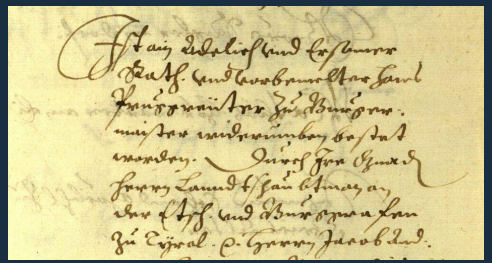
Apprentissage semi-supervisé multi-hypothèses pour la reconnaissance d'écriture manuscrite

Alexandre Chapin, Yann Soullard, Bertrand Couasnon, Aurélie Lemaître

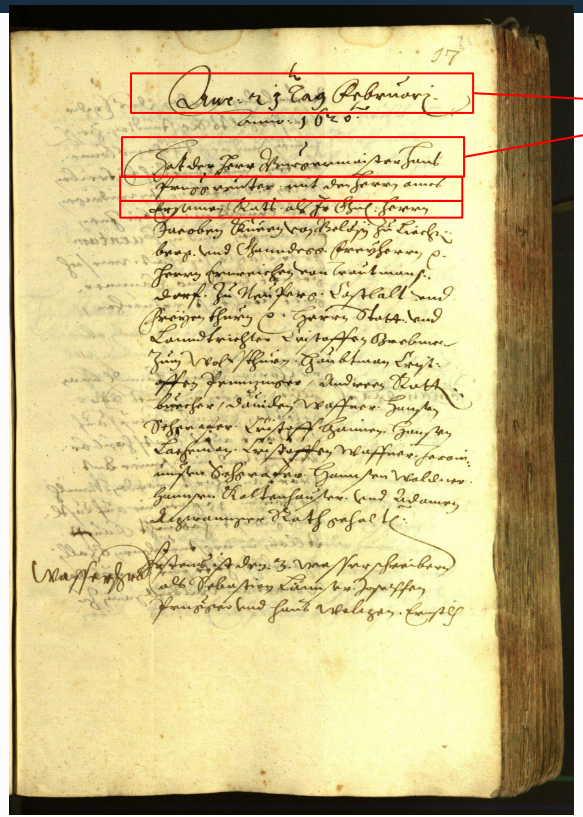


UNIVERSITÉ RENNES 2

intuitive user interaction for document

UMR IRISA

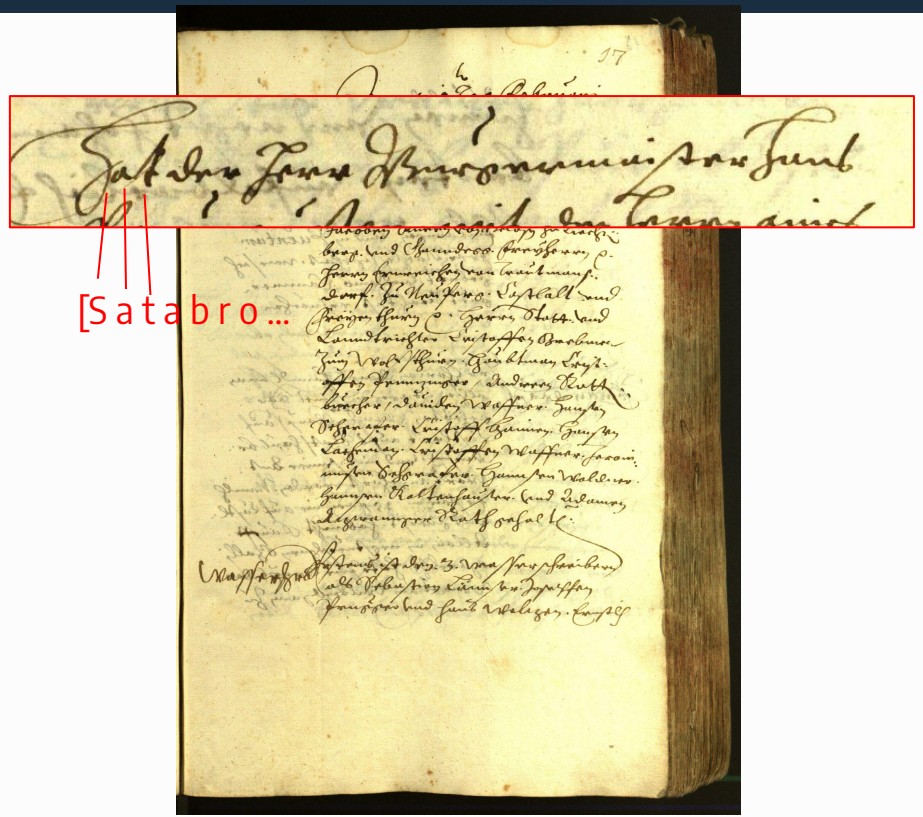
Contexte : reconnaissance d'écriture manuscrite



Identification des zones de texte

Image extraite de la base de données READ18

Contexte : reconnaissance d'écriture manuscrite



[Satabro...]

Reconnaissance de texte

Image extraite de la base de données READ18

Contexte : reconnaissance d'écriture manuscrite

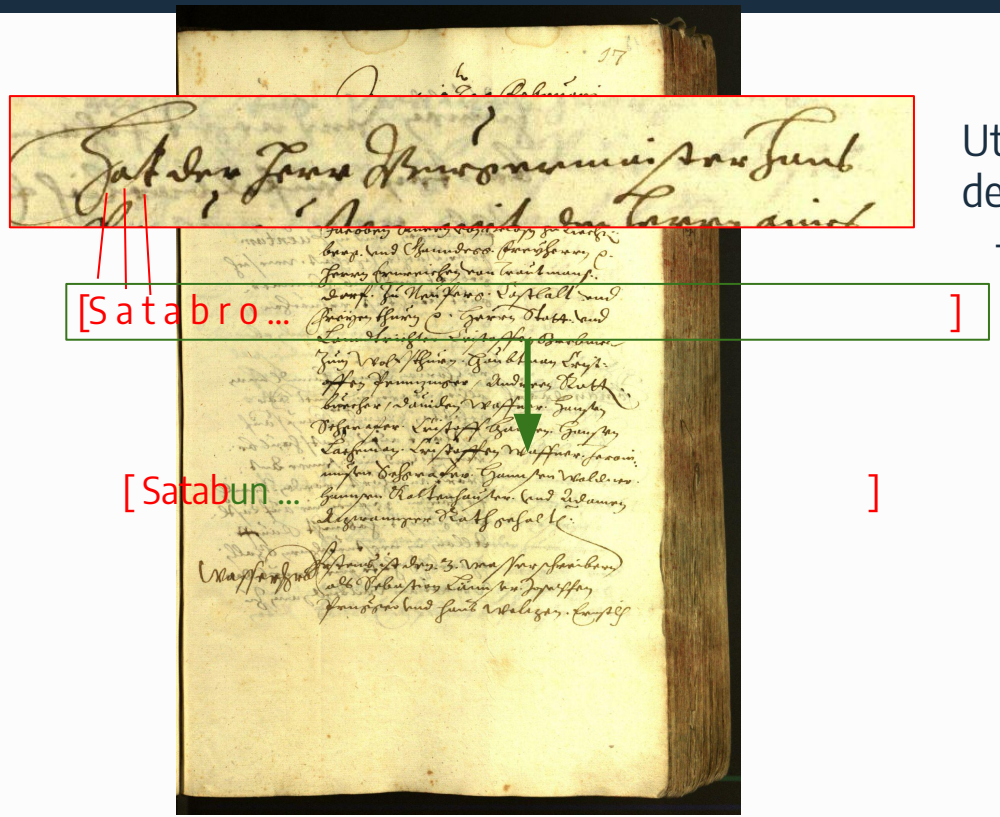


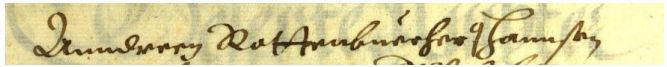
Image extraite de la base de données READ18

Utilisation d'un modèle de langage :

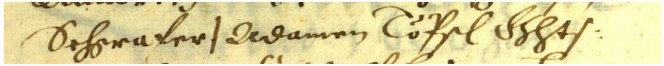
- Pour corriger les erreurs

Le problème du manque de données

Lignes de texte avec **transcription** associées **(annotations) :**



'Anndreen Rottenbūecher. Hannsen'

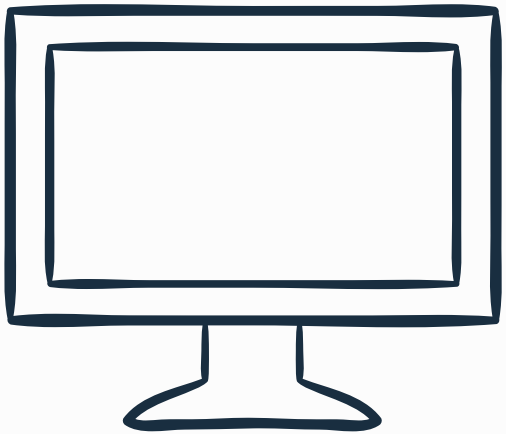


'Schgrafer/ Adamen TōPsl Rhts¬'

Peu de données annotées car difficile à annoter (même pour un expert)

Ont besoin de ...

