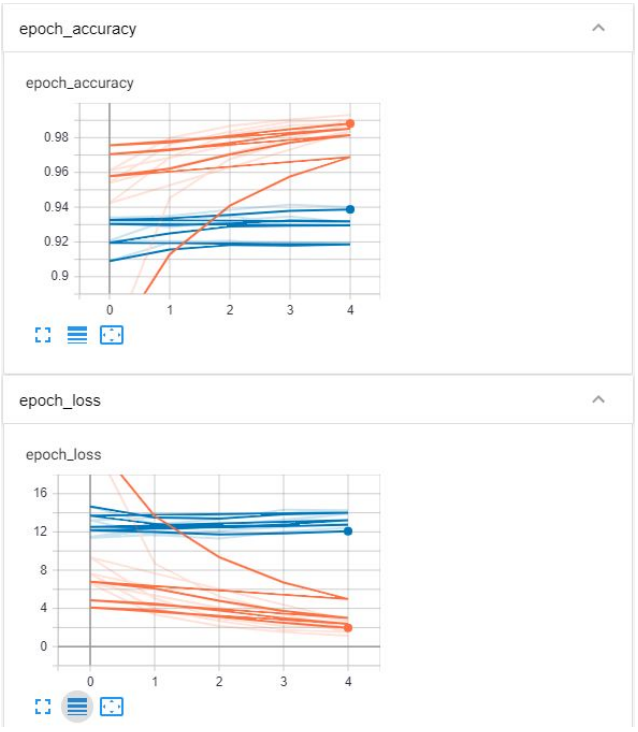
A noter qu'il est important de choisir des triplets durs (ou mixtes) et d'éviter les triplets simples car ces derniers ne mettent pas à jour les poids du réseau. Pour le moment les paires négatives sont choisies aléatoirement, pour la suite nous allons mettre en place un outil de recherche de triplet dur avec d'améliorer les performances du modèle.

L'architecture de notre réseau est donc la suivante :

Architecture du réseau de neurone utilisé



Nous avons enregistré la perte et la précision lors de l'apprentissage et l'avons affiché à l'aide de tensorboard :



L'entraînement était composé de 5 epochs réalisés 4 fois en mélangeant les paires négatives aléatoirement.
Une précision de 99% signifie que 99% du temps, notre modèle fait un meilleur choix qu'un choix aléatoire.

Applications

Nous avons donc entraîné le modèle décrit précédemment en utilisant Google Collab (puissance de calcule mis à disposition par Google) et le jeu de données Flicker 8k[4] composé de 8.000 images chacunes décrite par 5 textes différents. Les textes ont été traduits de l'anglais à l'aide de DeepL.
Pour implémenter ce modèle nous avons choisi Keras une API de Deep Learning Python (utilisant Tensorflow).

Les résultats obtenus sont encourageants mais nécessitent d'être améliorés (notamment la recherche d'image).

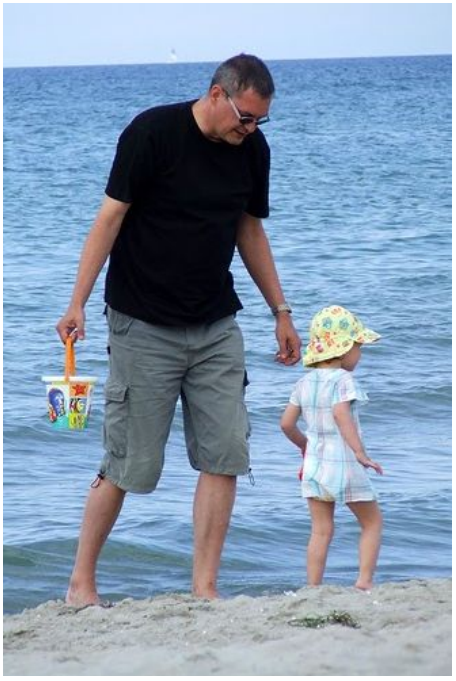
Voici quelques résultats obtenus :

Image d'entrée :

