

# OPTICAL CHARACTER RECOGNITION

OCR POUR L'EXTRACTION AUTOMATIQUE INFORMATIONS DE TICKETS DE CAISSES

CURE51 - GUILLAUME MACQUART DE TERLINE

# S O M M A I R E

01 Objectif

02 Présentation du Dataset

03 Enjeux du Dataset Considéré

04 Pipeline de prétraitement des données

05 Détection des bounding boxes

06 Reconnaissance de caractères

07 Performances du modèle

08 Limites et pistes d'amélioration

# 01. OBJECTIF

## **Objectif:**

Construire un pipeline de pré-traitement capable de nettoyer des images de reçus pour en extraire le montant total via un moteur OCR standard.

## **Contraintes:**

Interdiction d'utiliser des modèles de Deep Learning "boîte noire" pré-entraînés spécifiquement pour les reçus (type LayoutLM ou APIs type Azure Form Recognizer). Utilisez des méthodes de Computer Vision classiques ou des backbones généralistes type VGG ou ResNet.

## **Lien Github du projet:**

<https://github.com/Gdeterline/Optical-Character-Recognition-for-Receipts>

# 02. PRÉSENTATION DU DATASET

Contenu du dataset:

- **images**: 200 images, divisées en 2 ensembles:
  - 100 images “dev\_”
  - 100 images “test\_”
- **metadata.pkl**: fichier contenant les annotations pour l’ensemble des images
  - **file\_name**: nom du fichier associé
  - **split\_origin**: dev ou test
  - **words**: liste de mots (et nombres/montants) présents dans le ticket de caisse
  - **bboxes**: coordonnées des “bounding boxes”/zones de texte, pour chaque mot
  - **full\_text**: l’ensemble des mots du ticket de caisse



# 03. ENJEUX DU DATASET CONSIDÉRÉ

- **Nombre d'images restreint (100 images "dev\_")**

Nous ne considérons pas les images "test\_" (hormis comme ensemble de test!)

- **Tailles des images différentes**

- **Contenu des images diversifié:**

- Reçus de formes différentes
- Arrière-plans et angles de vue (perspective, rotations, etc.) inhomogènes
- Polices d'écriture différentes
- Zones d'ombre/Manque de contraste

- **Données bruitées:** Marquages sur les reçus / Tickets de caisse froissés dans certains cas

Examples of challenging receipt images in the dataset

dev\_receipt\_00002.png  
skewed/rotated



dev\_receipt\_00016.png  
notes/markings - challenging background



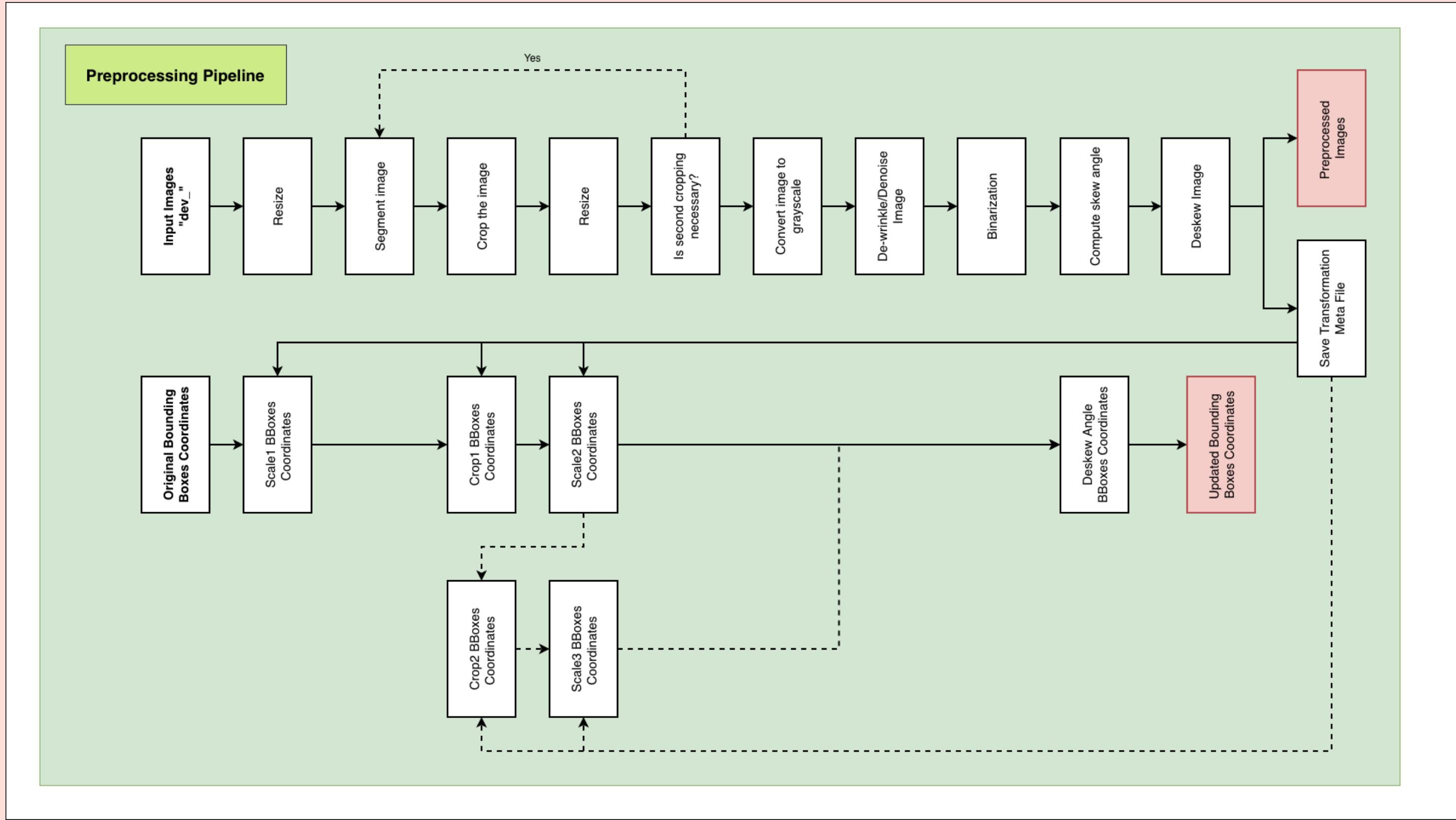
dev\_receipt\_00026.png  
folds/creases



dev\_receipt\_00043.png  
poor lighting



# 04 . PIPELINE DE PRÉTRAITEMENT DES DONNÉES



# 04 . PIPELINE DE PRÉTRAITEMENT DES DONNÉES

- **Fonction Resize:**

- Calcule la hauteur et la largeur médianes du dataset “dev\_”
- Arrondi ces valeurs au multiple de 32 le plus proche (afin de conserver l’aspect tout en se ramenant à une valeur divisible plusieurs fois par 2)
- Resize des images selon ces valeurs, en gardant la meilleure résolution possible

- **Segmentation des images:**

- Soit K-Means, soit GMM: GMM fait moins d’hypothèses sur la structure des données + soft-clustering
- n\_clusters = 2, afin de détacher l’avant-plan (supposé être le ticket de caisse) de l’arrière-plan
- En cas de second cropping, incrémentation du nombre de clusters à 3

- **Cropping des clusters selon le critère:**

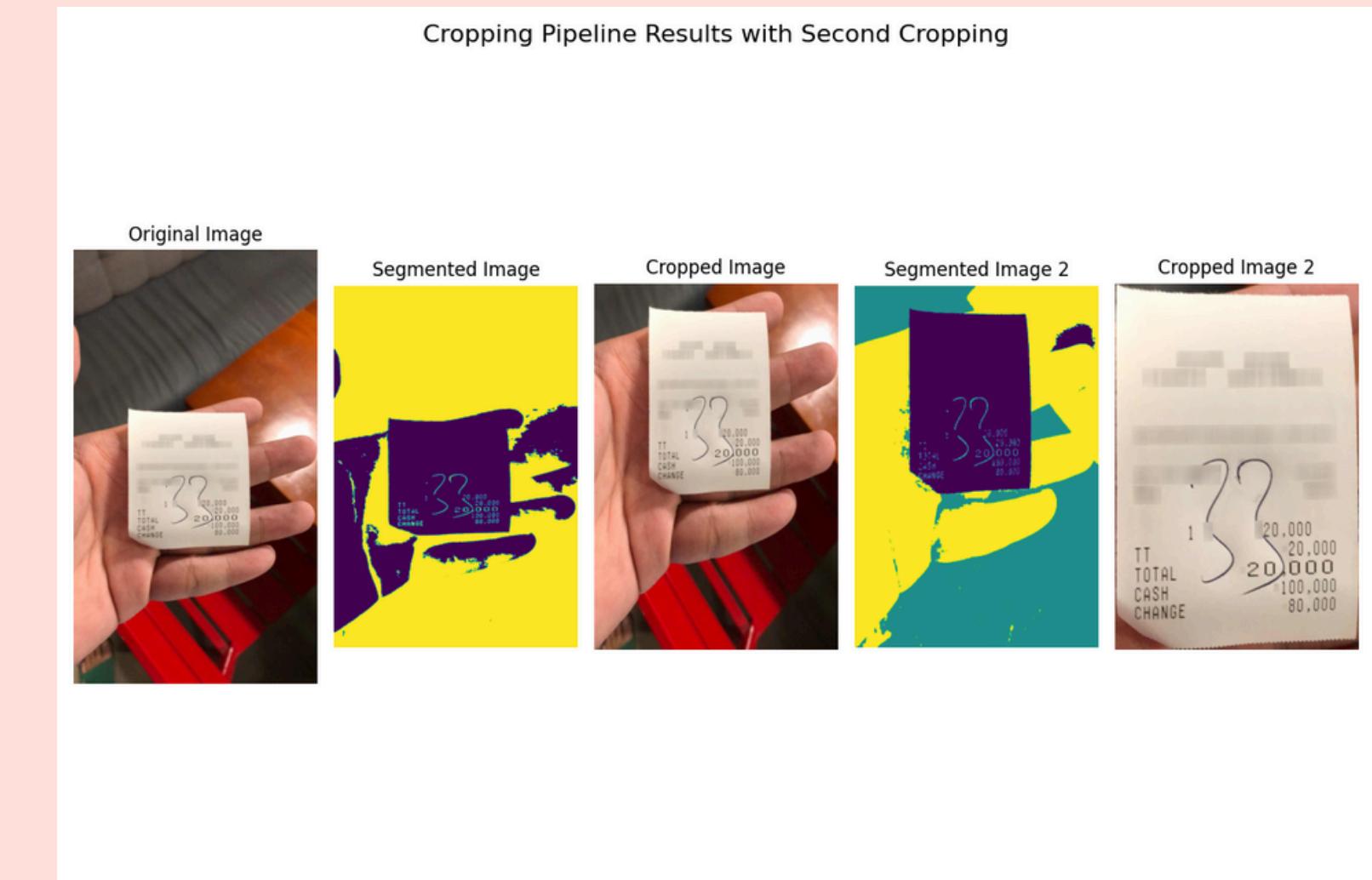
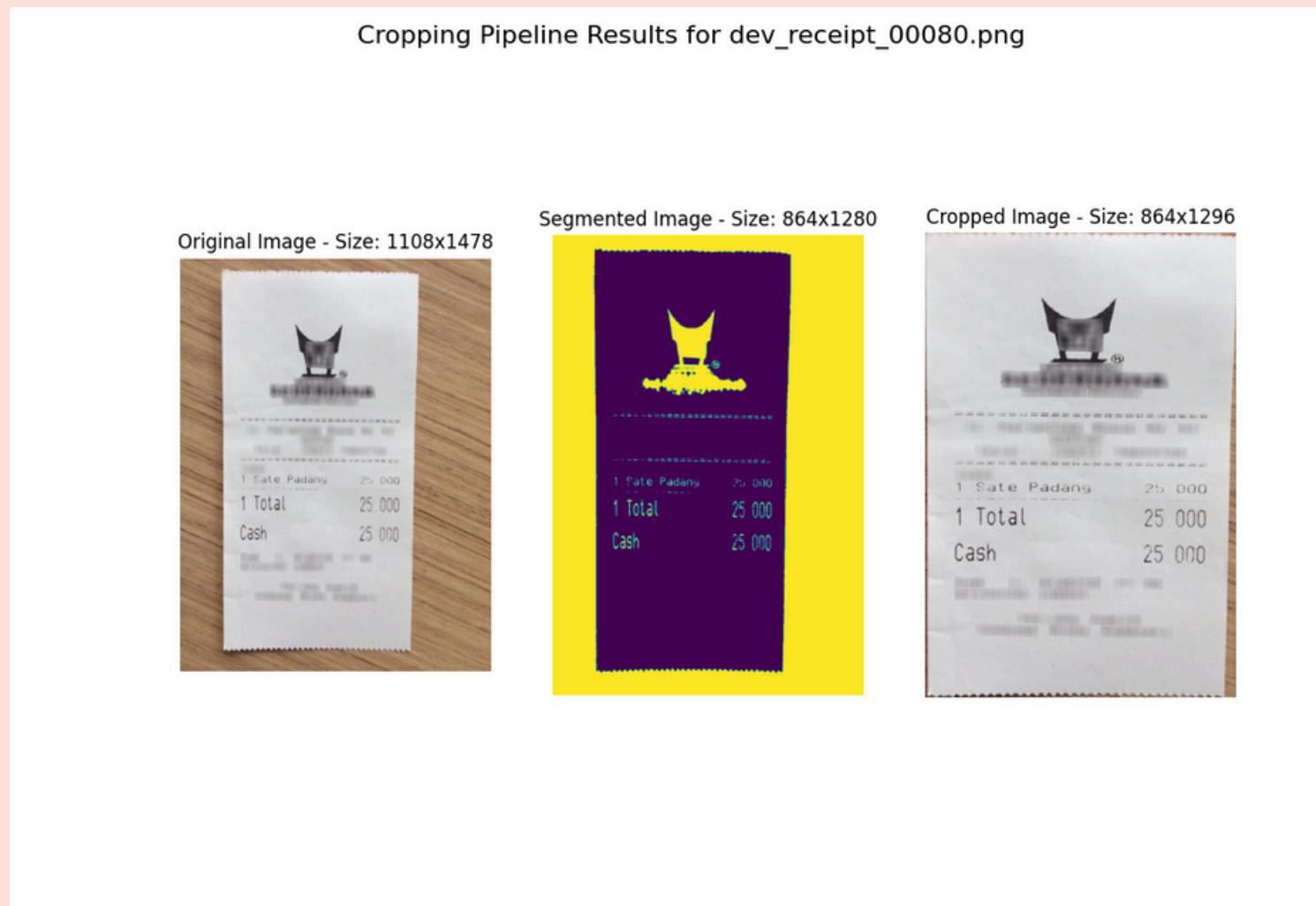
- Cluster le plus grand
- Cluster le plus central → meilleures performances