Modèles d'**apprentissage profond**



Notre contribution

- **Combiner des méthodes** d'apprentissage profond
- Apprendre avec **peu de données**
- Tirer profit d'**exemples non étiquetés**

01

Etat de l’art

Modèles classiques : CRNN

Convolutionnal **R**ecurrent **N**eural **N**etwork

Principe

- ***Couches de convolution*** en entrée : extraction caractéristiques de l'image
- ***Couches de récurrence*** : étude séquentielle des caractéristiques
- **Algorithme CTC** : fonction de coût pour entraîner le modèle en associant les sorties du réseau à la vérité terrain

Problème

- Impossible de paralléliser les calculs à cause de la séquentialité des couches de récurrence

Fully-Convolutional Networks (FCN)

Principe

Supprimer la partie récurrente au profit de couches de convolution

Avantages

- **Parallélisation** entraînement
- **Résultats à l'état de l'art**

Inconvénients

- Architectures très **complexes**

  → Plein de mécanismes combinés complexes (attention, normalisation, ...)

  → Mise au point expérimentale non triviale

[M. Yousef et al. (2020), D. Coquenet et al. (2020)]

Transformers [Vaswani et al. (2017)]

Émergence dans nombreux domaines :
- Reconnaissance de parole
- Traitement du langage
- ...

Avantages

- **Parallélisation** de l'entraînement
- **Analyse séquentielle** des données
- Mécanisme d'**Attention**

Inconvénients

- **Besoin** de **nombreuses données** pour l'entraînement

[Kang et al. (2020), Barrere et al. (2022), Wick et al. (2022)]

Faire face au manque de données

- **Génération de données**
  - GAN (Generative Adversarial Network) [S. Fogel et al. (2020)]
  - Utilisation de polices manuscrites

Faire face au manque de données

- Génération de données
  - GAN (Generative Adversarial Network) [S. Fogel et al. (2020)]
  - Utilisation de polices manuscrites
- **Augmentation de données** [Y. Soullard et al. (2019)]
  - Transformations (rotation, scaling, projection, ...)

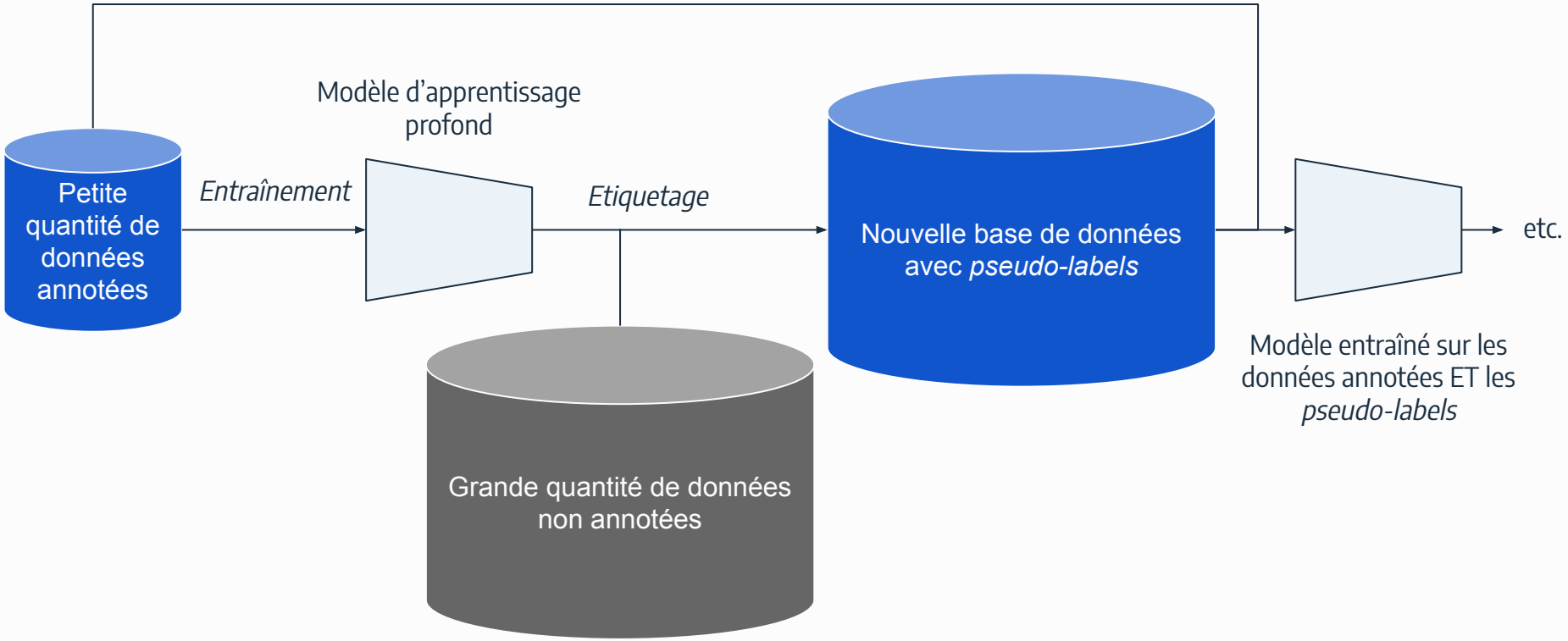
Faire face au manque de données

- Génération de données
  - GAN (Generative Adversarial Network) [S. Fogel et al. (2020)]
  - Utilisation de polices manuscrites
- Augmentation de données [Y. Soullard et al. (2019)]
  - Transformations (rotation, scaling, projection, ...)
- **Approches semi-supervisées**
  - Faire usage des données non étiquetées

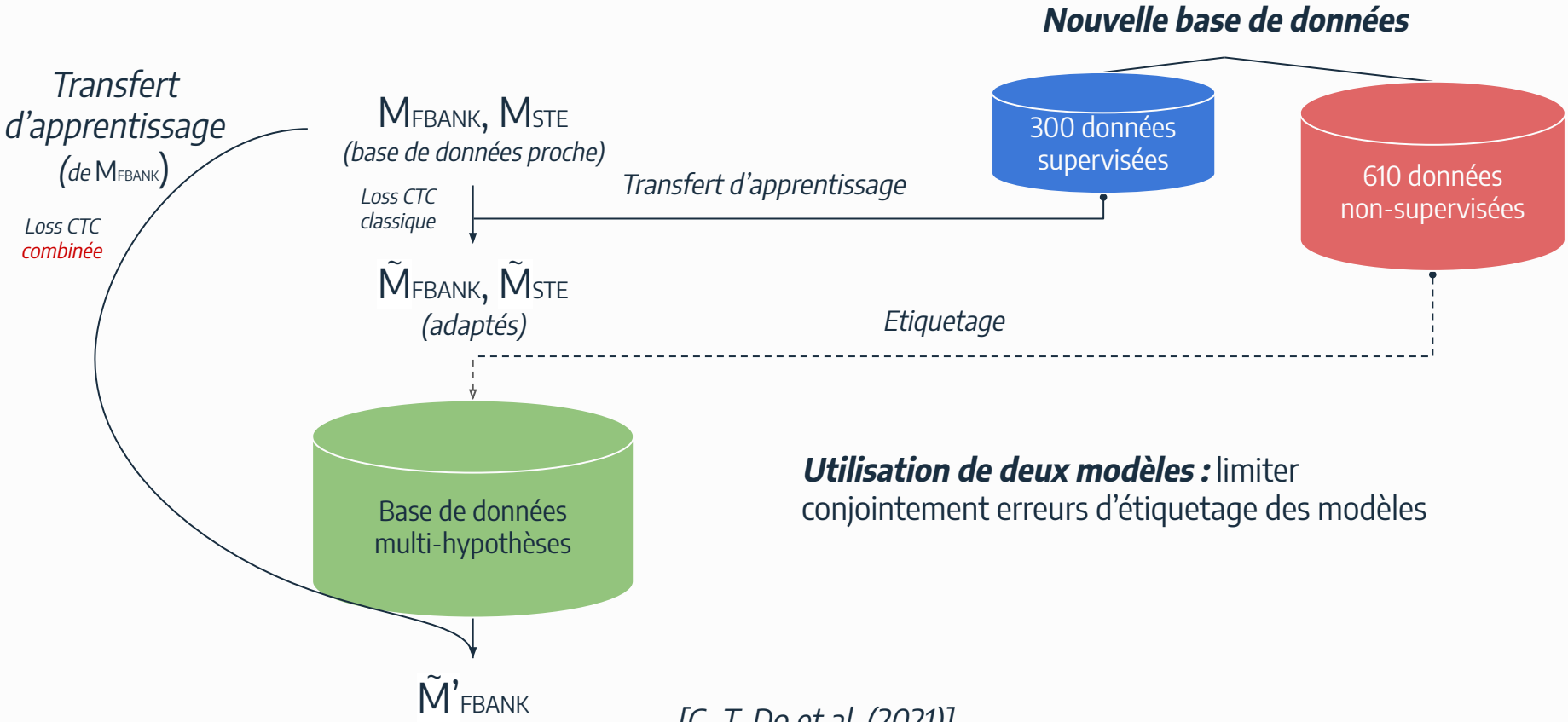
⇒ Peu de littérature en reconnaissance d'écriture manuscrite