Textes proposés :

```
6.764753 Brown and white dog jumps over purple and white bar .
6.530719 A dog sits with its paws on a laptop computer .
6.2190294 A small white dog jumps on a chair
6.014408 A dog jumps off a wooden porch .
5.9305477 The brown dog eats from a bowl on the wooden table as a black
dog jumps off .
5.8392015 A brown and white dog jumps onto a beige leather chair .
5.7665153 A brown dog sits on a couch .
5.759899 A small dog laying on a stuffed animal in front of a TV
5.753727 A brown dog is resting its paws on a laptop keyboard .
```

Image d'entrée :



Textes proposés :

8.634671 Une jeune fille porte une chemise et un short alors qu'elle joue dans les vagues de l'océan.
8.264197 Un garçon en chemise orange et chapeau rouge se tient à côté d'une fille en chemise blanche et pantalon bleu dans le sable, au bord de l'océan.
8.252998 Une femme et une petite fille jouant dans le sable au bord de l'océan
8.016668 Deux garçons, l'un en short jaune et l'autre en short rouge, jouent avec des body boards dans l'océan.
7.740966 Une fille marche à côté des petites vagues qui s'écrasent dans l'océan.
7.732913 Une femme sans tête tient la main de son enfant nu pendant qu'il se tient dans les vagues de l'océan.
7.6751504 Une jeune fille se tient sur le rivage et montre l'océan.
7.67205 Une femme en short bleu porte ses chaussures en marchant sur la plage au bord de l'océan.
7.641971 Une fille vêtue d'un sweat-shirt rose et d'une jupe à rayures roses et blanches joue dans les vagues à la plage .
7.6100235 Une femme en combinaison de surf dans l'océan .

Recherche texte -> texte correspondant :

Des enfants jouent au football dans l'herbe

9.215866 Deux jeunes garçons jouent avec un ballon de foot dans un étang au milieu d'un terrain herbeux .
9.206055 Un homme torse nu en short coloré frappe un ballon de volley-ball lors d'un match de beach-volley .
9.041428 De jeunes enfants tapent dans un ballon de foot sur un terrain de foot .

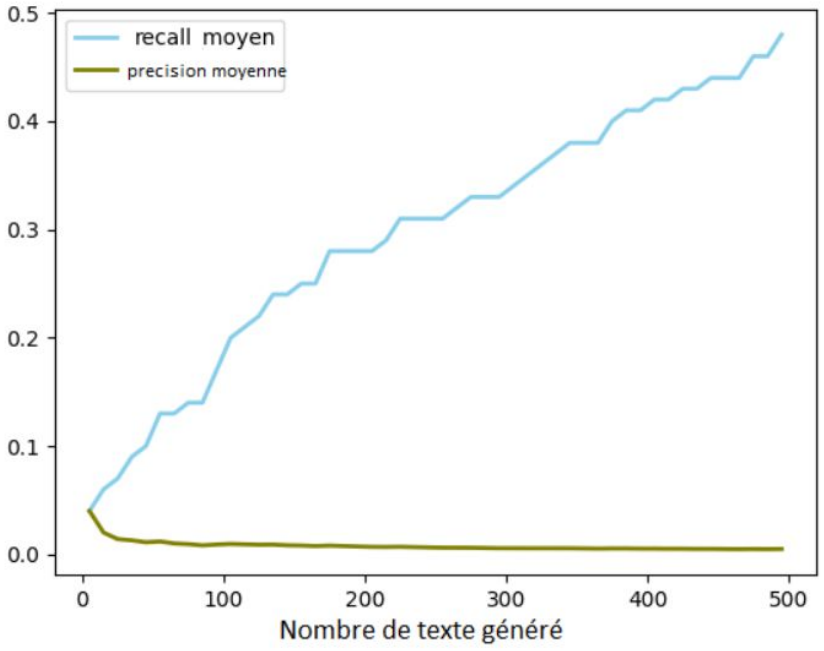
```
8.983473 Une jeune fille s'entraîne au volley-ball dans un champ de
gazon.
8.961634 Le numéro 13 lance un ballon de foot vers le but pendant un
match de foot pour enfants.
8.945414 Le joueur de football court avec le ballon alors qu'un joueur
de l'équipe adverse attrape sa jambe
8.876382 Sur un terrain de football, cinq jeunes garçons courent et
jouent au football.
8.868202 Quelques jeunes garçons lancent un ballon de football
au-dessus d'un filet de volley-ball .
8.837126 Quatre enfants courent le long d'un terrain en herbe pour
jouer au football.
8.785065 trois garçons jouent au football près d'un filet de
volley-ball sur la plage .
```

Les applications de ces programmes sont nombreuses : dans le e-commerce afin de trouver des articles en ajoutant des précisions écrites, dans l'imagerie médicale afin de générer des diagnostics, etc...

