HAI918I - Image, Sécurité et Deep Learning
TP4

Ingo Diab

2023

Table des matières

L'objectif de ce tp était de coder un CNN afin d'extraire le vecteur caractéristique de certaines images et d'afficher la valeurs des neurones après le réseau fully connected (seulement la forward propagation). Pour ce tp, j'ai utilisé le dataset Cats-vs-Dogs disponible ici : `https://www.kaggle.com/datasets/shaunthesheep/microsoft-catsvsdogs-dataset?select=PetImages`

1 Modèle CNN

Le modèle de CNN que j'ai créé est composé de 4 couches (filtres, fonction d'activation et maxpooling), d'un flattening puis d'une output layer composée de 2 neurones.

```
vector<ImageBase> _outputLayer1, _outputLayer2, _outputLayer3, _outputLayer4;

CNNLayer(_in, {gauss, moyenneur, laplacien}, _outputLayer1);
CNNLayer(_outputLayer1, {contourH, moyenneur2, contourV}, _outputLayer2);
CNNLayer(_outputLayer2, {sharpen}, _outputLayer3);
CNNLayer(_outputLayer3, {moyenneur, flou}, _outputLayer4);

vector<unsigned char> _outputFlattened;
Flatten(_outputLayer4, _outputFlattened);

vector<float> _normalizedLayer, _outputLayer;
Normalisation(_outputFlattened, _normalizedLayer);

NeuralLayer(2, _normalizedLayer, _outputLayer);
```

FIGURE 1 – Mon modèle CNN

2 Convolutions

Chaque image propagée dans le réseau se voit appliquer des filtres. Nous avons choisi des filtres passe bas et passe haut déjà existant au lieu d'utiliser des filtres avec des poids random. Ces filtres nous permettent de réduire la taille des images en s'intéressant à certaines caractéristiques.

FIGURE 2 – Image originale - filtre Gaussien - filtre Laplacien

3 Fonction d'activation

Dans ce modèle, j'utilise la fonction ReLU qui renvoie 0 si la valeur est négative ou la valeur elle-même si elle est positive.

4 MaxPooling

Le MaxPooling nous sert à sous-échantillonner l'image en entrée afin d'avoir une image plus petite que celle d'entrée en perdant le minimum de qualité possible.

FIGURE 3 – Image originale - MaxPooling

5 Flattening

Après avoir traversé toutes les couches du CNN, nous obtenons un certains nombres de petites images. Nous allons les transformer en un vecteur 1D puis les concaténer afin d'avoir un unique vecteur 1D. Pour pouvoir l'afficher nous allons le retransformer en une image 2D.

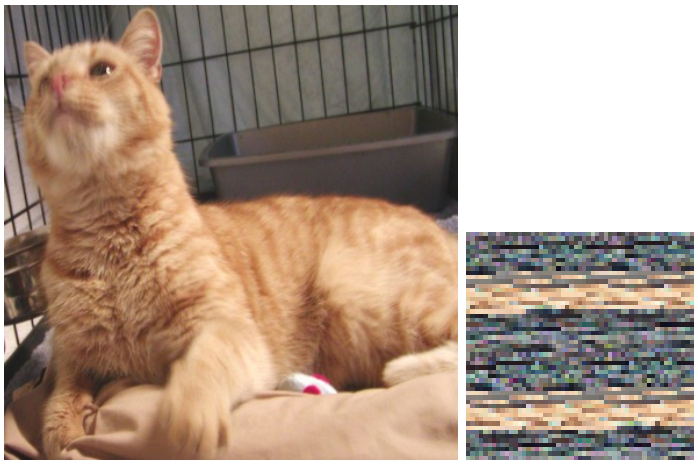

FIGURE 4 – Image originale - Vecteur Caractéristique

Nous pouvons observer deux bandes étant de la même couleur que la couleur dominante de l'image original (orange). Ces bandes sont les imagettes qui ne sont pas passées par les filtres passe haut (par exemple les imagettes ayant subi gaussien puis moyenneur puis sharpen puis moyenneur).