Approches semi-supervisées

Une façon de faire : produire des pseudo-labels sur les données non annotées



Semi-supervisé multi-hypothèses : parole



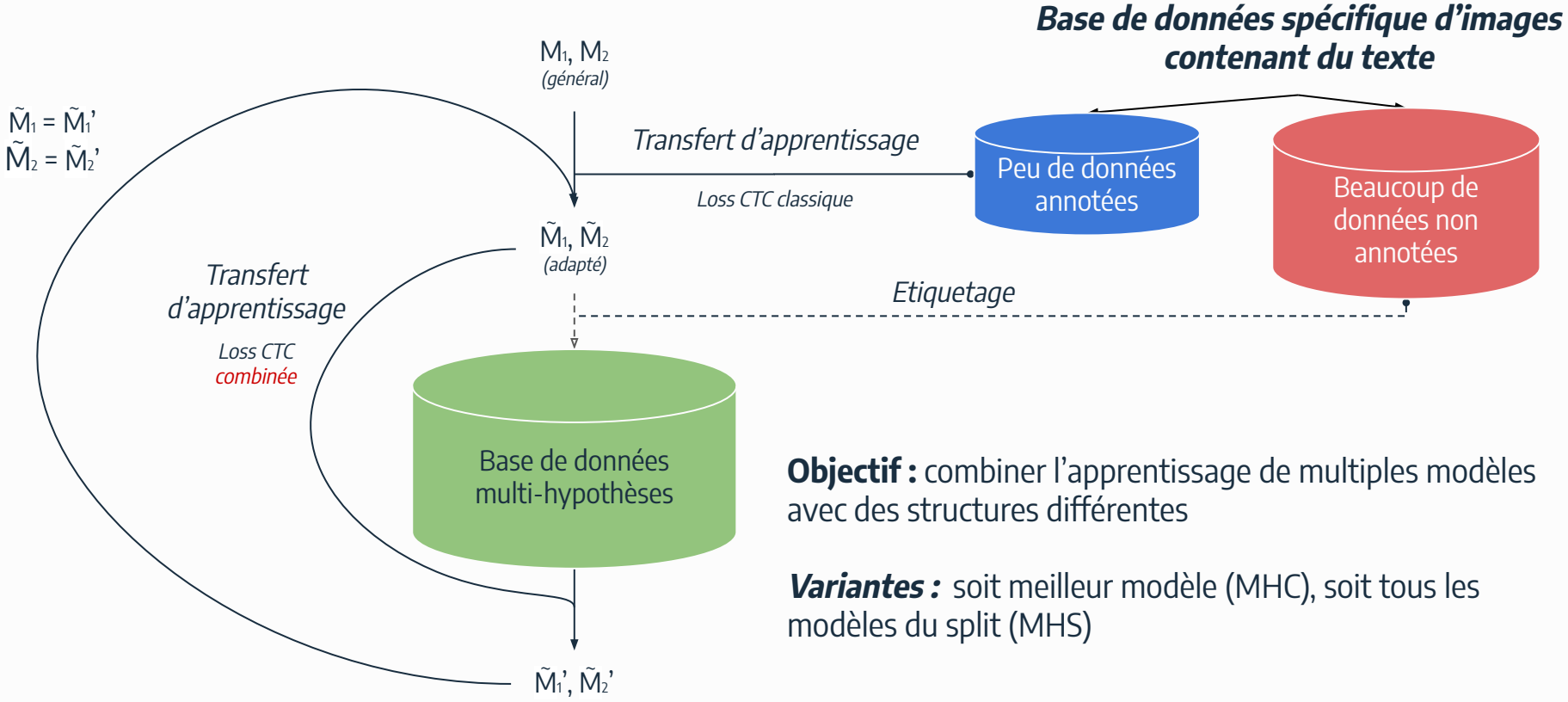
Utilisation de deux modèles : limiter conjointement erreurs d'étiquetage des modèles

[C.-T. Do et al. (2021)]

02

Contribution

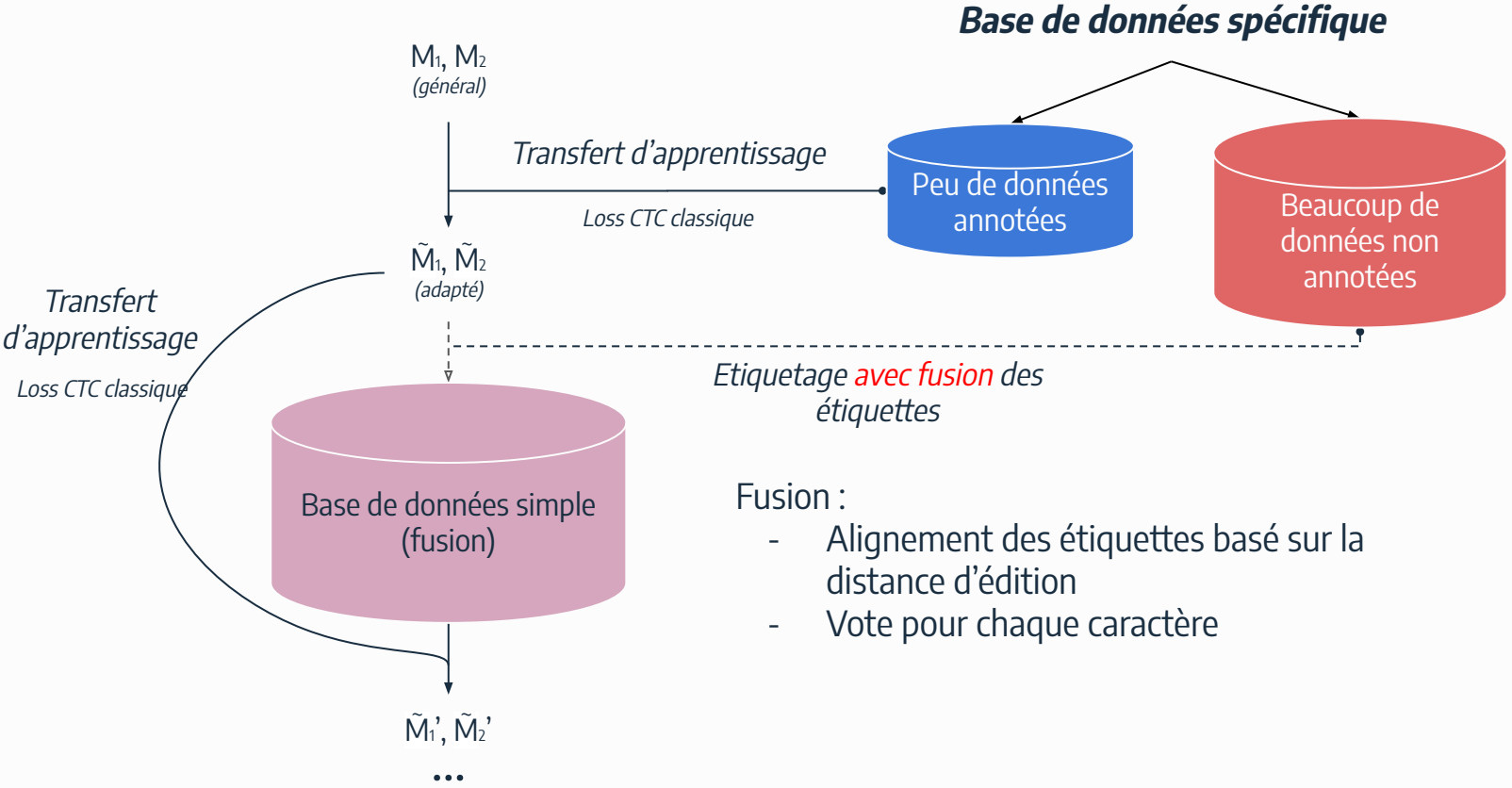
Semi-supervisé multi-hypothèses : principe



