

# **Classification d'images satellites de champs**

Musab Karatas  
Mohammed Chahbaoui  
Thomas Hitchon

# Contexte et Problématique

## Contexte :

L'exploitation des données satellites est essentielle pour surveiller et optimiser les pratiques agricoles.

La classification des images satellitaires permet d'identifier les types de cultures sur des parcelles agricoles.

## Problématiques :

**Méthodologie** : Comment développer des modèles de classification?

**Classes déséquilibrées** : Quelle approche adopter face au déséquilibre des classes de cultures ?

**Choix du modèle** : Quels sont les meilleurs modèles ?

**Paramètres critiques** : Quels facteurs impactent les performances des modèles ?

# Description des données

Les données proviennent d'images satellites mensuelles sur 10 mois, une entrée du jeu de donnée correspond à:

- 10 images : une par mois. ( Février -> Novembre )
- avec 3 canaux spectraux : B08 proche infrarouge, B04 rouge, B03 vert
- dimension des images : 32x32 pixels
- la classe associée à la série d'image (le type de culture)

→ L'objectif est donc de les **classifier** en fonction du type de culture parmi 20 classes distinctes, telles que le blé tendre de printemps ou d'hiver, le maïs, ou encore la luzerne.



# Proposition d'approche



## Régression logistique

Modèle simple et interprétable, efficace pour la classification multiclasse avec des données tabulaires.

## Random Tree Forest

Résilient au surapprentissage, gère bien les déséquilibres de classes, permet d'ajuster le poids en fonction de la représentation des classes.

## CNN (Convolutional Neural Networks)

Puissant pour les données d'images, capable d'extraire des caractéristiques visuelles complexes.

# Protocole expérimental

1

**Préparation des données**

2

**Modélisation**

3

**Entraînement & évaluation**



# Régression logistique

# Régression logistique



## Préparation des données

Aplatition et standardisation des données pour une meilleure convergence et gestion du déséquilibre des classes avec `class_weights`.



## Modélisation

Création de deux modèles : un sans régularisation et un avec régularisation L2 ( $C=0.01$ ), trouvé via **GridSearchCV**.



## Entraînement & évaluation

Les deux modèles ont été entraînés avec l'algorithme **lbfgs**, puis évalués à l'aide de métriques (F1-score, précision, rappel, accuracy) et d'une matrice de confusion.

# Régression logistique

## - Présentation des résultats du modèle et discussions :

Accuracy : 0.62

0.63

	Modèle 1 sans régularisation l2			Modèle 2 avec régularisation l2		
	Précision	Rappel	F1-score	Précision	Rappel	F1-score
marco avg	0.43	0.44	0.42	0.47	0.46	0.45
weighted avg	0.71	0.62	0.63	0.73	0.63	0.64





# Random Tree Forest

# Random Tree Forest



## Préparation des données

Aplatissement des données. Pas besoin de normalisation.

Utilisation de `class_weights` pour équilibrer le jeu de données



## Modélisation

Utilisation de GridSearchCV: on a retenu 2 modèles:

- un modèle profonds complexe
- modèle limité en profondeur, plus simple, généralise plus



## Entraînement & évaluation

Comme pour régression logistique, plusieurs entraînements et plusieurs tests.

# Random Tree Forest (résultats)

- modèle 1: grande profondeur, capture relations complexes. Tend au surapprentissage
- modèle 2: capacité réduite, favorise généralisation et identifie mieux les instances de la classe minoritaire, ce qui réduit les faux négatifs

	RandomForest Modèle 1			RandomForest Modèle 2		
	Précision	Rappel	F1-score	Précision	Rappel	F1-score
marco avg	0.53	0.45	0.45	0.53	0.47	0.46
weighted avg	0.72	0.71	0.69	0.75	0.73	0.71



# **Convolutional Neural Network (CNN)**

# Réseaux de convolution



## Préparation des données

Transposition et normalisation min max des données.



## Modélisation

Deux modèles aux architectures similaires, le second utilisant des méthodes de régularisation.



## Entraînement & évaluation

Optimiser Adam, sparse\_categorical\_crossentropy, early stopping sur le second modèle.