- **Second cropping nécessaire?**

- Segmentation avec GMM de l’image (n\_clusters = 2)
- Basé sur la proportion de l’image recouverte par le cluster le plus grand
  - $\geq 85\%$ : pas de second cropping
  - $< 85\%$ : second cropping nécessaire
- Seuil de 85% défini de manière empirique

# 04. PIPELINE DE PRÉTRAITEMENT DES DONNÉES

## CROPPING DES ARRIÈRE-PLANS



# 04. PIPELINE DE PRÉTRAITEMENT DES DONNÉES

- **Image Converties en Grayscale**

- **De-wrinkling/Denoising des Images:**

- Opération Morphologique de Fermeture (Dilatation puis Erosion) sur l'image pour créer un arrière-plan
  - On définit un kernel (matrice de taille 8×8) qui "glisse" à travers l'image, afin de capturer les variations locales d'intensité
  - Taille 8×8 choisie empiriquement, pour être > taille des caractères mais < taille des variations d'éclairage
  - **Étape 1 - Dilatation:** Remplace chaque pixel par le maximum dans le voisinage 8×8: les zones claires s'étendent et le texte (sombre) disparaît
  - **Étape 2 - Érosion:** Remplace chaque pixel par le minimum dans le voisinage 8×8: restaure les formes globales du fond
- On divise l'image originale par son arrière plan, afin d'obtenir une image où les plis, le bruit (logos, etc.) et les zones d'ombres ont été atténus

- **Mise à l'échelle et binarization:**

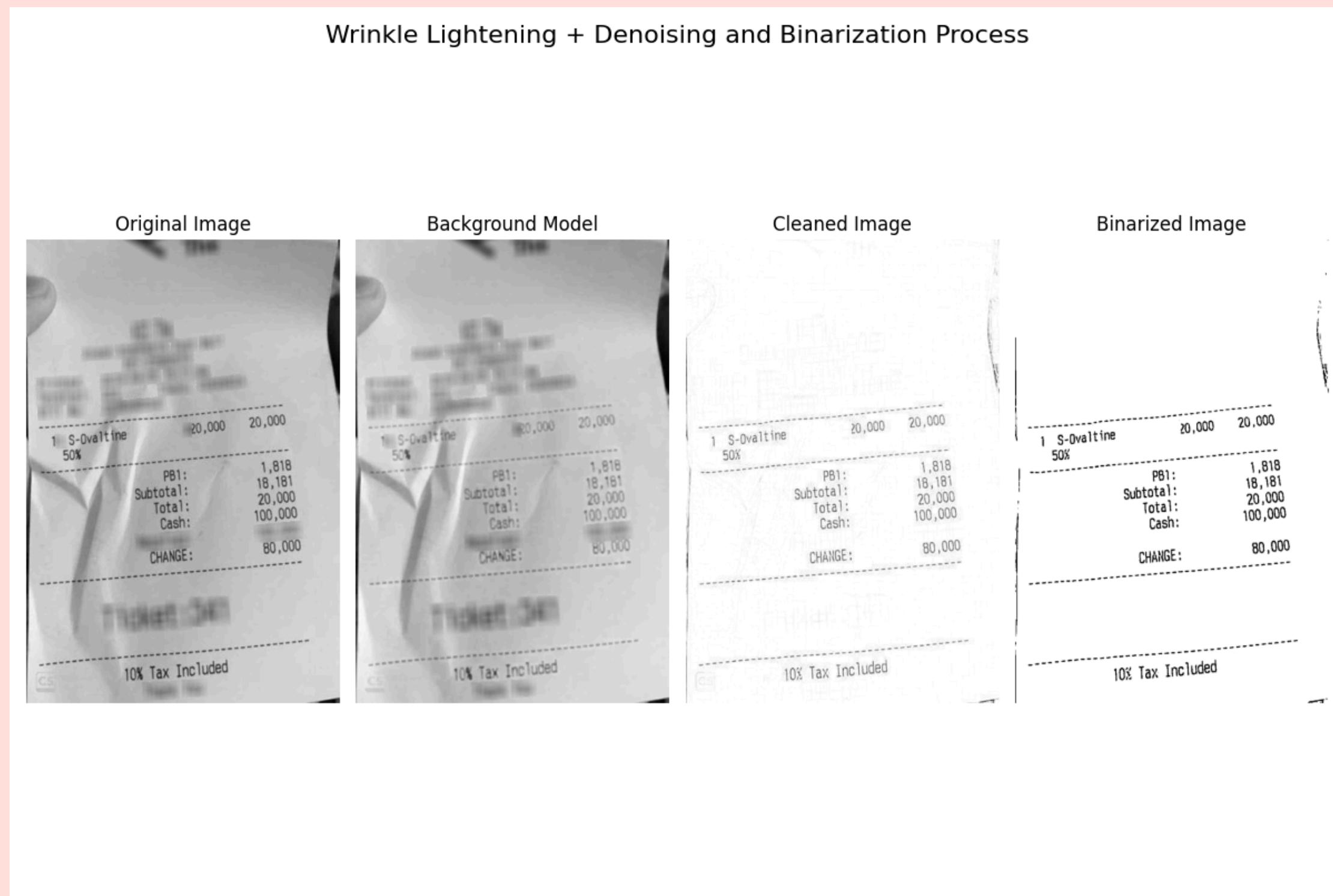
- On multiplie les pixels par 255 pour les ramener dans une plage de valeur de [0, 255]
- Binarization de l'image en noir et blanc pur (0 ou 255) avec le threshold d'Otsu (calcul automatique du seuil optimal)

- **Calcul de l'angle de rotation (skew) et redressement (deskewing) des images**

- Inversion des pixels (texte blanc sur fond noir)
- Dilatation horizontale (kernel 20x1) afin de connecter les caractères d'une même ligne de texte → bandes horizontales
- Pour chaque bande, on estime les contours, et on calcule le plus petit rectangle les englobant. On extrait son angle de rotation.
- On retourne la médiane de ces angles
- **Filtrage:** angles > 45° ignorés, formes verticales ignorées, petits contours ignorés
- Calcul de la matrice de rotation et application à l'image

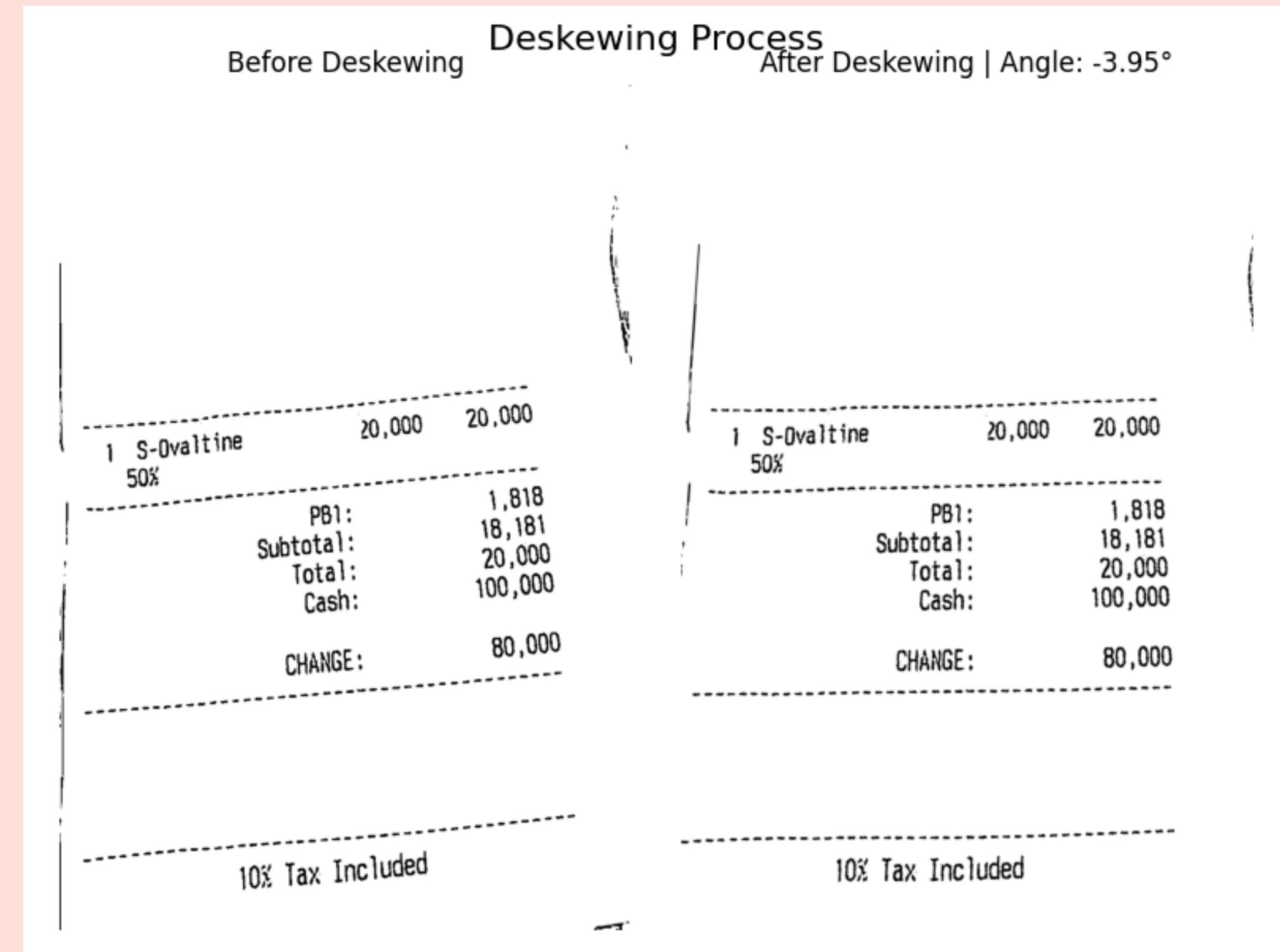
# 04. PIPELINE DE PRÉTRAITEMENT DES DONNÉES