Objectif : combiner l'apprentissage de multiples modèles avec des structures différentes

Variantes : soit meilleur modèle (MHC), soit tous les modèles du split (MHS)

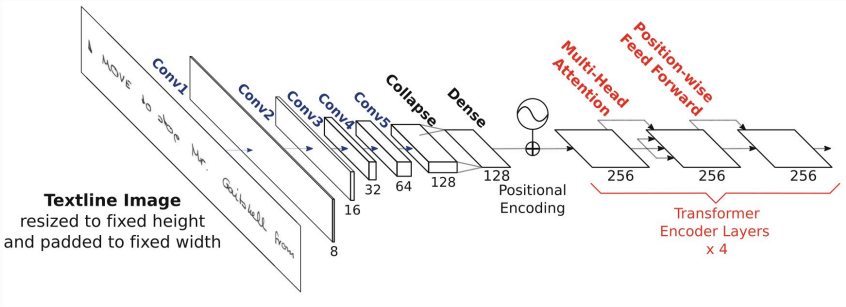
Variante : Stratégie avec fusion



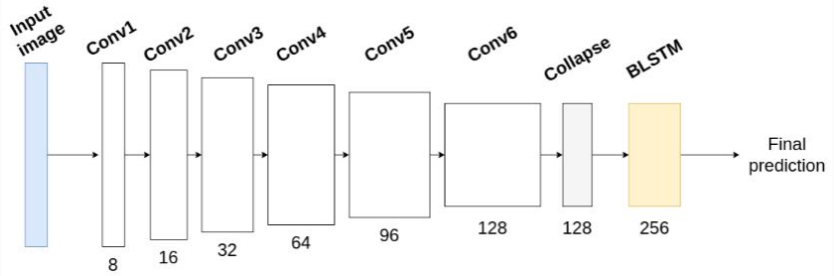
Fusion :

- Alignement des étiquettes basé sur la distance d'édition
- Vote pour chaque caractère

Choix des modèles



CTNN léger (Transformer) → M1
[K. Barrere et al. (2022)]



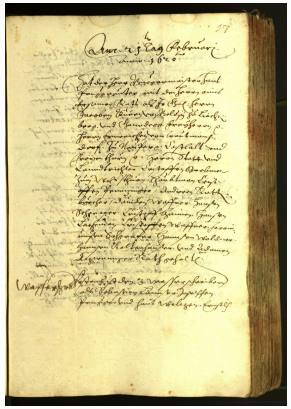
CRNN classique →M2

03

Expériences

Base de données READ18

ICFHR 2018 : compétition "*Automated Text Recognition on a READ dataset*"



	general data	specific data	test data	total
documents	17	5	5	22

Différentes langues : Allemand moderne et médiéval, Anglais, Danois, Italien,...

Objectif :

- Pré-entraînement sur *general data*
- Obtenir un CER minimum sur *specific data*
- Vérifier l'impact du nombre de données avec : 1, 4 et 16 pages annotées utilisées

***Caracter Error Rate* (CER)** : taux de caractères incorrectement reconnus par rapport au texte de référence (**distance de Levenshtein**).

Plus le CER est faible, mieux c'est !

Quelle stratégie adopter ? Et dans quelle configuration ?

Fusion (FUS) ? Multi-Hypothèses Choix (MHC) ?
Multi-Hypothèses Split (MHS) ?

Comparaison des stratégies

0 page for adaptation

	NoA[1]	SA[2]	MHC	MHS	FUS
Konzils.	17.10	-	15.15	**13.97**	16.22
Ricordi	29.27	-	25.94	26.45	**24.46**
Schiller	**31.33**	-	41.44	100	34.57
Patzig	35.99	-	40.08	42.01	**35.45**
Schwerin	33.03	-	33.05	**26.25**	26.68
Mean M_1	29.35	-	29.77	41.73	**27.48**
Konzils.	19.04	-	10.54	**9.55**	11.59
Ricordi	43.96	-	23.81	24.09	**22.48**
Schiller	33.79	-	41.13	100	**32.94**
Patzig	47.97	-	36.40	39.00	**29.96**
Schwerin	53.25	-	26.73	**25.33**	26.06
Mean M_2	39.60	-	27.72	39.67	**24.61**

[1]NoA: No adaptation

[2]SA: Simple adaptation

1 page for adaptation

	NoA[1]	SA[2]	MHC	MHS	FUS
Konzils.	17.10	**10.96**	**10.89**	12.81	13.32
Ricordi	29.27	22.36	**20.97**	**21.05**	22.56
Schiller	31.33	**23.04**	26.74	27.89	26.61
Patzig	35.99	**25.67**	27.15	30.09	31.24
Schwerin	33.03	12.29	9.54	**8.75**	9.63
Mean M_1	29.35	**18.86**	19.06	20.12	20.67
Konzils.	17.10	12.38	9.06	**8.53**	9.46
Ricordi	29.27	28.57	19.04	**17.70**	19.68
Schiller	31.33	23.76	**20.35**	20.95	**20.37**
Patzig	35.99	29.08	27.15	24.90	**23.01**
Schwerin	33.03	13.26	**7.92**	9.37	9.06
Mean M_2	39.26	21.41	16.70	**16.29**	**16.32**

[1]NoA: No adaptation

[2]SA: Simple adaptation

4 pages for adaptation

	NoA[1]	SA[2]	MHC	MHS	FUS
Konzils.	17.10	**7.82**	**7.81**	9.54	9.53
Ricordi	29.27	**17.69**	18.50	18.85	19.66
Schiller	31.33	**20.10**	**20.08**	21.70	23.99
Patzig	35.99	**20.89**	**20.90**	24.38	21.13
Schwerin	33.03	8.83	**7.33**	**7.22**	8.80
Mean M_1	29.35	**15.07**	**14.92**	16.34	16.62
Konzils.	17.10	8.13	**7.14**	7.32	**7.03**
Ricordi	29.27	21.42	16.43	**15.87**	**16.01**
Schiller	31.33	20.03	16.60	**16.18**	18.03
Patzig	35.99	21.64	19.88	**19.10**	**19.19**
Schwerin	33.03	7.25	5.84	**5.37**	5.65
Mean M_2	39.26	15.69	13.18	**12.77**	13.18

[1]NoA: No adaptation

[2]SA: Simple adaptation

- Meilleure stratégie : **FUS**
- Le **CRNN** semble d'avantages profiter de la procédure que le **CTNN**
- FUS semble capable de limiter les erreurs de chaque modèle

- CRNN, CTNN : **MHS, SA**
- **CRNN** améliore grandement ses résultats grâce à notre méthode **MHS**
- Impact de la **fusion** semble **moins important** étant donné que les modèles sont adaptés

Comment se comportent nos stratégies face à une approche semi-supervisée classique ?