Métrique d'évaluation

Nous avons réussi à obtenir des résultats cohérents sur les fonctions de génération de texte à partir d'une image. Nous souhaitons maintenant évaluer ces résultats. Pour cela nous avons généré des textes décrivant une image du jeu d'entraînement et nous avons comparé ces textes aux 5 vraies descriptions de cette image.
Voici un graphique du recall (proportion des 5 vrais descriptions présents dans les textes générés) et de la précision (proportion de textes correctement prédits) :

Recall et précision en fonction du nombre de texte généré



Il est important de noter que même si les textes générés ne sont pas toujours les ‘vraies’ descriptions de l’image, ceux-ci sont sémantiquement très proches et cohérents.
De nouvelles mesures vont être faites notamment en considérant comme valide les textes les plus proches des captations de l’image.

Amélioration du modèle

Une manière d’améliorer les performances du modèles consiste à construire des triplets de telle sorte que les poids du réseau soient mis à jour de la manière la plus efficiente possible. Des triplets simples ne mettent pas à jour les poids et donc le réseau n’apprend rien, tandis que des triplets durs vont mettre à jour les poids du réseau de manière trop importante et ‘effacer’ les notions qu’il aura appris précédemment.
Nous avons donc choisi l’implémentation suivante [5] : 50% de nos triplets sont construits aléatoirement et 50% sont des triplet semi-difficile. Cela garantit un apprentissage constitué des trois types de triplets possible.

Dans un second temps nous avons réalisé un ‘fine-tuning’ du réseau de neurones Resnet50. Cela consiste à utiliser le réseau pré-entraîner et de réaliser un second entraînement sur notre jeu de données d’image. Un tel entraînement permet au réseau de mieux coller à nos image en apprenant par exemple des concepts qu’il n’aurait pas vu lors de son entraînement initial.

Conclusion

Ce que j’ai appris

Ce projet m’a permis de manipuler des réseaux de neurones appliqués à deux domaines différents (NLP et Computer Vision). Cela m’a permis de comprendre l’architecture des réseaux de neurones, leur entraînement et leur utilisation.

Afin de répondre aux différentes problématiques j’ai également dû manipuler des données mixtes avec un volume important (8.091 images et 40.455 textes).

Ce projet m’a également permis de découvrir et de manipuler des techniques propres aux NLP : le word embedding et l’utilisation de transformers qui n’ont pas étés étudiés au cours de la formation IS.

Je peux donc dire que j’ai beaucoup appris au cours de ce projet.

Difficultés rencontrées

La principale difficulté a été de réaliser ce projet à distance. Pour pallier cela nous avons fait des réunions en visioconférence une fois par semaine et nous échangions régulièrement par mail, notamment en cas de difficulté ou de problème de mon côté.

La seconde difficulté a été de s’approprier l’architecture du réseau utilisé. Ayant au début peu de connaissance dans le domaine du Deep Learning il m’a fallu du temps pour comprendre le parcours des données dans le réseau.

Conclusion

Pour conclure ce projet, je souhaiterais remercier Hazem pour leur encadrement. Ce projet a été très enrichissant car j'ai pu y découvrir des concepts avancés de Deep Learning. De plus le sujet a été captivant de par sa technicité mais également par le fait qu'il réponde à des problématiques actuelles (e-commerce, santé, etc..).

Ce projet m'a permis de beaucoup apprendre et était stimulant.

Bibliographie

1 Joint text and image representations - Benoit Favre 2017

2. Architecture détaillée du modèle Resnet50

3. Understanding Ranking Loss, Contrastive Loss, Margin Loss, Triplet Loss, Hinge Loss and all those confusing names - Raúl Gómez blog

4. Flicker 8k Dataset

5. Triplet Loss and Online Triplet Mining in TensorFlow

6. https://projector.tensorflow.org