

# Visualisation de réseaux de neurones.

Louët Joseph & Karmim Yannis  
Spécialité **DAC**



## **Table des matières**

<b>1</b>	<b>Carte de saillance</b>	<b>3</b>
<b>2</b>	<b>Exemples adversaires</b>	<b>5</b>
<b>3</b>	<b>Visualisation de classes</b>	<b>7</b>

# 1 Carte de saillance

## Questions

1) Nous avons obtenu des cartes de saillance de deux manière différentes. La première est, comme dans l'énoncé, calculer le gradient à partir de la sortie de la classe de l'image. La deuxième consiste à calculer le gradient à partir d'une cross-entropie sur la sortie du réseau et la classe de l'image Voici les cartes de saillance que nous avons obtenu :

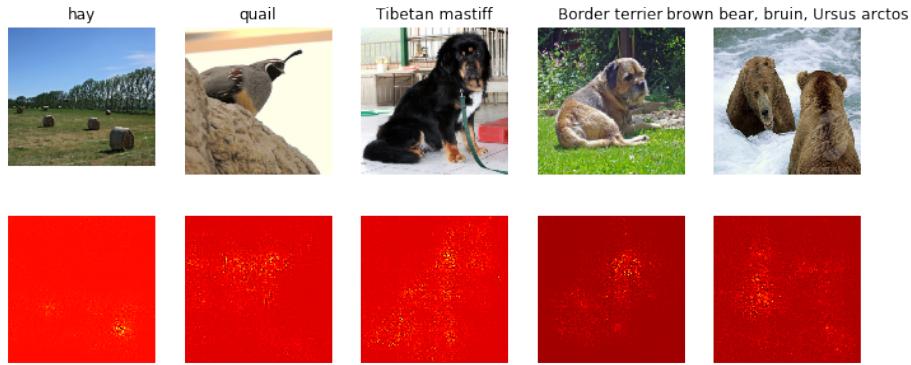


FIGURE 1 – Carte de saillance pour le réseau SQUEEZENET avec le gradient calculé à partir de la sortie

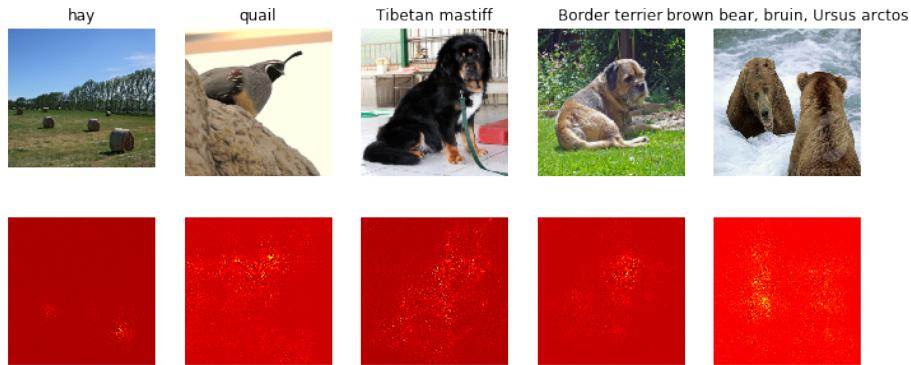


FIGURE 2 – Carte de saillance pour le réseau SQUEEZENET avec le gradient calculé à partir de la cross-entropy

Pour l'image de foin, on peut remarquer que les pixels les plus importants sont ceux situés sur les bottes de foin. Pour l'image de caille, les pixels les plus importants se situent principalement sur le corps de la caille et plus particulièrement autour de ses yeux. En revanche, il y a quelques pixels importants situés

au bas de l'image, sur le socle de la caille. Pour l'image de *Mastiff Tibétain*, on remarque que les pixels les plus importants se situent sur l'ensemble du corps de l'animal. On peut faire le même constat pour l'image de *Border Terrier*. Enfin, pour l'image d'ours brun, les pixels les plus importants se situent principalement sur le visage de l'ours brun (de face) et très peu sur le dos de l'autre ours brun.

2) Cette technique a pour but d'identifier les pixels les plus importants pour la décision. En revanche, nous pouvons fausser cette prédition en modifiant quelques pixels qui ne seront pas visibles à l'oeil nu. De plus, grâce aux couches de convolutions, le réseau identifie des courbures ou des particularités sur plusieurs pixels. De ce fait, nous ne pouvons pas vraiment résumer l'importance des pixels un par un puisque ces derniers font partie d'un ensemble de pixels où nous pouvons identifier quelque chose de particulier.

3) Cette technique peut nous permettre d'améliorer notre réseau. En effet, nous pouvons capter les pixels qui influe le plus sur notre décision. Avec cette information, nous pouvons modifier les pixels les plus importants sur nos images d'apprentissage et utiliser ces images pour que notre réseau devienne plus robuste au bruit.