Comparaison à du semi-supervisé classique

0 page for adaptation			
	NoA[1]	Semi-sup.	FUS
Konzils.	17.10	16.78	**16.22**
Ricordi	29.27	29.15	**24.46**
Schiller	31.33	**31.16**	34.57
Patzig	35.99	35.82	**35.45**
Schwerin	33.03	33.05	**26.68**
Mean M_1	29.35	29.19	**27.48**
Konzils.	19.04	18.90	**11.59**
Ricordi	43.96	44.03	**22.48**
Schiller	33.79	33.64	**32.94**
Patzig	47.97	47.71	**29.96**
Schwerin	53.25	53.27	**26.06**
Mean M_2	39.60	39.51	**24.61**

[1]NoA: No adaptation

1 page for adaptation			
	NoA[1]	Semi-sup.	MHS
Konzils.	17.10	**10.05**	12.81
Ricordi	29.27	**19.85**	21.05
Schiller	31.33	**21.80**	27.89
Patzig	35.99	**24.62**	30.09
Schwerin	33.3	10.32	**8.75**
Mean M_1	29.35	**17.33**	20.12
Konzils.	17.10	9.40	**8.53**
Ricordi	29.27	25.03	**17.70**
Schiller	31.33	22.12	**20.95**
Patzig	35.99	27.15	**24.90**
Schwerin	33.03	10.96	**9.37**
Mean M_2	39.60	18.93	**16.29**

[1]NoA: No adaptation

4 pages for adaptation			
	NoA[1]	Semi-sup.	MHS
Konzils.	17.10	**7.76**	9.54
Ricordi	29.27	**16.08**	18.85
Schiller	31.33	**19.33**	21.70
Patzig	35.99	**20.39**	24.38
Schwerin	33.03	7.35	**7.22**
Mean M_1	29.35	**14.8**	16.34
Konzils.	17.10	**6.98**	7.32
Ricordi	29.27	19.33	**15.87**
Schiller	31.33	21.09	**16.18**
Patzig	35.99	21.10	**19.10**
Schwerin	33.03	6.44	**5.37**
Mean M_2	39.26	15.11	**12.77**

[1]NoA: No adaptation

- **Notre stratégie obtient de meilleurs résultats** : Correction commune des erreurs

- **CRNN** profite grandement de l'entraînement combiné
- **CTNN** obtient de meilleurs résultats seul avec une approche classique.

- Les meilleurs résultats globaux sont obtenus grâce à notre stratégie MHS dans le cadre d'un processus d'adaptation

Discussion sur notre approche multi-hypothèse

- Le **CRNN bénéficie** des bonnes capacités prédictives du CTNN et s'améliore le plus.

- Le **CTNN est impacté par des prédictions imparfaites** issues du CRNN.

- La stratégie de Fusion semble utile pour **réduire les erreurs de reconnaissance** lorsque le modèle se trompe souvent (cas sans adaptation).

- Dans le cas contraire, où les pseudo-labels sont plus faiblement erronées, **l'apprentissage des multiples hypothèses sans correction par fusion semble préférable**.

04

Conclusion

Conclusion

- Proposition d'une **nouvelle procédure d'apprentissage semi-supervisée** pour de la reconnaissance d'écriture manuscrite ancienne lorsque **peu de données annotées** sont accessibles

- Combinaison de réseaux de neurones profonds et expérimentation de différentes stratégies

- Des **résultats encourageants et proches de l'état de l'art** sur une base de données de référence malgré des modèles initiaux pas aussi performants

Perspectives

- Expérimenter l'approche avec des architectures à l'état de l'art

- Intégrer un modèle de langues au sein de l'approche pour apporter des corrections linguistiques aux prédictions

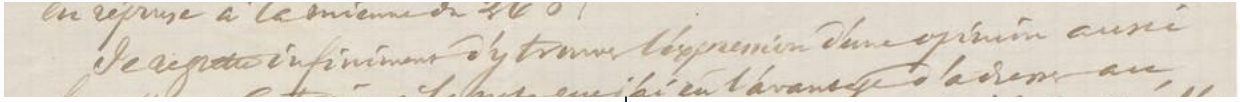
Bibliographie

- Graves et al. (2006).
  "*Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks*"
- Strauß et al. (2018).
  "*Icfhr2018 competition on automated text recognition on a read dataset*"
- M. Yousef et al. (2020).
  "*Accurate, data-efficient, unconstrained text recognition with convolutional neural networks*"
- Vaswani et al. (2017).
  "*Attention is all you need*"
- K. Barrere et al. (2022).
  "*A light Transformer-Based Architecture for Handwritten Text Recognition*"
- L. Kang et al. (2020).
  "*Pay attention to what you read: Non-recurrent handwritten text-line recognition*"
- Y. Soullard et al. (2019).
  "*Improving text recognition using optical and language model writer adaptation*"
- E. Chammas et al. (2018).
  "*Handwriting recognition of historical documents with few labeled data*"

Bibliographie

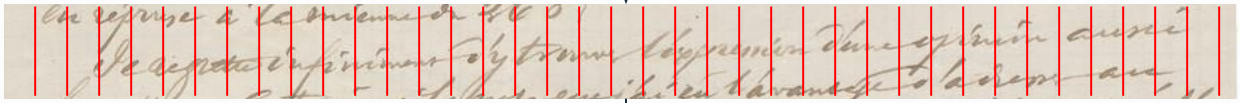
- S. Fogel et al. (2020).
  "*Scrabblegan: Semi-supervised varying length handwritten text generation*"
- C.-T. Do et al. (2021).
  "*Multiple-hypothesis ctc-based semi-supervised adaptation of end-to-end speech recognition*"
- G. Leifert et al. (2020).
  "*Two semi-supervised training approaches for automated text recognition*"
- B. Shi et al. (2016).
  "*An end-to-end trainable neural network for image-based sequence recognition and its application to scene-text recognition*"

Modèles classiques : CRNN



Extraction des caractéristiques de l'image d'entrée

CNN



Etude des caractéristiques de manière séquentielle

BLSTM

Relie séquences à vérité terrain

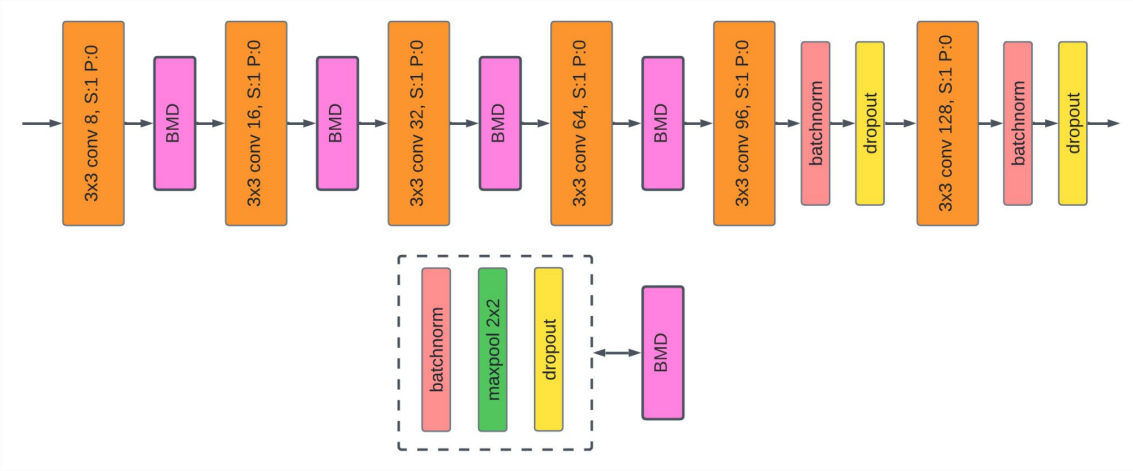
CTC

Matrice de proba.

⇔

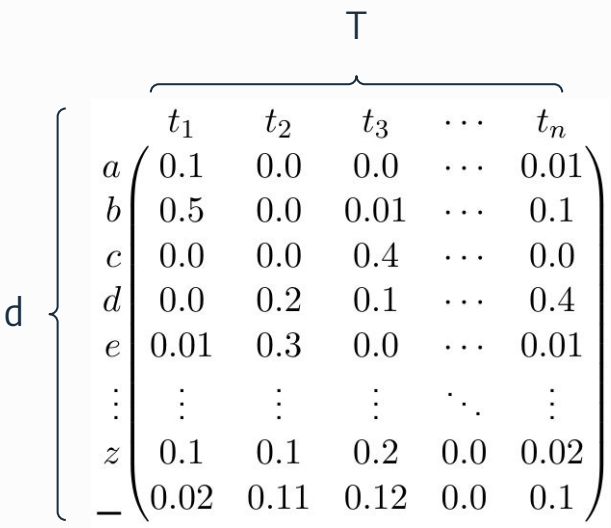
$$\begin{array}{c} \\ a \\ b \\ c \\ d \\ e \\ \vdots \\ z \\ \_ \end{array} \begin{array}{c} \begin{array}{ccccc} t_1 & t_2 & t_3 & \cdots & t_n \end{array} \\ \begin{pmatrix} 0.1 & 0.0 & 0.0 & \cdots & 0.01 \\ 0.5 & 0.0 & 0.01 & \cdots & 0.1 \\ 0.0 & 0.0 & 0.4 & \cdots & 0.0 \\ 0.0 & 0.2 & 0.1 & \cdots & 0.4 \\ 0.01 & 0.3 & 0.0 & \cdots & 0.01 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0.1 & 0.1 & 0.2 & 0.0 & 0.02 \\ 0.02 & 0.11 & 0.12 & 0.0 & 0.1 \end{pmatrix} \end{array}$$