# Réseaux de convolution

## Architectures des Modèles :

Conv3D(32, (3,3,3), relu)

MaxPooling3D

Conv3D(64, (3,3,3), relu)

MaxPooling3D

Dense(128, relu)

Dense(20, softmax)

## Ajouts pour éviter l'overfit :

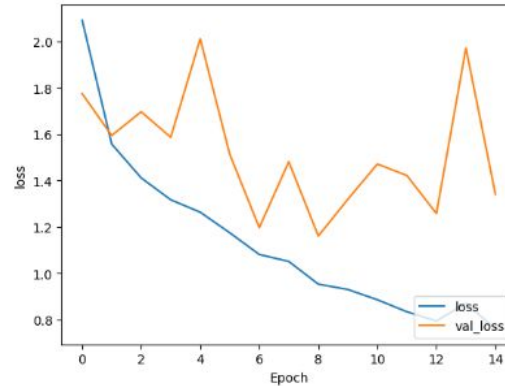
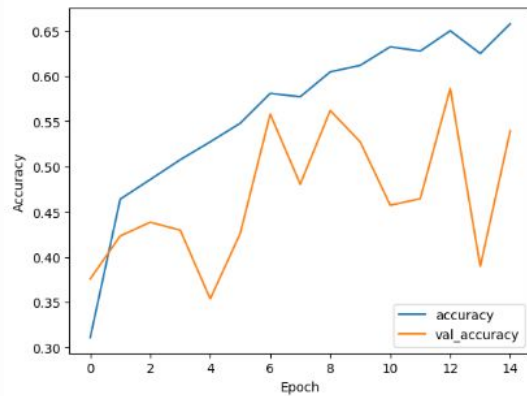
Normalisation l2, C=0.01

BatchNormalization après Pooling

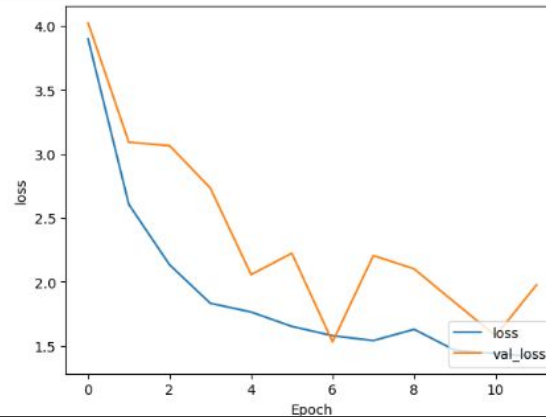
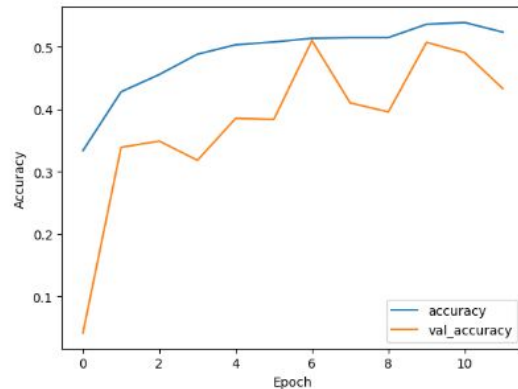
Dropout(0.5) entre les couches

Denses

## Modèle simple



## Modèle avec les ajouts



# Résultats des CNN

	CNN Modèle 1			CNN Modèle 2		
	Précision	Rappel	F1-score	Précision	Rappel	F1-score
marco avg	0.45	0.48	0.39	0.43	0.39	0.34
weighted avg	0.72	0.54	0.57	0.73	0.43	0.48



# Modèles avec pondération sur les mois

- Test des mois indépendamment deux par deux pour identifier les mois les plus discriminants
- Attribution de poids plus importants aux mois discriminants lors de l'entraînement et les tests
- Régression logistique et CNN

# Résultats avec pondération sur les mois

- Mois les plus discriminants: Avril-Mai
- Résultats insatisfaisants pour les 2 modèles
- 2 théories sur l'échec:
  - Overfitting
  - Mauvaise implémentation

# Conclusion

**Meilleurs scores globaux** : modèle Random Forest 2 (le moins profond) weighted avg f1-score et accuracy les plus élevés.

**Modèles les plus robustes** : régression logistique avec régularisation et les deux Random Forest car ils ont la meilleure prise en compte des classes déséquilibrées (macro-avg : 0.45-0.46).