

Reconnaissance des formes pour l’analyse et l’interprétation d’images

Rapport TP 5–6 : Réseaux convolutionnels pour l’image

**Etudiant** : **Numéro :**

DJEGHRI Amine 3801757

MAMOU Idles 3803676

Novembre 2020

# Partie 1 – Introduction aux réseaux convolutionnels

1. En considérant un seul filtre de convolution

* La taille de sortie sera :

* Nombre de poids à apprendre est: pour le bias
* Le nombre de poids qu’il aurait fallu apprendre si une couche fully-connected devait produire une sortie de la même taille :

**2.**

Les avantages apportés par la convolution par rapport à des couches fully-connected :

* Complexité inférieure en terme de poids (apprendre moins de poids)
* On est indépendant aux dimensions d’entrée
* Voir des bouts de l’image (notion de localité) et détecter les patterns

Sa limite est : La perte d’information

**3.** L’intérêt à l’usage du pooling spatial est : réduire la dimension des features maps par conséquent réduire le nombre de paramètres et aussi l’invariance aux transformations.

**4.** On peut calculer sans modifier l’image s’il y a au début du réseau **des couches de convolutions** car elles ne dépendent pas de la taille de l’entrée contrairement aux couchesfully connected, **mais on va s’arrêter au passage de la couche de convolution à la couche fully connected**

**Pour parier à ca : on peut faire un global pulling**

**5.** On peut voir les couches fully-connected comme des convolutions particulières de taille 1x1 et ont la même taille que l’image d’entrée avec un padding nul

**6.**

**7.**

# Partie 2 – Apprentissage from scratch du modèle

## 2.1 Architecture du réseau

**8.** Les valeurs de padding et de stride à choisir sont :

La règle :

Exemple :

Stride = 1

Padding :

**9.** Pour réduire les dimensions spatiales d’un facteur 2 :

un padding = 0 , un kernel de taille= 2x2 , stride =2

**10.** bias ? (rajouter le bias après confirmation)

Calculer les conv avec un padding qui vaut toujours 2 ?

|  |  |  |
| --- | --- | --- |
| Couches | Taille de sortie | Nombre de poids |
| Entrée taille  32 x 32 x 3 |  |  |
| —32 conv1 : convolutions 5×5, suivie de ReLU | 32x 32 x 32 | 5 x 5 x3 x 32 = |
| — pool1 : max-pooling 2×2 | 16x16x32 | 0 |
| — conv2 : 64 convolutions 5×5, suivie de ReLU | 16x16x64 | 5x5x32x64 |
| — pool2 : max-pooling 2×2 | 8x8x64 | 0 |
| — conv3 : 64 convolutions 5×5, suivie de ReLU | 8x8x64 | 5x5x64x64 |
| — pool3 : max-pooling 2×2 | 4x4 x 64 | 0 |
| — fc4 : fully-connected, 1000 neurones en sortie, suivie de ReLU | 1000 | 4x4 x 64 x 1000 |
| — fc5 : fully-connected, 10 neurones en sortie, suivie de softmax | 10 | 10 x 1000 |

**11.** Nombre total de poids : + + +

Nombre d’exemples 50k trop petit par rapport au nombre de paramètres, risque de overfitting

**12.** BoW 128 x 1001= 128000, 10 svms pour 10 classes = 10 parametres

## 2.2 Apprentissage du réseau

**14.**  La différence entre la façon de calculer la loss et l’accuracy en train et en test :

- Dans le test on ne calcule pas le gradient, on ne fait pas de backpropagation et donc on ne met pas à jour les poids du modèle

- Dans le test on utilise model.eval() au lieu de model.train()

**16. Effets sur la convergence du modèle**

**Pas d’apprentissage (Learning rate) : c’est un hyper-paramètre**

* Si le pas et trop petit, le modèle risque de mettre trop de temps à converger
* Si le pas est trop grand, risque de non convergence

**La taille de mini-batch**

batch size trop petit =mauvaise estimation du gradient + trop de bruit

trop grand = calacul long + la courbe moins bruité

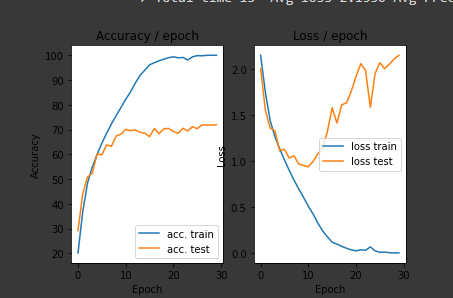
batch\_size —> inf: moins de bruit, donc meilleur gradient mais moins de généralisation

LR l'inverse

le lien entre  lr et batch size

**17.** L’erreur au début de la première époque correspond à un modèle pas encore entrainé et qui utilise des poids initialisés aléatoirement, les prédictions sont donc aléatoires

**18.**



**Figure 1** accuracy et loss de l’apprentissage du modèle sur le dataset CIFAR10

Nous remarquons que la loss du train diminue jusqu’à atteindre 0, cependant la loss du test diminue jusqu’à une certaine epoch ou elle remonte, on peut donc dire qu’il y a un **overfitting** (sur-apprentissage)

En ce qui concerne l’accuracy, nous remarquons que les deux accuracy du train et du test augmentent, mais à partir d’une certaine epoch, l’accuracy du train continue d’augmenter en revanche celle du test n’augmente plus et stagne.

# Partie 3 – Améliorations des résultats

## 3.1 Normalisation des exemples

19.

20. meme echelle -> ne pas faire de mauvais pas de backprop

Autre schémas : entre -1 et 1

https://jermwatt.github.io/control-notes/posts/zca\_sphereing/ZCA\_Sphereing.html

pas de random crop dans le test

on a pas augmenter le nombre de données, mais avec le crop et flip et on fais de la data augmentation en ajoutant de la robustesse

23. Non, chiffre, lettre

24.

33. dropout -> train ex 50% neurone desactiver , donc àa la sortie le input sera faible contrairemrnt à si on n’a pas de dropout la sortie sera grande vu qu’il n y a pas de 0