Choix des modèles



CRNN inspiré par celui de l'Université de l'Ohio [B. Shi et al. (2016)]

Modèles classiques : Loss CTC



Soit une **séquence de *features*** de T caractères :

$$X = \{x_t \in \mathbb{R}^d | t = 1, \ldots, T\}$$

Soit une **transcription** de L caractères :

$$C = \{c_l \in U | l = 1, \ldots, L\}$$

La **fonction de coût CTC** est calculée en intégrant sur tous les chemins qui peuvent être associés à C, $B^{-1}(C)$:

$$L_{CTC} = -\log \sum_{\alpha \in B^{-1}(C)} P_\theta(\alpha | X)$$

Semi-supervisé multi-hypothèses : loss combinée

$\hat{\mathrm{C}}_i,\ i = 1, \ldots, N$: les ***pseudo-labels*** générés par les *N* modèles (N = 2 dans le papier)

Redéfinition de la **fonction de coût CTC :**

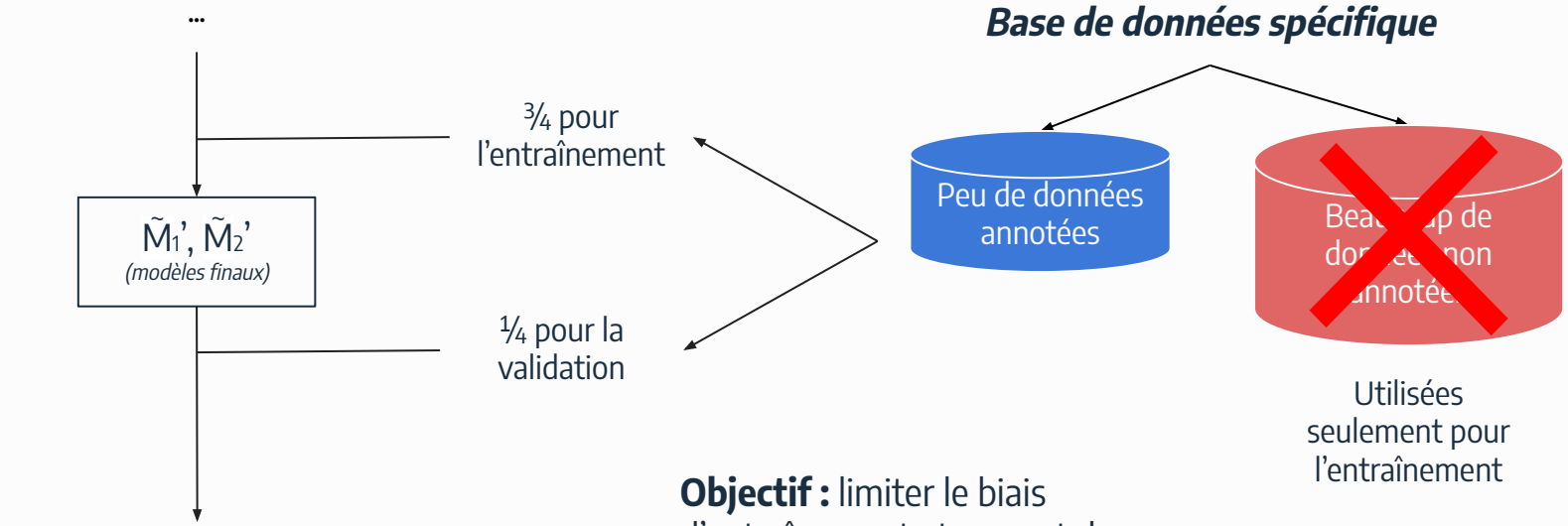
$$L^*_{CTC} = -\left(\sum_{i=1}^{N} \log P_\theta(\hat{\mathrm{C}}_i|X)\right)$$

Avec les propriétés de la fonction *log* on peut écrire :

$$L^*_{CTC} = -\log\left[\left(\sum_{a_i \in B^{-1}(\hat{\mathrm{C}}_1)} P_\theta(a_i|X)\right)\left(\sum_{b_i \in B^{-1}(\hat{\mathrm{C}}_2)} P_\theta(b_i|X)\right)\right]$$

[C.-T. Do et al. (2021)]