4) Voici les cartes de saillance que nous avons obtenu avec le réseau VGG16 :

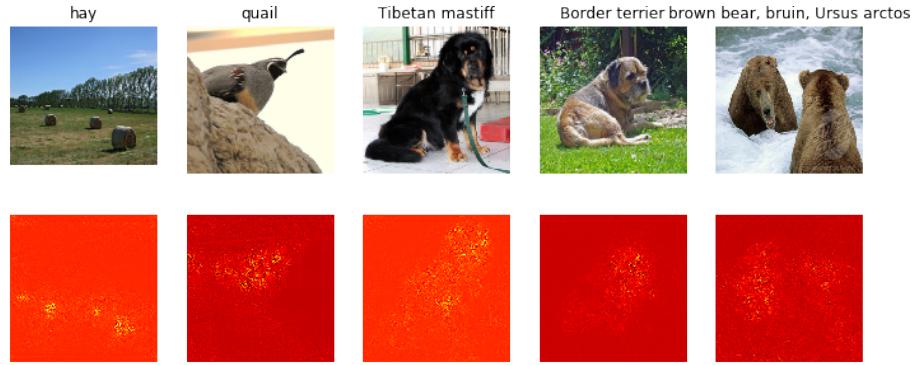


FIGURE 3 – Carte de saillance pour le réseau VGG16 avec le gradient calculé à partir de la sortie

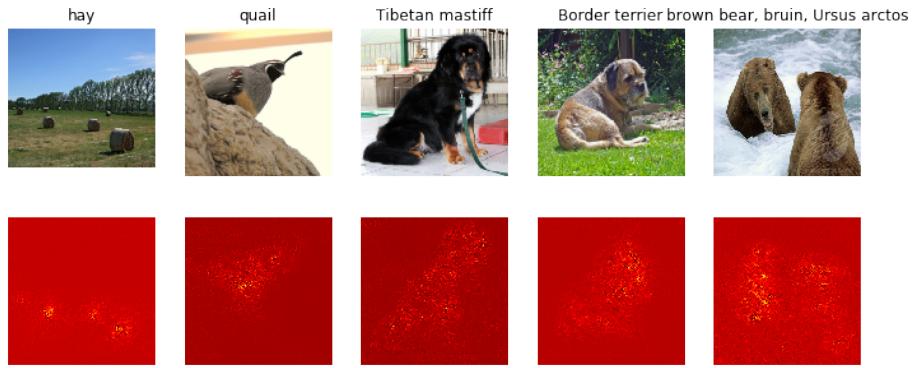


FIGURE 4 – Carte de saillance pour le réseau VGG16 avec le gradient calculé à partir de la cross-entropy

## 2 Exemples adversaires

### Questions

5) De même que pour les cartes de saillance, nous avons procédé de deux manière différentes. La première manière est de faire une montée de gradient sur le gradient calculé à partir de la sortie. La seconde manière est d'effectuer une descente de gradient sur le gradient calculé à partir d'une cross-entropie sur la sortie et la classe que nous désirons obtenir.

Voici les résultats obtenus avec la technique des exemples adversaires :



FIGURE 5 – Résultats obtenus avec les exemples adversaires avec une montée de gradient sur le gradient calculé à partir de la sortie

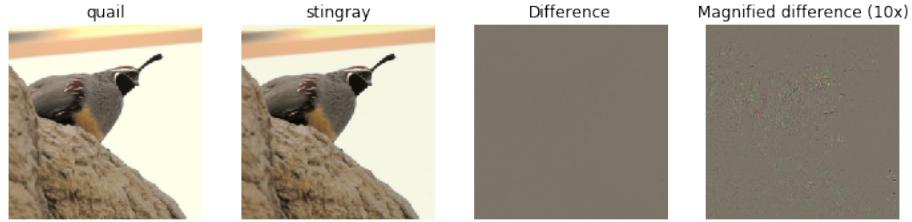


FIGURE 6 – Résultats obtenus avec les exemples adversaires avec une montée de gradient sur le gradient calculé à partir d'une cross-entropie

Ainsi, nous pouvons remarquer que les différences entre l'image initial et l'image modifiée ne sont pas visibles à l'oeil nu. De plus, nous pouvons aussi observer que les « différences » sont très peu visibles malgré la troisième carte. Seulement, la quatrième carte permet de visualiser où les différences ont été faites. On remarque également que le réseau se trompe pour la classification de l'image en entrée et ne prédit plus *quail* mais *stingray*. De plus, il est intéressant de remarquer que la montée de gradient a trompé le réseau en 85 itérations alors qu'en effectuant une descente de gradient, le réseau a été trompé en 37 itérations.