## DÉBRUITAGE DES DONNÉES ET BINARIZATION



# 04. PIPELINE DE PRÉTRAITEMENT DES DONNÉES

## DESKewing DES DONNÉES

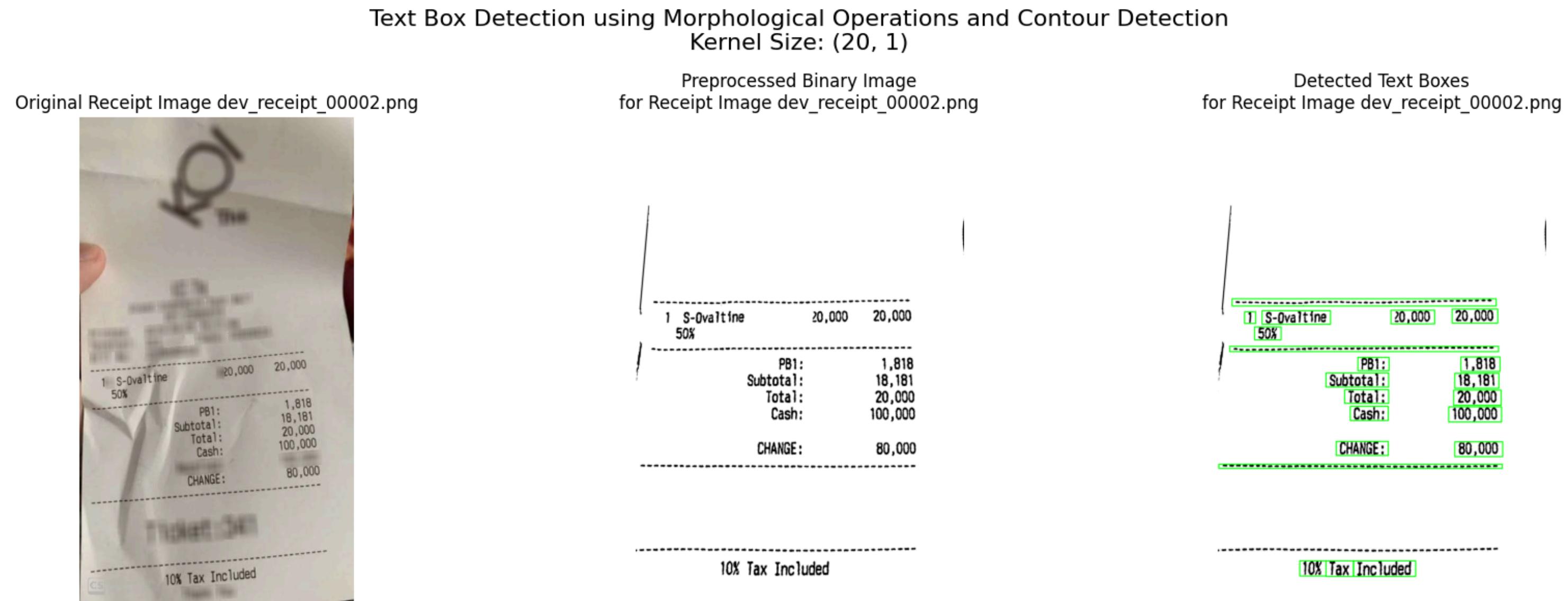


## 05 . DÉTECTION DES BOUNDING BOXES

- **L'Algorithme de détection de zones de texte** repose sur le même principe que celui pour le calcul de l'angle de rotation de l'image.
- **Inversion des pixels** (text blanc sur fond noir)
- **Dilatation horizontale** (kernel 20x1) afin de connecter les caractères d'une même ligne de texte → bandes horizontales
  - Kernel 20x1 défini empiriquement
- **Détection des régions dilatées et calcul du rectangle englobant**, pour chaque région.
- **Filtrage:**
  - aire < 400 pixels → bruit (points, artefacts)
  - hauteur > 15% de l'image → probablement bordure verticale et non du texte
  - exclusion des bords de l'image

# 05. DÉTECTION DES BOUNDING BOXES

## EXEMPLE 1 / 3



# 05. DÉTECTION DES BOUNDING BOXES

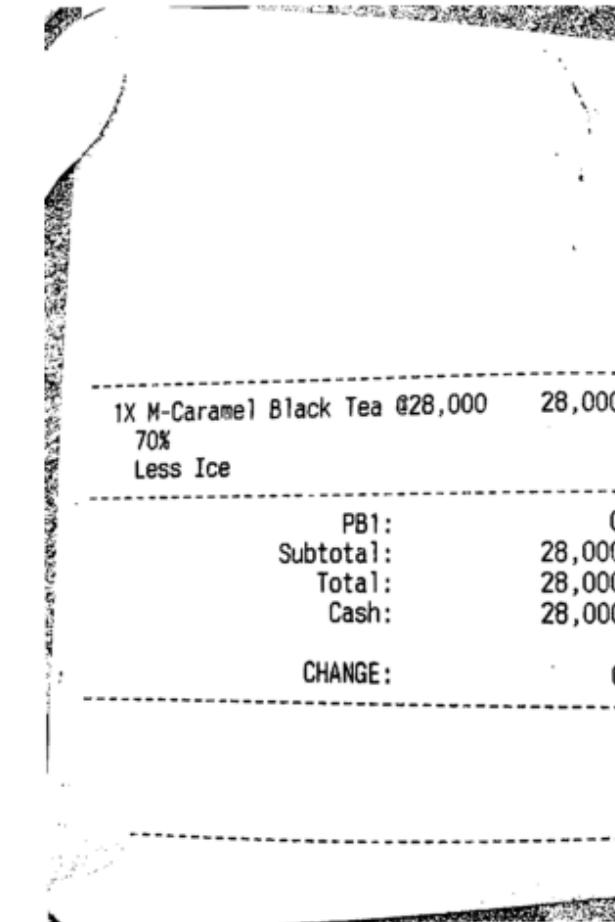
## EXEMPLE 2 / 3

Text Box Detection using Morphological Operations and Contour Detection  
Kernel Size: (20, 1)

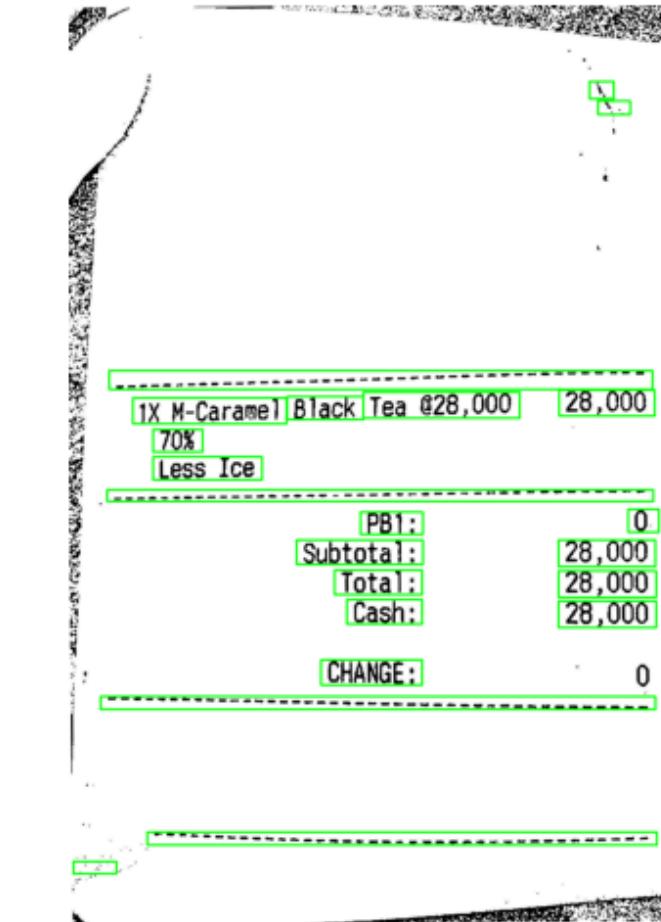
Original Receipt Image dev\_receipt\_00003.png