Validation

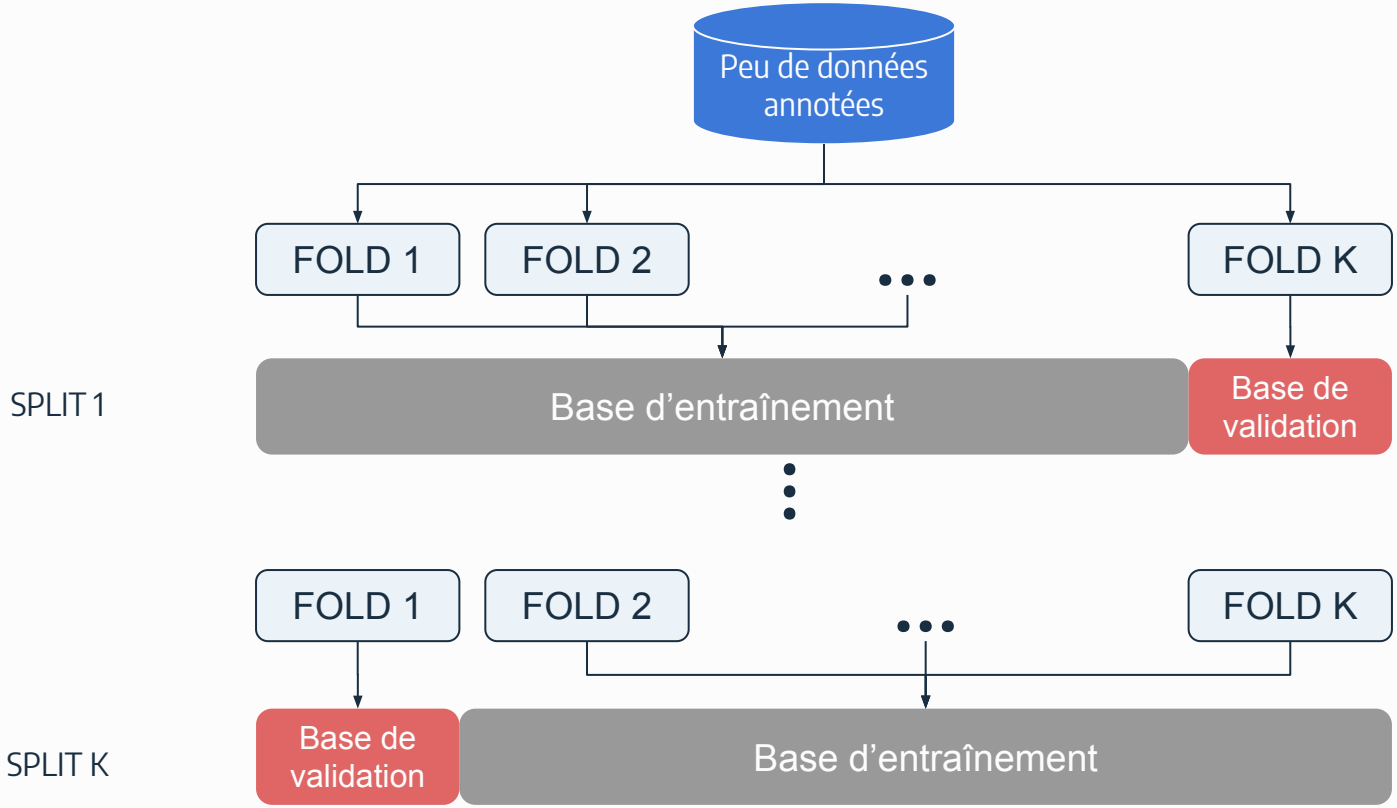


Calcul CER :
prédictions du modèle VS vérité terrain

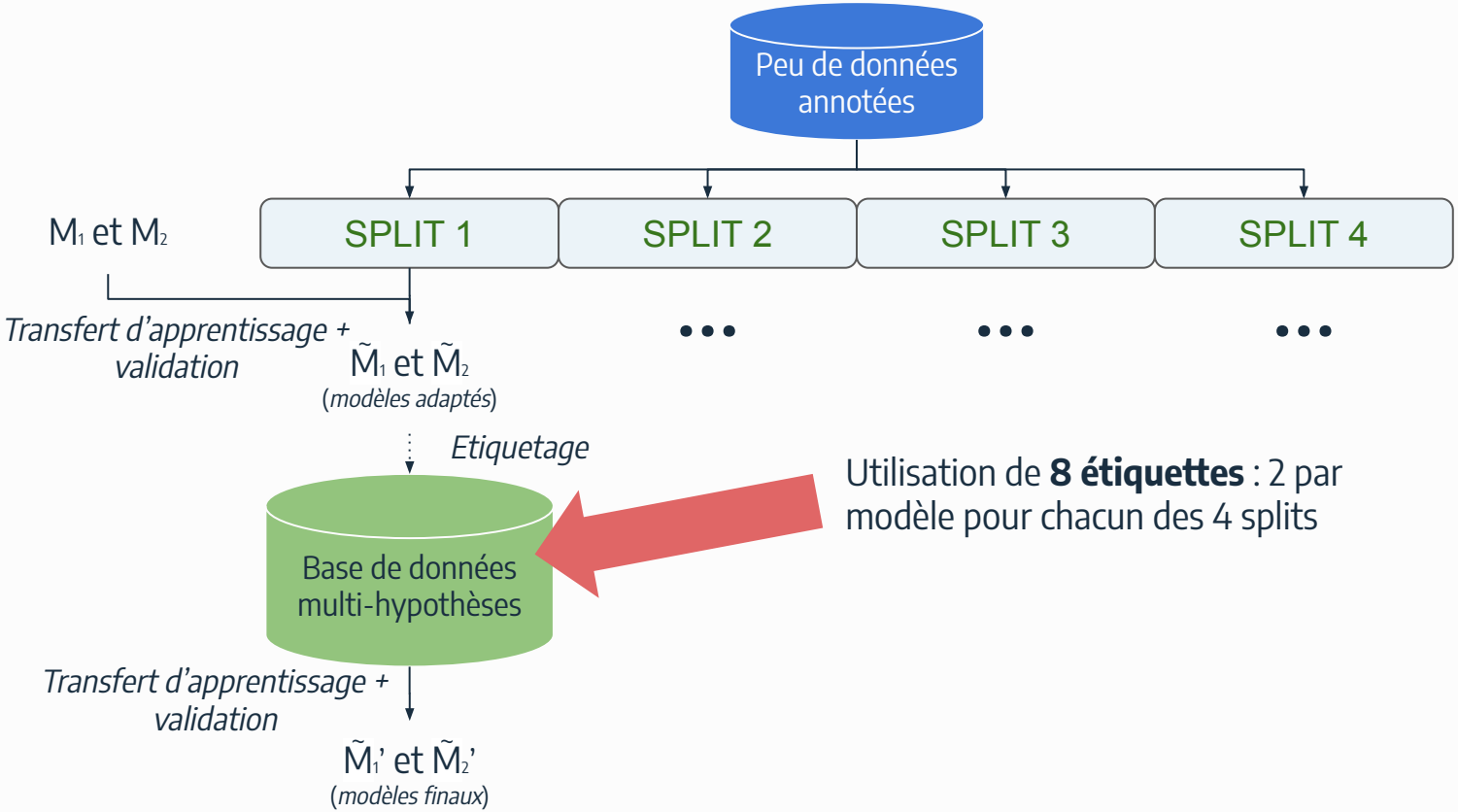
Objectif : limiter le biais d'entraînement et permet de vérifier un seuil

Problème : lorsque peu de données, cette méthode n'est pas efficace !

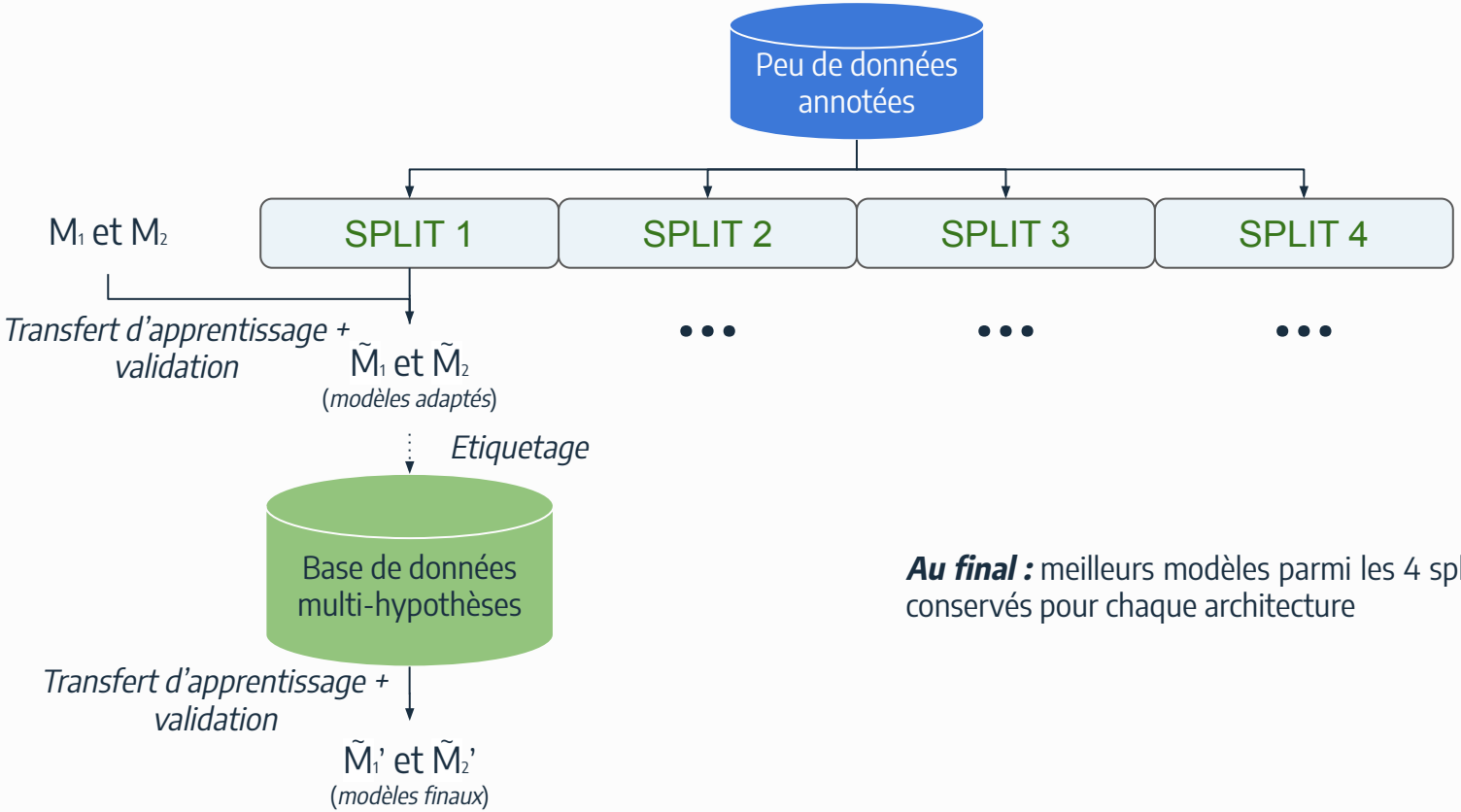
Validation croisée (*k-fold cross validation*)



Variante : Combinaison multi-split



Seuil par validation croisée (4-fold)



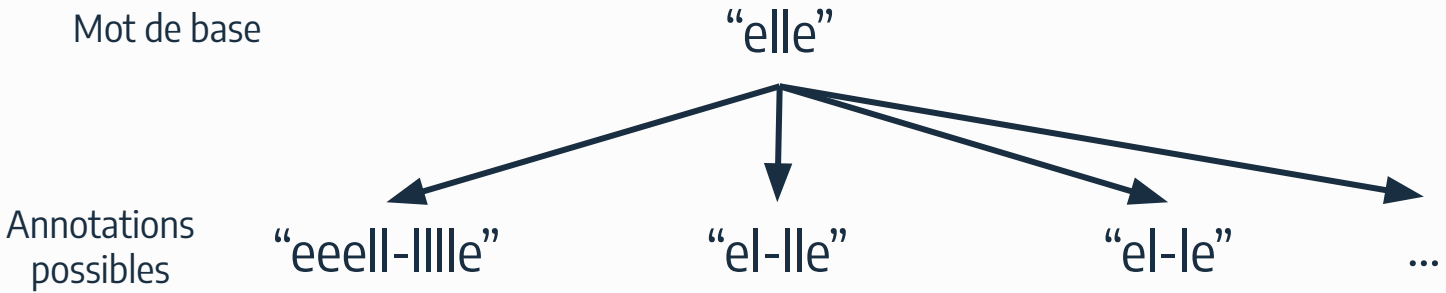
Au final : meilleurs modèles parmi les 4 splits sont conservés pour chaque architecture