6 Fully Connected

Après avoir normalisé les valeurs dans le vecteur caractéristique, nous allons les utiliser dans un réseau de neurone fully connected. Chaque neurone va être la somme pondérée des neurones précédents à laquelle nous allons appliqué la fonction d'activation (ReLU). Ces valeurs ne sont pas comprises entre 0 et 1 car nous n'avons pas appliqué le softmax. Les poids entre les neurones sont random car nous n'avons pas de backward propagation, les valeurs finales sont donc randoms.

```
Neurone 1: 8.79332
Neurone 2: 15.3662
```

FIGURE 5 – Résultats

7 Autres résultats

J'ai ensuite testé mon CNN sur d'autres photos de Chats du dataset ainsi que sur la classe Chien.

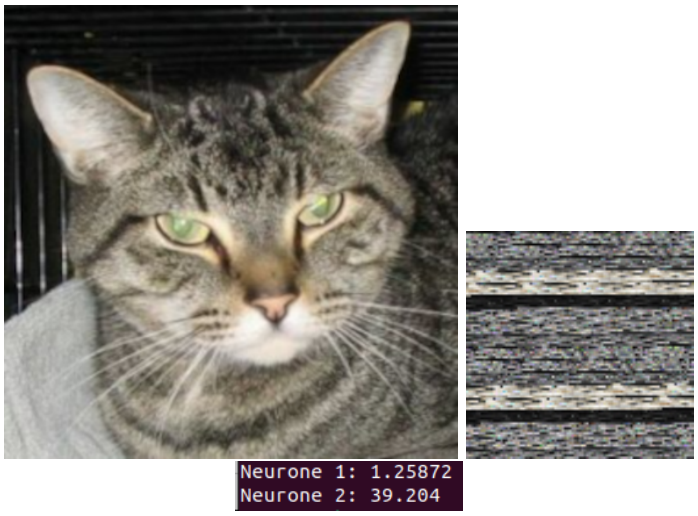

FIGURE 6 – Image originale - Vecteur Caractéristique - Résultat

FIGURE 7 – Image originale - Vecteur Caractéristique - Résultat

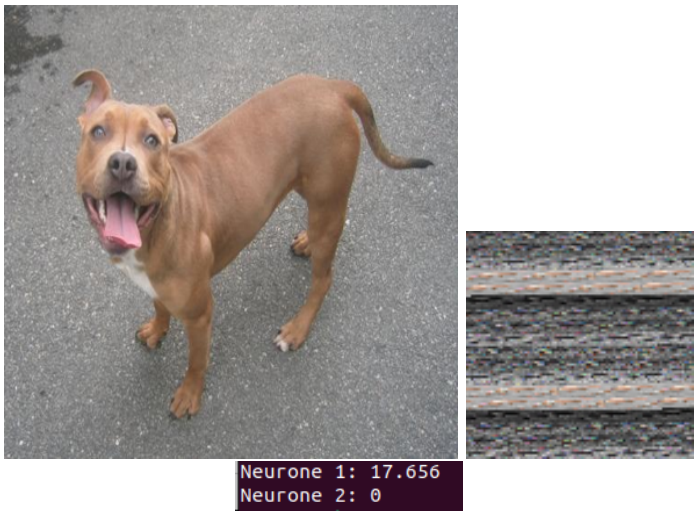

FIGURE 8 – Image originale - Vecteur Caractéristique - Résultat

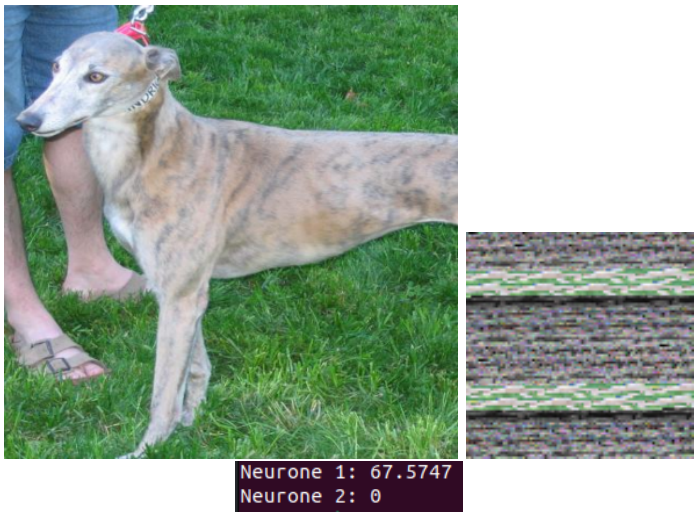

FIGURE 9 – Image originale - Vecteur Caractéristique - Résultat