Preprocessed Binary Image  
for Receipt Image dev\_receipt\_00003.png

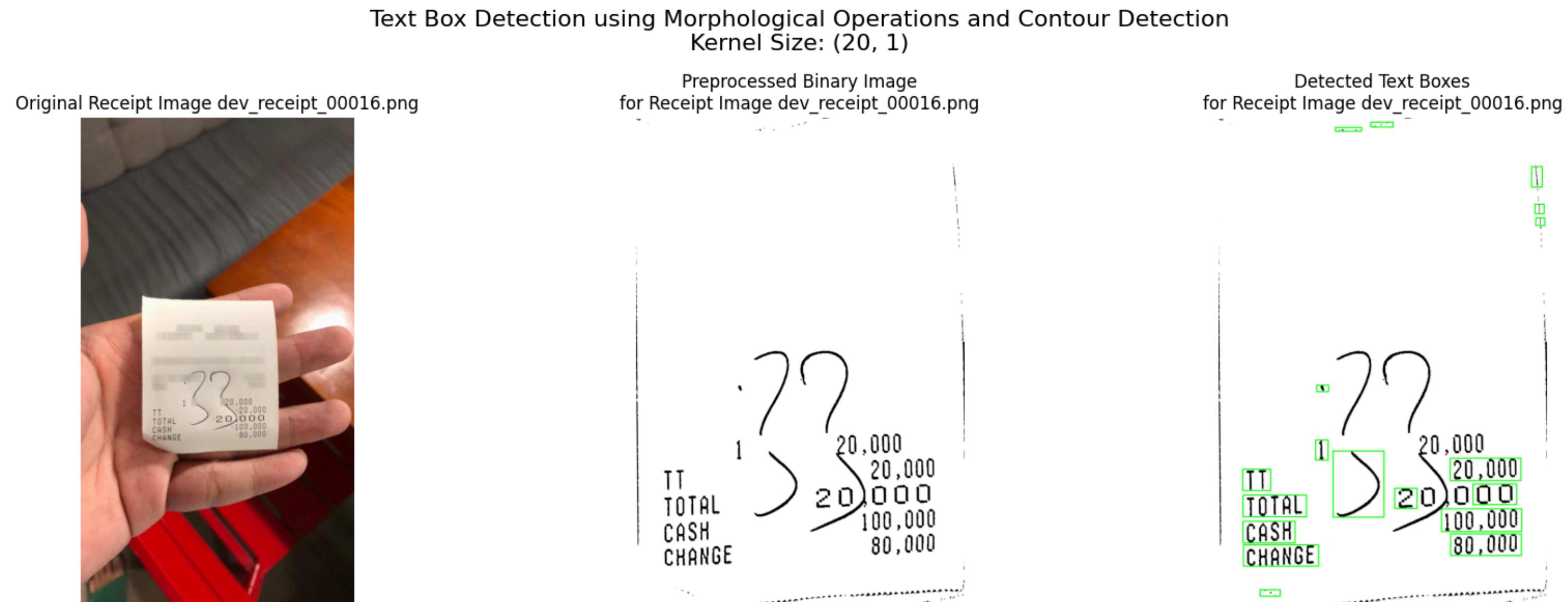


Detected Text Boxes  
for Receipt Image dev\_receipt\_00003.png



# 05. DÉTECTION DES BOUNDING BOXES

## EXEMPLE 3 / 3



## **06. RECONNAISSANCE DE CARACTÈRES**

### **DÉCOUPAGE DES TICKETS DE CAISSE EN MOTS**

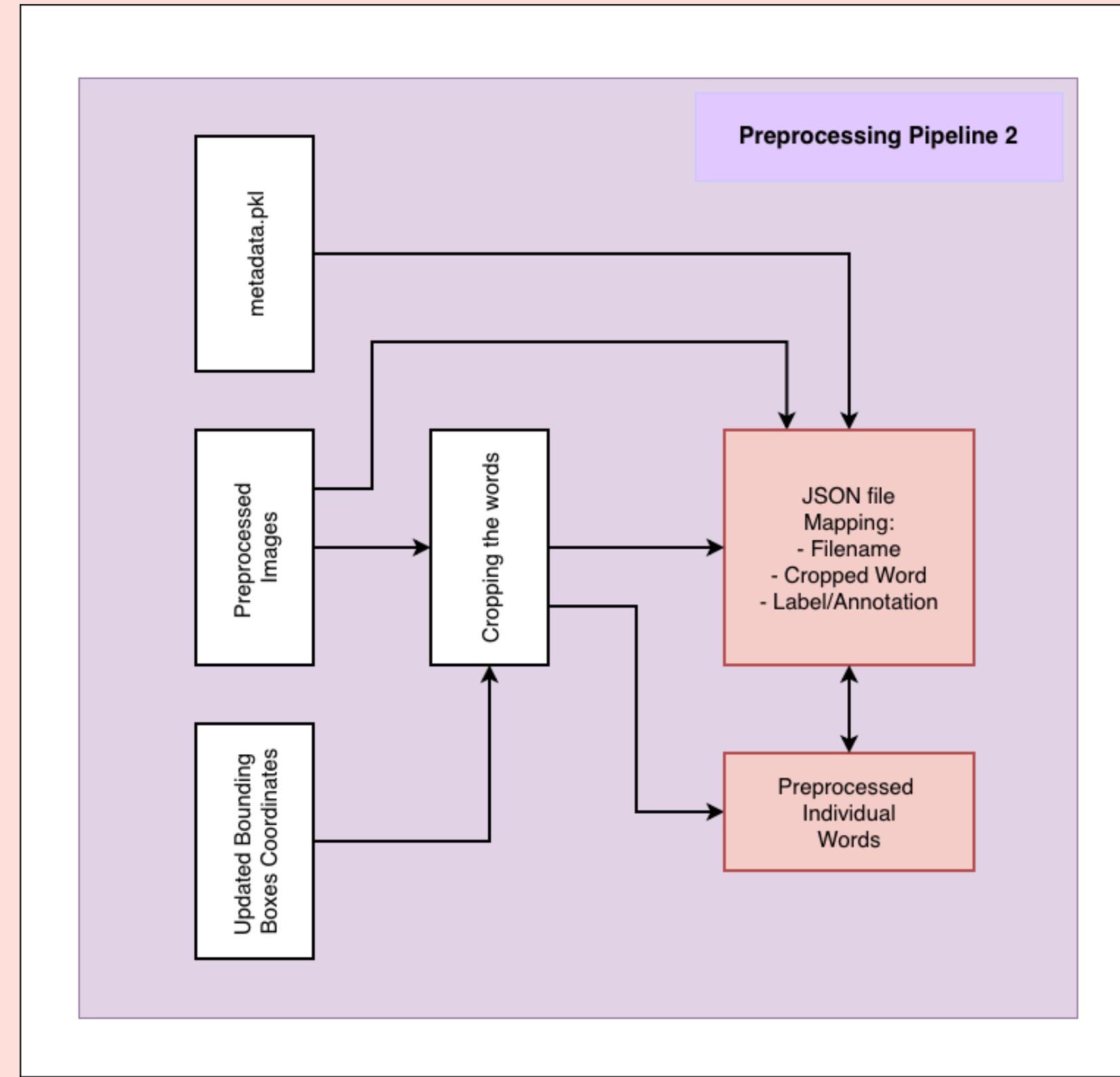
**Mais la taille du jeu de données reste excessivement faible pour entraîner n'importe quel modèle de deep learning...**

Pour rappel, nous avons seulement 100 images “dev\_”

**Solution:** Considérer les mots individuellement pour le modèle de reconnaissance

# 06. RECONNAISSANCE DE CARACTÈRES

Nombre de mots:  
2186



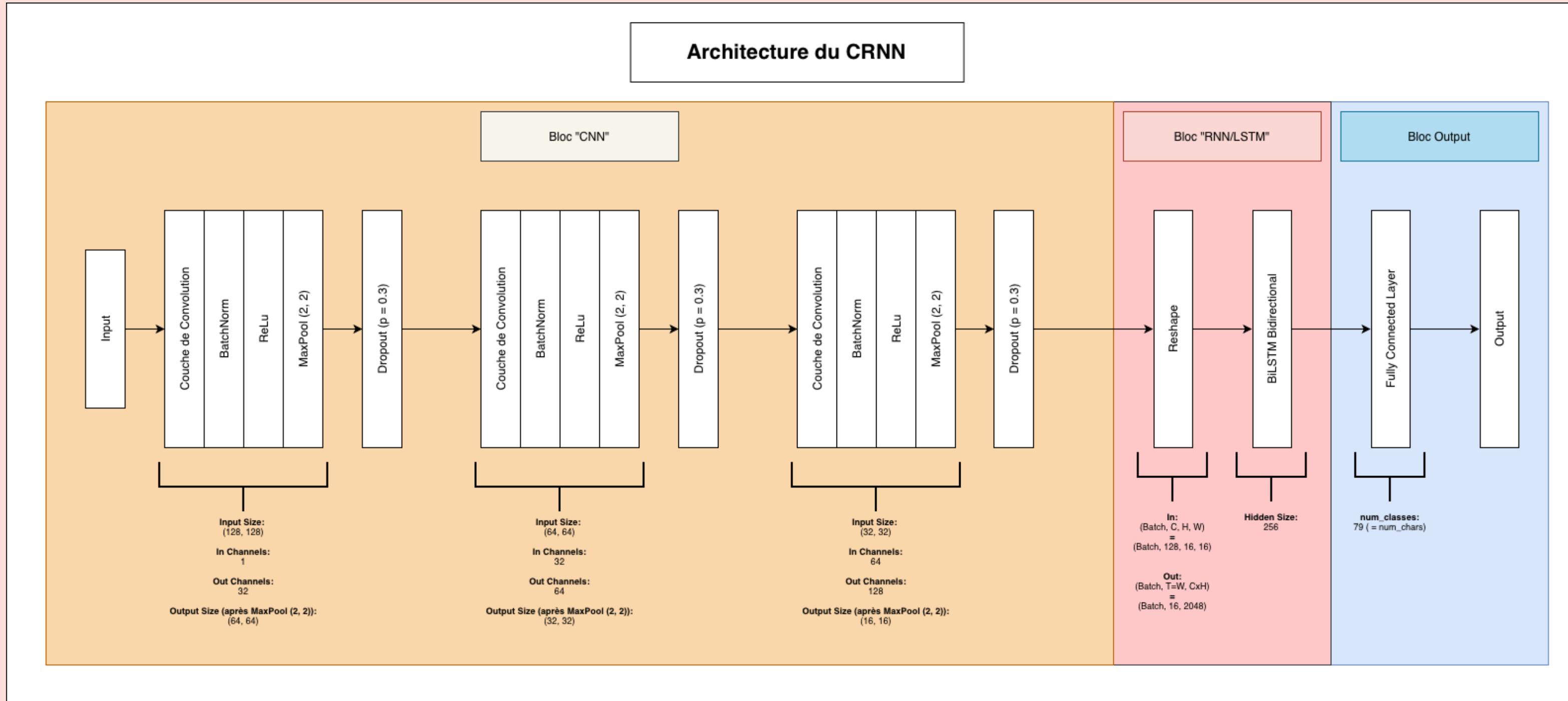
# 06 . RECONNAISSANCE DE CARACTÈRES

## DATASET ET MODÈLES CONSIDÉRÉS

- **Division du dataset** (cropped word image, label) en jeux d'entraînement et validation:
  - ~85% pour l'entraînement
  - ~15% pour la validation
- **Stratification faite sur les reçus** (pour éviter toute fuite de données). On stratifie les reçus en fonction de leur type de mot majoritaire (alphabétique, numérique, alphanumérique (alpha + num), autre).
- **Trois modèles ont été entraînés puis évalués:**
  - Architecture CRNN définie en slide 19 (sans BatchNorm, sans Dropout, sans Weight Decay) → mode collapse
  - Architecture CRNN définie en slide 19 (avec BatchNorm, sans Dropout, sans Weight Decay)
  - Architecture CRNN définie en slide 19 (avec BatchNorm, avec Dropout, avec Weight Decay)
- **CTC Loss**
- **Optimizer Adam (Learning Rate: 5e-4, Weight Decay: 1e-4)**
- **Batch size: 16**
- **Nombre d'epochs: 80**
- **Early Stopping**
- **Note:** L'algorithme de détection étant très sensible au bruit, nous avons utilisé les coordonnées des bounding boxes présentes dans le fichier metadata.pkl lors de l'entraînement, **et lors de l'inférence**