6) En pratique, cela montre que les réseaux convolutionnels s'attachent à des éléments qui ne sont pas visibles par l'oeil humain. Ainsi, toutes les images d'un dataset peuvent être modifiées pour obtenir une classe choisie sans pour autant changer sa classification par un oeil humain.

### 3 Visualisation de classes

8). En gardant les paramètres initiaux et en implémentant le code permettant de visualiser les classes, nous obtenons les figures ci dessous :

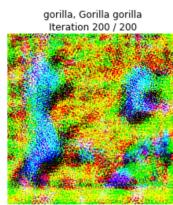


FIGURE 7 – Visualisation de la classe gorille

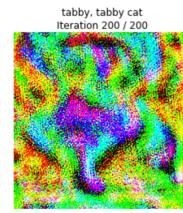


FIGURE 8 – Visualisation de la classe chat

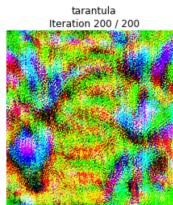


FIGURE 9 – Visualisation de la classe tarentule

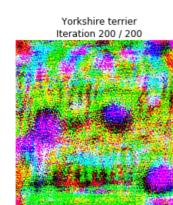


FIGURE 10 – Visualisation de la classe terrier

Pour la **Figure 9** on visualise bien les pattes et le corps volumineux pouvant caractériser une tarentule. On ne peut pas trop interpréter les résultats pour les autres classes avec ces paramètres.

9.)

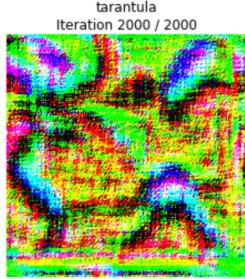


FIGURE 11 – Visualisation de la classe tarentule avec : learning\_rate = 2, num\_iterations = 2000, l2\_reg = 10e-2

Avec 10 fois plus d’itérations, un learning\_rate plus petit et une pénalité L2 10 fois plus grande, on obtient des résultats plus satisfaisant. Dans la **Figure 11** on discerne mieux que les caractéristiques qui maximisent la classe tarentule sont les longues et multiples pattes. En diminuant le learning\_rate ainsi qu’en augmentant le nombre d’itérations, nos images ont plus de temps pour converger et plus précisément.

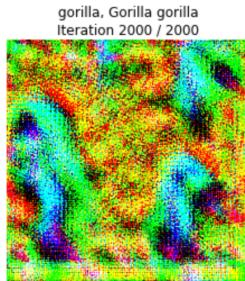


FIGURE 12 – Visualisation de la classe gorille avec : learning\_rate = 2, num\_iterations = 2000, l2\_reg = 10e-2

On distingue maintenant mieux les formes caractéristique d’un gorille dans la **Figure 12**.

10.) Afin de visualiser une classe, nous ne partons plus désormais d'une image aléatoire, mais directement d'une image d'ImageNet qui correspond à la classe choisie.

On distingue bien que les caractéristiques de la classe *Tibetan mastiff* sont de

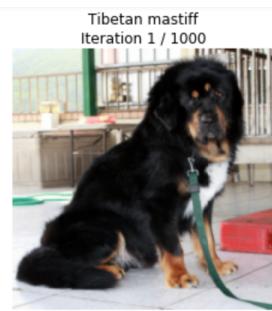


FIGURE 13 – Image de chien de la base ImageNet

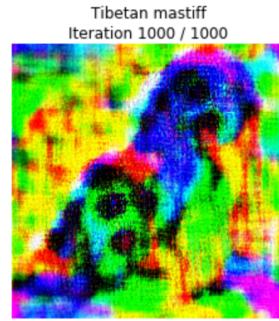


FIGURE 14 – Visualisation de la classe chien après 1000 itérations et un learning\_rate de 0.1.

grandes oreilles, un grand museau et des oreilles tombantes.

L'intérêt est que notre image convergera plus rapidement vers une solution satisfaisante, car en partant d'une image initialisé aléatoirement le résultat peut être beaucoup moins stable et la montée de gradient plus difficile.

11.) On va maintenant visualiser quelques classes avec le réseau **VGG16** et non plus Squeeze\_Net. On a choisi de partir de la même image de chien tiré de la base d'imageNet, puisque le modèle VGG16 est plus lourd et on converge beaucoup plus lentement que Squeeze\_Net vers une solution convenable à partir de données générées aléatoirement.



FIGURE 15 – Image de chien de la base ImageNet

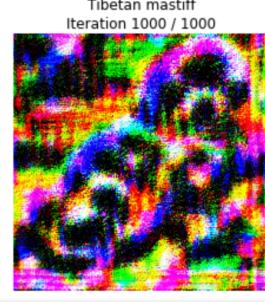


FIGURE 16 – Visualisation de la classe chien après 1000 itérations et un learning\_rate de 0.5.