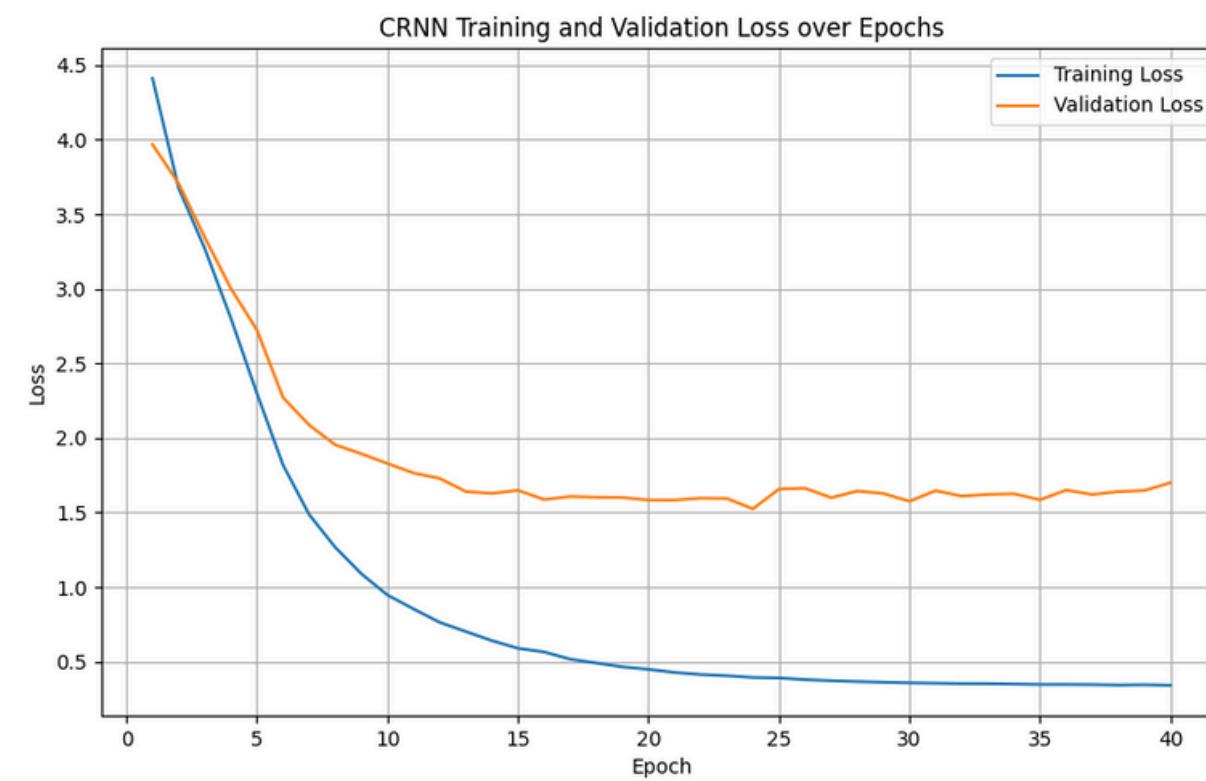
# 06. RECONNAISSANCE DE CARACTÈRES

## ARCHITECTURE DU CRNN FINAL

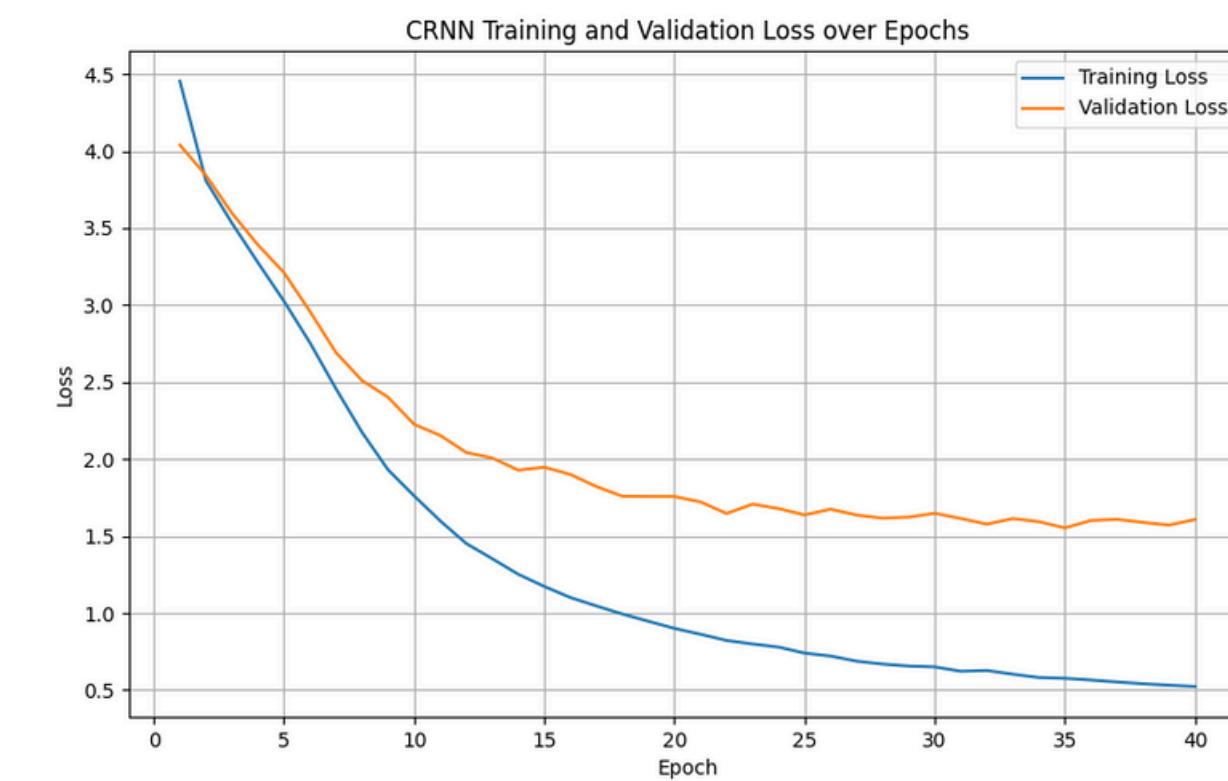


# 07. PERFORMANCES DU MODÈLE

CRNN "vanilla", sans Weight Decay ni Dropout



CRNN avec Weight Decay et Dropout



Configuration du CRNN	Dataset	CER	Word Accuracy
"Vanilla"	Train + Val	0.13097513952957687	0.8183897529734675
"Vanilla"	Test	0.27819552468436537	0.5050933786078099

Configuration du CRNN	Dataset	CER	Word Accuracy
Weight Decay = 1e-4 & Dropout = 0.3	Train + Val	0.14309295842918898	0.7703568161024703
Weight Decay = 1e-4 & Dropout = 0.3	Test	0.2727942858892692	0.515704584040747

# 07. PERFORMANCES DU MODÈLE

## GROUND TRUTHS VS. PREDICTION - CER - WORD ACCURACY

1 / 3



Item Description	Quantity	Unit Price	Total Price
1 [LG] BLACK SAKURA	1	59,091	59,091
1 LONGAN	1	0	0
1 ROASTED ALMOND	1	0	0
1 MANGGO	1	0	0
1 MINERAL WATER (bundling) 5k	1	4,545	4,545
<b>Sub Total</b>		63,636	63,636
PB1 (10%)		6,364	6,364
Rounding		0	0
<b>Total</b>		70,000	70,000
Cash Payment		100,000	100,000
Change		30,000	30,000

receipt_filename	ground_truth	prediction	cer	word_accuracy
test_receipt_00016.png	1	1	0.00000	1.0
test_receipt_00016.png	[LG]	IL0	0.75000	0.0
test_receipt_00016.png	BLACK	BLACK	0.00000	1.0
test_receipt_00016.png	SAKURA	SAKURA	0.00000	1.0
test_receipt_00016.png	59,091	59,091	0.00000	1.0
test_receipt_00016.png	1	1	0.00000	1.0
test_receipt_00016.png	LONGAN	LNGAN	0.16667	0.0
test_receipt_00016.png	0	0	0.00000	1.0
test_receipt_00016.png	1	1	0.00000	1.0
test_receipt_00016.png	ROASTED	ROASTED	0.00000	1.0
test_receipt_00016.png	ALMOND	ALNND	0.33333	0.0
test_receipt_00016.png	0	0	0.00000	1.0
test_receipt_00016.png	1	1	0.00000	1.0
test_receipt_00016.png	IVANGGO	VANNGG0	0.42857	0.0
test_receipt_00016.png	0	0	0.00000	1.0
test_receipt_00016.png	1	1	0.00000	1.0
test_receipt_00016.png	IVINERAL	NNERAL	0.25000	0.0
test_receipt_00016.png	WATER	AYATER	0.40000	0.0
test_receipt_00016.png	(bundling)	6bunding	0.30000	0.0
test_receipt_00016.png	5k	S5x	1.00000	0.0
test_receipt_00016.png	4,545	4,545	0.00000	1.0
test_receipt_00016.png	Sub	Sub	0.00000	1.0
test_receipt_00016.png	Total	Total	0.00000	1.0
test_receipt_00016.png	63,636	63,636	0.00000	1.0
test_receipt_00016.png	PB1	PB1	0.00000	1.0
test_receipt_00016.png	(10%)	(1096	0.60000	0.0
test_receipt_00016.png	6,364	6,364	0.00000	1.0
test_receipt_00016.png	Rounding	Rounoing	0.12500	0.0
test_receipt_00016.png	0	0	0.00000	1.0
test_receipt_00016.png	Total	Total	0.00000	1.0
test_receipt_00016.png	70,000	70,000	0.00000	1.0
test_receipt_00016.png	Cash	Cash	0.00000	1.0
test_receipt_00016.png	Payment	Paynent	0.14285	0.0
test_receipt_00016.png	100,000	100,000	0.00000	1.0
test_receipt_00016.png	Change	Change	0.00000	1.0
test_receipt_00016.png	30,000	30,000	0.00000	1.0

**CER Moyen:**

0.125

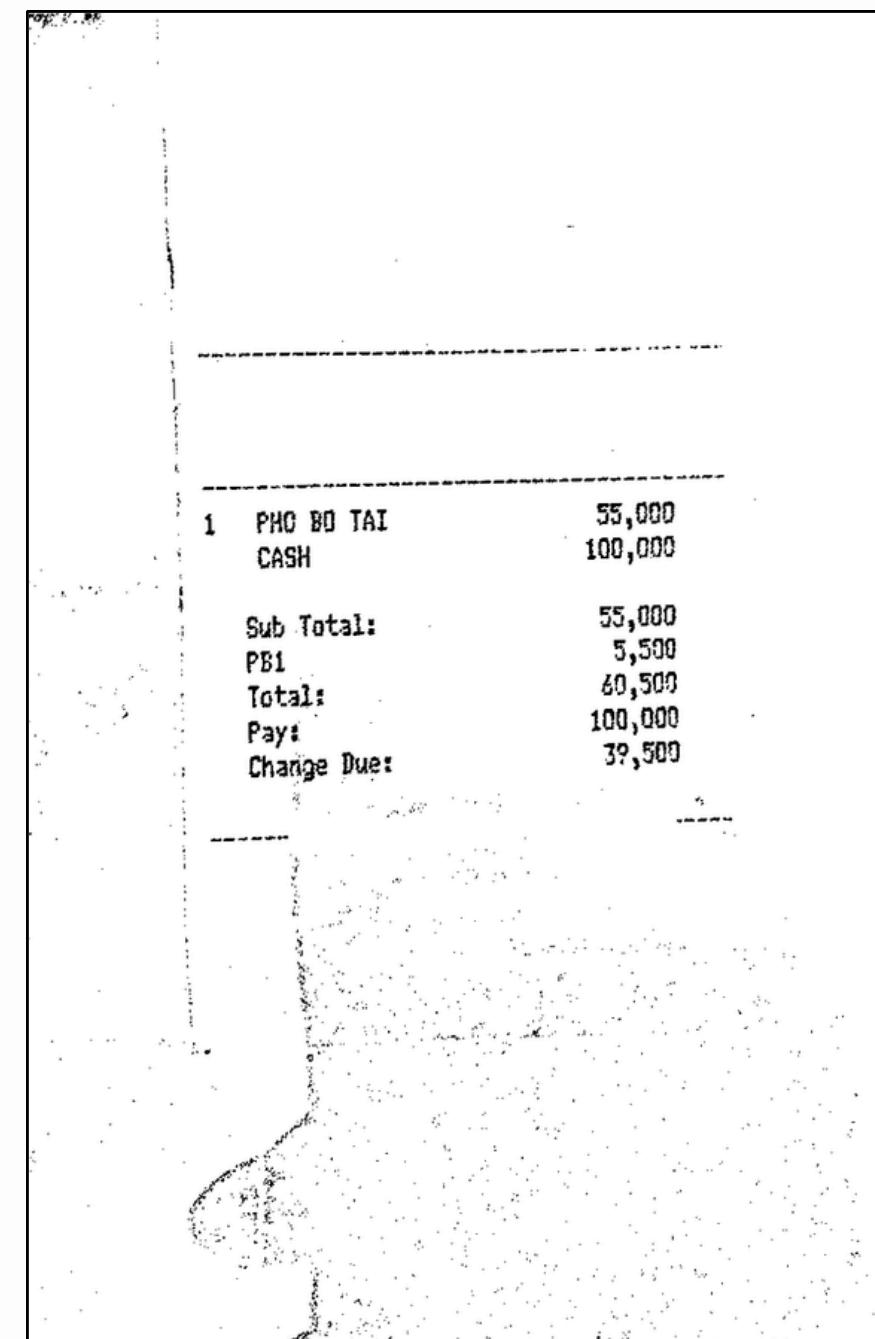
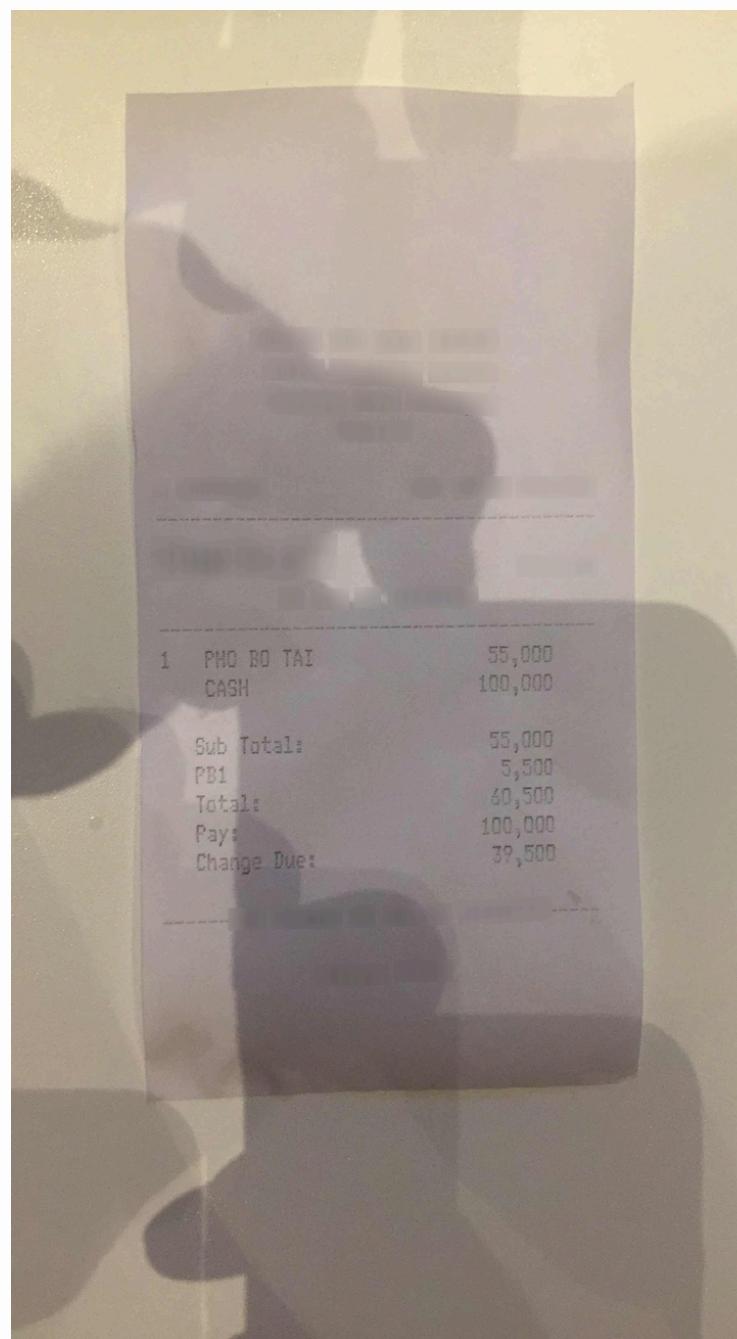
**Word Accuracy Moyenne:**

0.694

# 07. PERFORMANCES DU MODÈLE

## GROUND TRUTHS VS. PREDICTION - CER - WORD ACCURACY

2 / 3



receipt_filename	ground_truth	prediction	cer	word_accuracy
test_receipt_00058.png	Pay:	Pay	0.250000	0.0
test_receipt_00058.png	100,000	100,000	0.000000	1.0
test_receipt_00058.png	1	1	0.000000	1.0
test_receipt_00058.png	PHO	PO	0.666667	0.0
test_receipt_00058.png	BO	BO	0.500000	0.0
test_receipt_00058.png	TAI	TAI	0.000000	1.0
test_receipt_00058.png	55,000	55,000	0.000000	1.0
test_receipt_00058.png	Sub	Sub	0.000000	1.0
test_receipt_00058.png	Total:	Total:	0.000000	1.0
test_receipt_00058.png	55,000	55,000	0.000000	1.0
test_receipt_00058.png	PB1	PB1L	0.333333	0.0
test_receipt_00058.png	5,500	5,500	0.000000	1.0
test_receipt_00058.png	Total:	Total:	0.000000	1.0
test_receipt_00058.png	60,500	60,500	0.000000	1.0
test_receipt_00058.png	CASH	CASH	0.000000	1.0
test_receipt_00058.png	100,000	100,000	0.000000	1.0
test_receipt_00058.png	Change	Chamge	0.166667	0.0
test_receipt_00058.png	Due:	Due:	0.000000	1.0
test_receipt_00058.png	39,500	39,500	0.000000	1.0

CER Moyen:

0.101

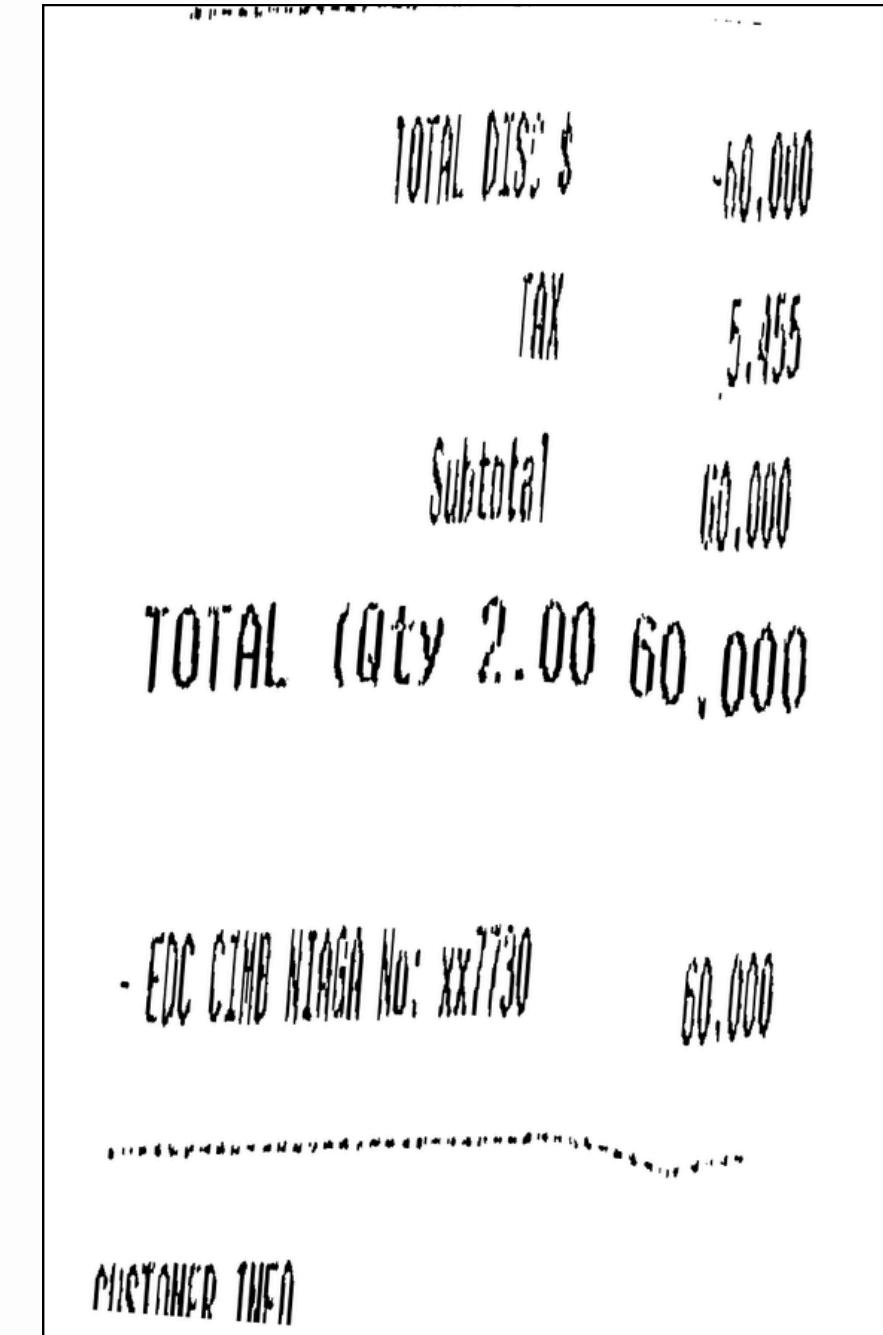
Word Accuracy Moyenne:

0.737

# 07. PERFORMANCES DU MODÈLE

## GROUND TRUTHS VS. PREDICTION - CER - WORD ACCURACY

### 3 / 3



receipt_filename	ground_truth	prediction	cer	word_accuracy
test_receipt_00000.png	TAX	TAX 0.00000	1.0	
test_receipt_00000.png	5.455	5.455 0.20000	0.0	
test_receipt_00000.png	TOTAL	TOTAL 0.00000	1.0	
test_receipt_00000.png	60.000	60.000 0.00000	1.0	
test_receipt_00000.png	(Qty	ctou 1.00000	0.0	
test_receipt_00000.png	2.00	2.00 0.00000	1.0	
test_receipt_00000.png	EDC	EDC 0.00000	1.0	
test_receipt_00000.png	CIMB	C1NB 0.50000	0.0	
test_receipt_00000.png	NIAGA	NIAGA 0.00000	1.0	
test_receipt_00000.png	No:	MNo: 0.333333	0.0	
test_receipt_00000.png	xx7730	730 0.50000	0.0	
test_receipt_00000.png	60.000	60.000 0.00000	1.0	
test_receipt_00000.png	901016	1 0.833333	0.0	
test_receipt_00000.png	-TICKET	1 1.00000	0.0	
test_receipt_00000.png	CP	1 1.00000	0.0	
test_receipt_00000.png	2	1 1.00000	0.0	
test_receipt_00000.png	60.000	1 1.00000	0.0	
test_receipt_00000.png	60.000	1 1.00000	0.0	
test_receipt_00000.png	Subtotal	Subtotal 0.00000	1.0	
test_receipt_00000.png	60.000	00.000 0.166667	0.0	
test_receipt_00000.png	TOTAL	TOTAL 0.00000	1.0	
test_receipt_00000.png	DISC	DISC 0.00000	1.0	
test_receipt_00000.png	\$	NaN 1.00000	0.0	
test_receipt_00000.png	-60.000	-60.000 0.00000	1.0	

**CER Moyen:**

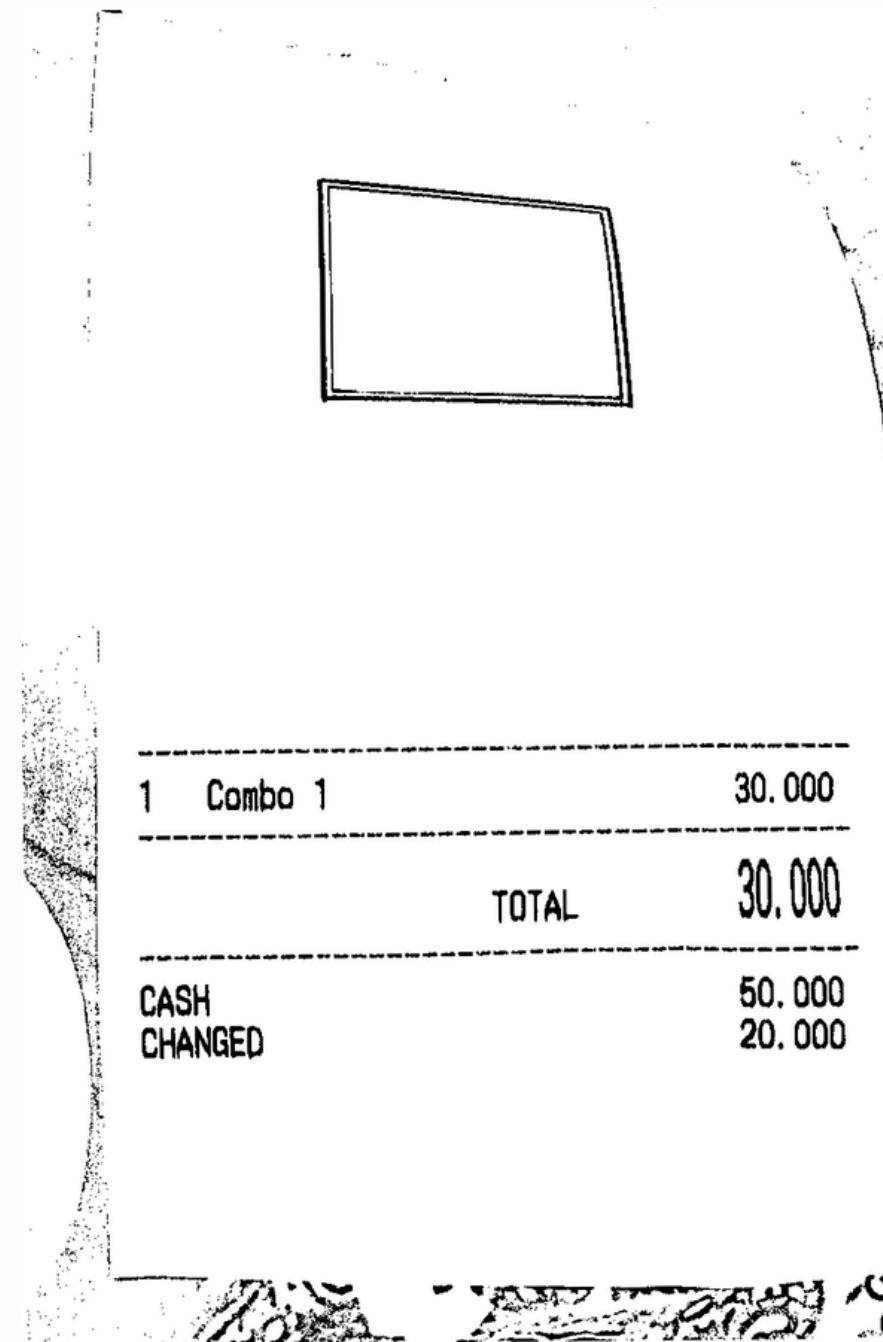
0.397

**Word Accuracy Moyenne:**

0.458

# 07. PERFORMANCES DU MODÈLE

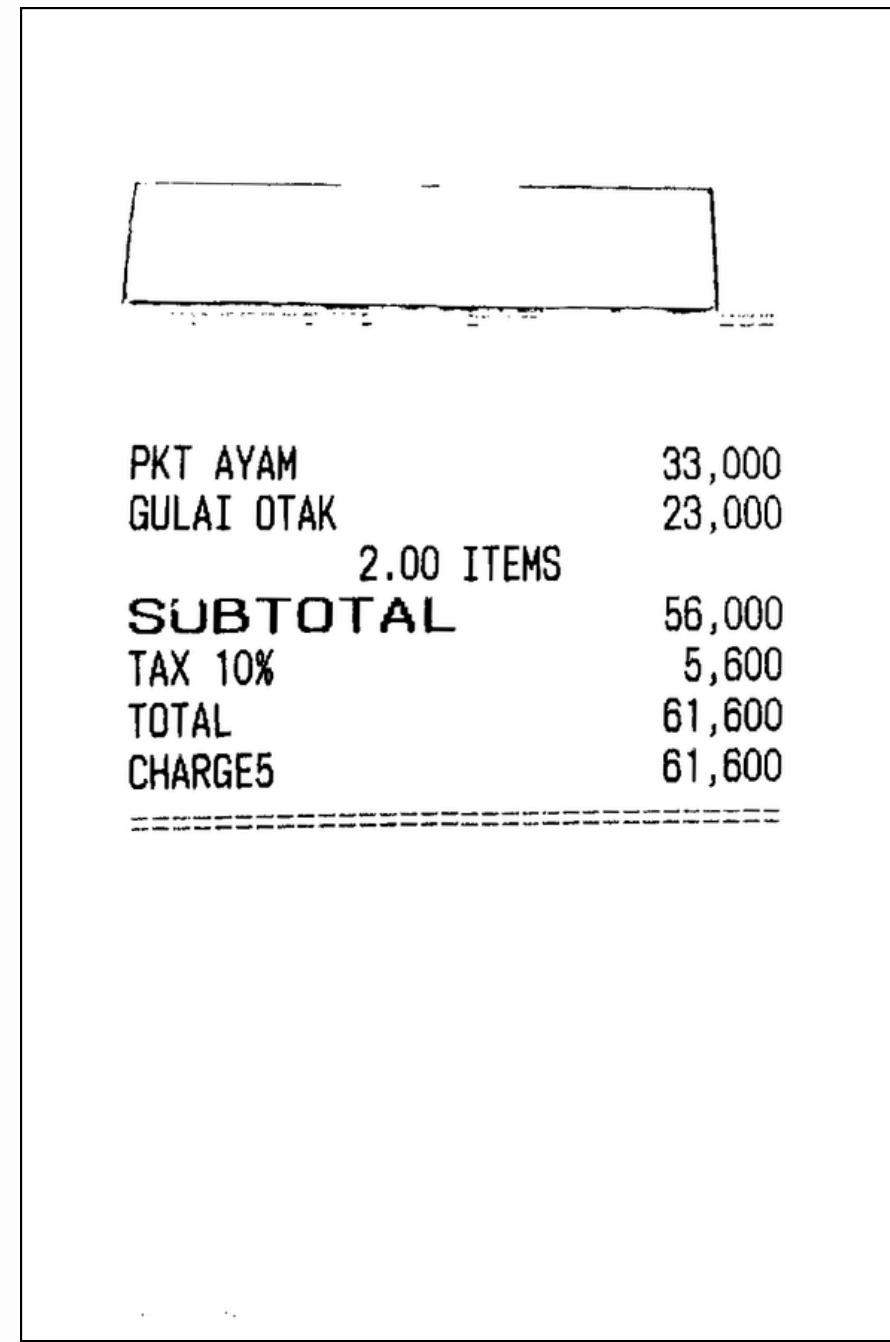
## ASSOCIATION OBJET - MONTANT - 1 / 3



receipt_filename	object	amount
test_receipt_00044.png	1 Combo	1 30.000
test_receipt_00044.png		TOTAL 30.000
test_receipt_00044.png		CASH 50.000
test_receipt_00044.png		CHANGED 20.000

# 07. PERFORMANCES DU MODÈLE

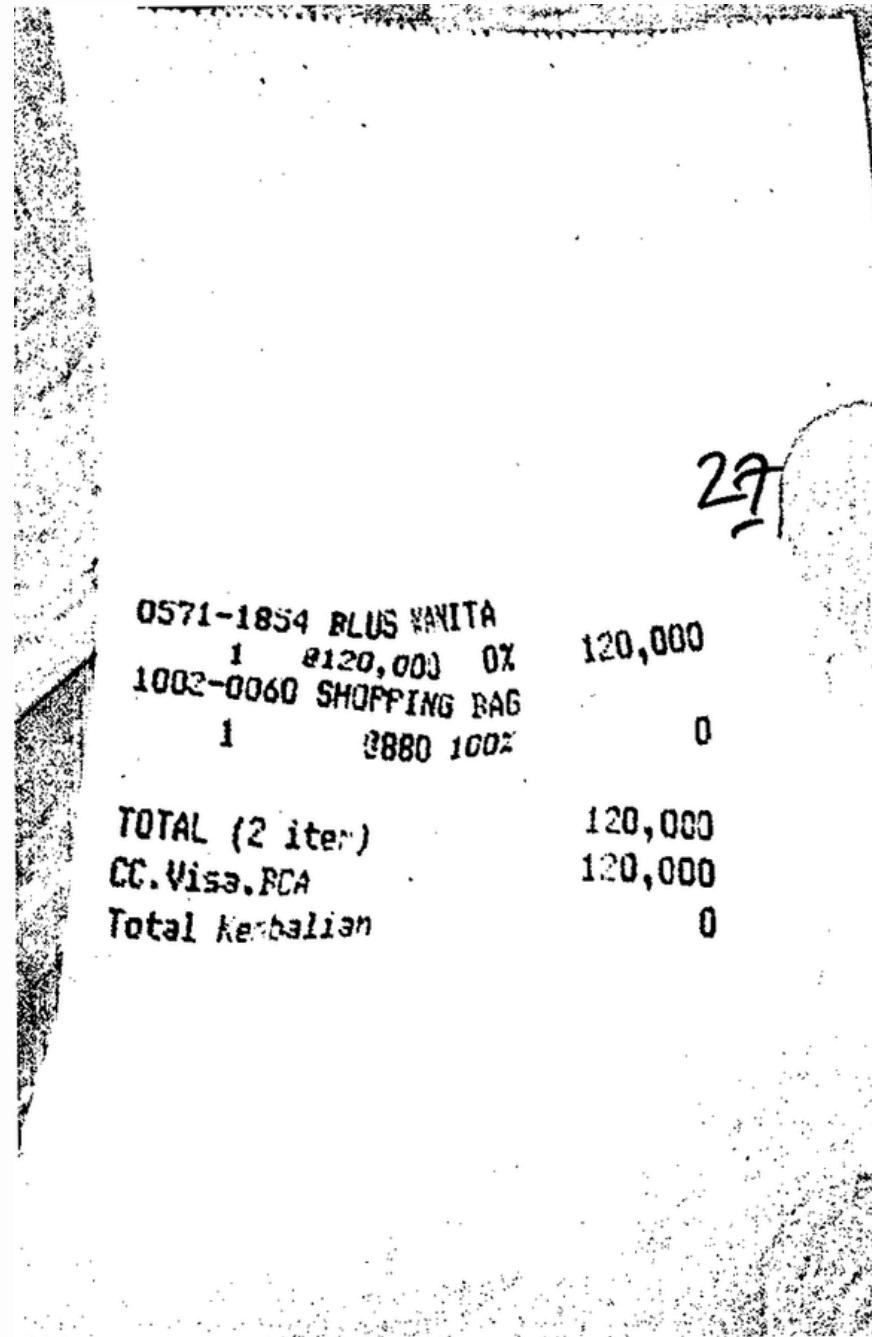
## ASSOCIATION OBJET - MONTANT - 2 / 3



receipt_filename	object	amount
test_receipt_00027.png	PKT AYAM	33,000
test_receipt_00027.png	GULAI OTAX	23,000
test_receipt_00027.png	SUBTOTAL	56,000
test_receipt_00027.png	TAX 10%	5,600
test_receipt_00027.png	TOTAL	61,600
test_receipt_00027.png	CHARGE5	61,600

## 07. PERFORMANCES DU MODÈLE

### ASSOCIATION OBJET - MONTANT - 3 / 3



receipt_filename	object	amount
test_receipt_00012.png	mst-tesw UUs	ww0ITA
test_receipt_00012.png	1 8a0,00 0X	120,000
test_receipt_00012.png	1 8B0 o	0
test_receipt_00012.png	TOTAL 2 iter	120,00
test_receipt_00012.png	CCN0	120,000
test_receipt_00012.png	Total Kadalian	0

# 08. LIMITES ET PISTES D'AMÉLIORATION

## PRÉTRAITEMENT DES DONNÉES

- **Hypothèses effectuées:**

- 1. Le ticket de caisse est initialement "centré" sur l'image (relativement au reste des éléments de l'image)
- 2. Second cropping est effectué en se basant sur la proportion de l'image que recouvre le ticket de caisse (85%)
- 3. L'ordre, dans le fichier metadata.pkl est le même pour les bounding boxes que pour les mots/labels associés

- **Limites du prétraitement:**

- 1. Si la segmentation par GMM reste plus souple que K-Means, elle repose sur les différences de couleurs des pixels
  - Sensible aux zones d'ombre au sein de l'image
  - Pas de possibilité de binariser l'image auparavant

- **Pistes d'amélioration:**

- 1. Plutôt que d'effectuer de la segmentation par Machine Learning, utiliser une autre méthode de détection des contours: par exemple détecter les contours du reçu puis de les lisser afin d'extraire le contenu du plus grand contour.
  - Approche qui risque d'être toutefois limitée par le bruit ou le fond s'il n'est pas vierge

# 08. LIMITES ET PISTES D'AMÉLIORATION

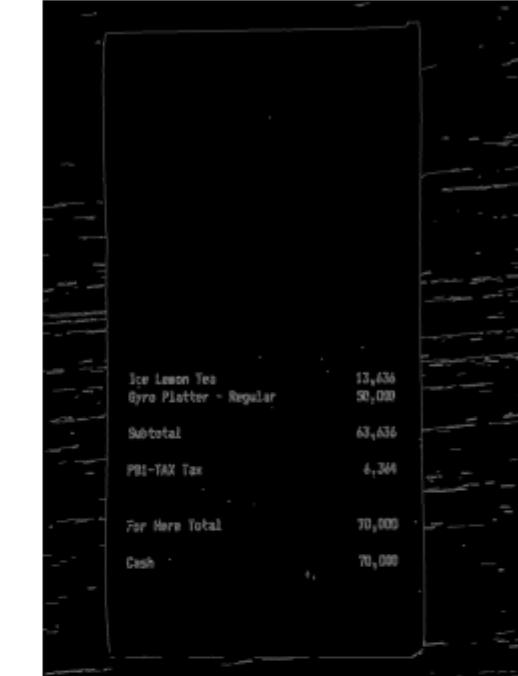
## PRÉTRAITEMENT DES DONNÉES

Edge Detection using Canny Method

Original: dev\_receipt\_00037.png



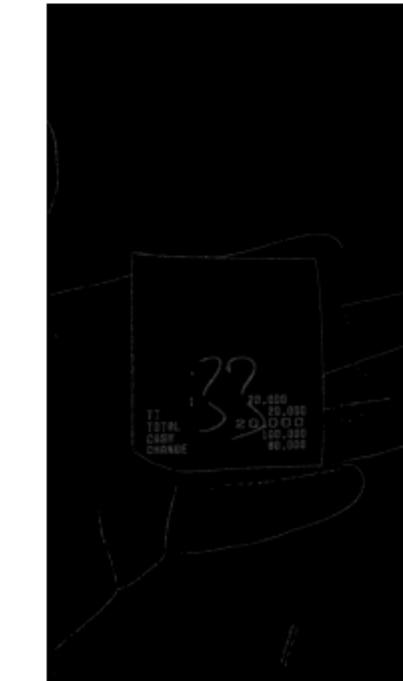
Edges: dev\_receipt\_00037.png



Original: dev\_receipt\_00016.png



Edges: dev\_receipt\_00016.png



# 08. LIMITES ET PISTES D'AMÉLIORATION

## DÉTECTION DES BOUNDING BOXES/ZONES DE TEXTE

- **Hypothèses effectuées:**

- 1. Le texte est à priori plus large que haut (avec une certaine tolérance)

- **Limites de l'algorithme de détection:**

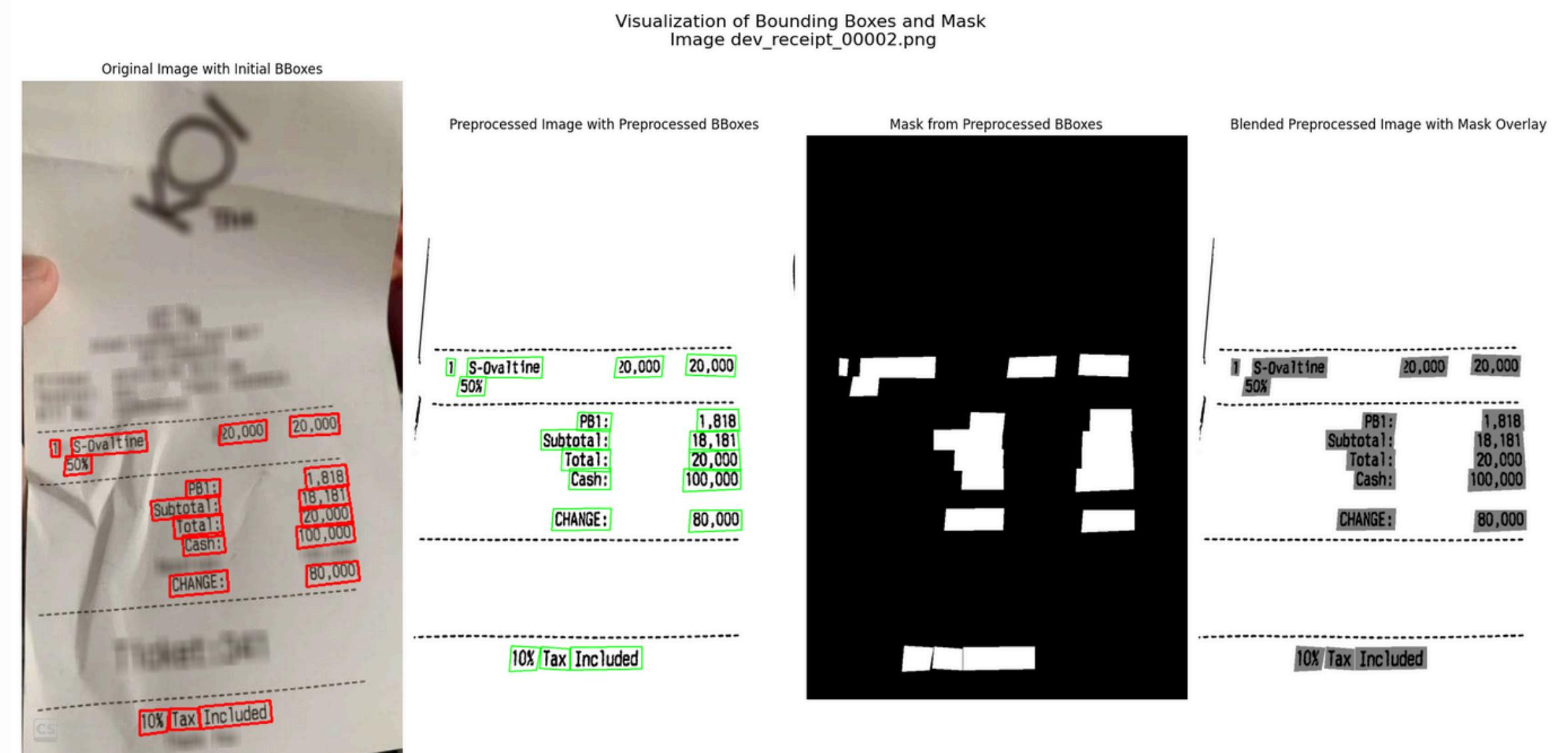
- 1. L'algorithme de détection de bboxes implémenté est sensible au bruit, malgré les contraintes imposées.

- **Pistes d'amélioration:**

- 1. Affiner les contraintes appliquées à l'algorithme de détection, ou en définir un autre type
    - 2. Cas typique d'une approche par Deep Learning:
      - Utilisation d'un CNN (VGG, ResNet, etc.) s'il s'agit de renvoyer les coordonnées des bboxes dans l'espace des images pré-traitées
      - Utilisation d'un AutoEncodeur (avec couches convolutées) voire d'un CNN (U-Net, etc.) s'il s'agit de segmenter le texte dans les reçus
- Nécessité d'un **très grand nombre de données** que nous n'avons pas ici

# 08. LIMITES ET PISTES D'AMÉLIORATION

## DÉTECTION DES BOUNDING BOXES/ZONES DE TEXTE



# 08 . LIMITES ET PISTES D'AMÉLIORATION

## DATASET D'ENTRAÎNEMENT + VALIDATION & CRNN

- Hypothèses effectuées:

- 1. Le jeu d'entraînement est représentatif de l'ensemble des données comprises dans le jeu de test.

- Limites:

- 1. Le jeu de données issu des tickets "dev\_" ne contiennent pas l'ensemble des caractères que nous pouvons trouver dans les reçus de l'ensemble de test. Le nombre de caractères distincts dans l'ensemble d'entraînement (+ validation) est de 79, contre 84 pour les tickets "test\_".
  - 2. La stratification est faite au niveau des reçus, pour éviter toute fuite de données. L'approche est limitée, mais semble la seule viable.

- Pistes d'amélioration:

- 1. **Complexification du modèle CRNN.** Ajout de couches de convolution, de couches BiLSTM, etc. Il peut alors être nécessaire d'effectuer davantage de **techniques de régularisation** (limiter l'overfitting)
  - 2. Considérer une **autre architecture**, type ViT, mais qui nécessite pour l'entraînement beaucoup plus de données.
  - 3. Considérer du **Transfer learning**
  - 4. **Si nous avions voulu utiliser l'algorithme de détection implémenté nous-mêmes, il aurait fallu ajouter du bruit aux données d'entraînement et de validation du CRNN, afin de le rendre robuste au bruit lors de l'inférence.**