Algorithme CTC : Decodage

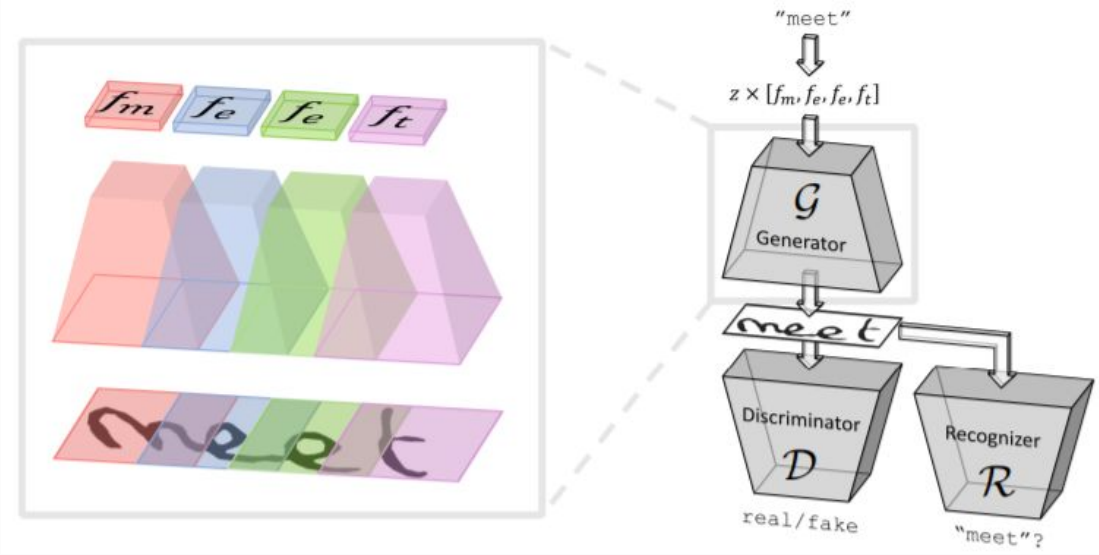
Problème : mots qui ont des caractères qui se répètent : *'belle'*, *'bille'*, *etc.*

⋮

Solution : ajout du caractère "***blank***" (noté **'_'**) entres caractères qui se répètent



Faire face au manque de données : génération de données



Architecture ScrabbleGan utilisée

Ensembles de mots générés par ScrabbleGan en suivant un certain style sur chaque ligne

[S. Fogel et al. (2020)]

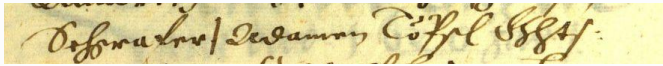
Algorithme CTC

Modèle à entraîner

- Séquence de *features* de nombre inconnu (**non-segmentée**)

Annotation du document manuscrit

'Anndreen Rottenbūecher. Hannsen'



'Schgrafer/ Adamen TōPsl Rhts¬'

- Séquence de caractères finie (**segmentée**)

Comment aligner les deux ?

Grâce à l'algorithme CTC !

Algorithme CTC : Decodage

	t_1	t_2	t_3	$\cdots$	t_n
a	0.1	0.0	0.0	$\cdots$	0.01
b	0.5	0.0	0.01	$\cdots$	0.1
c	0.0	0.0	0.4	$\cdots$	0.0
d	0.0	0.2	0.1	$\cdots$	0.4
e	0.01	0.3	0.0	$\cdots$	0.01
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
z	0.1	0.1	0.2	0.0	0.02
_	0.02	0.11	0.12	0.0	0.1

argmax(t1) = 1
→ caractère ‘b’

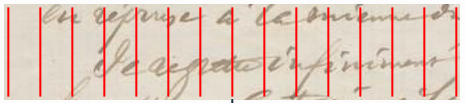
...

argmax(tn) = 3
→ caractère ‘d’

⇒

Taille n

bec...d



Caractères qui se répètent car ils apparaissent dans plusieurs “tranches”

Solution : supprimer les caractères qui se répètent

Problème : mots qui ont des caractères qui se répètent : *‘belle’*, *‘bille’*, *etc.*

Couches de convolution

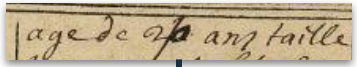
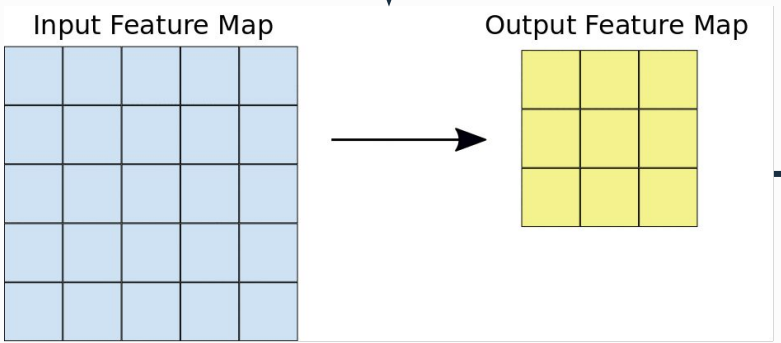
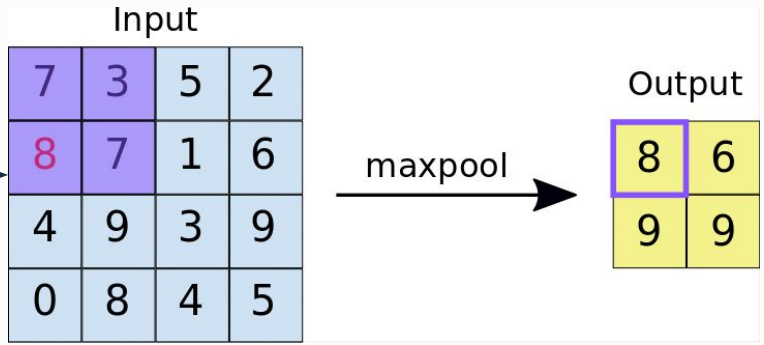


Image d'entrée



Opération de **convolution**

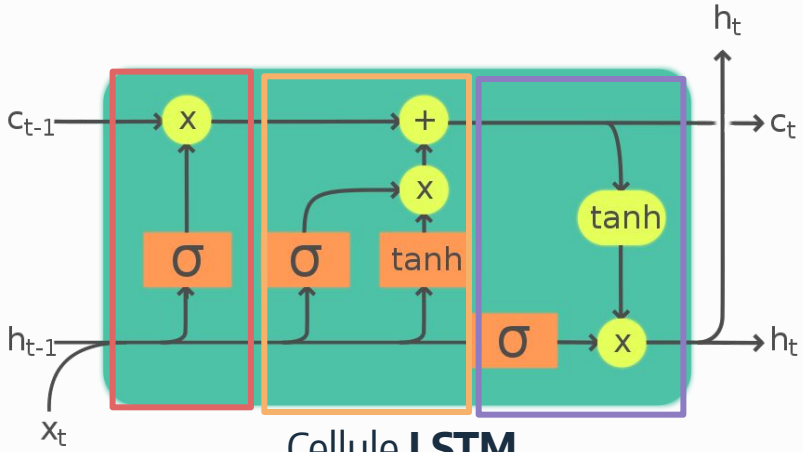
- Fenêtre glissante
- Application d'un noyau permettant d'extraire différents types de *features*



Opération de **max pooling**

- Fenêtre glissante
- Permet d'agréger localement les informations

Couches récurrentes : LSTM et BLSTM



Cellule **LSTM**

BLSTM 2 modèles LSTM apprennent la séquence en parallèle, un à l'endroit, l'autre à l'envers.

Idées

- *Séquence d'entrée* ***x***
- *Cell state* **c** permet de conserver ce qui est important à long terme
- *Hidden state* **h** permet de conserver ce qui est important à court terme
- ***Input*** *gate* : ajoute informations pertinentes pour long-terme à c
- ***Forget*** *gate* : détermine informations pertinentes venant du passé
- ***Output*** *gate* : récupère dans c les informations pertinentes à court terme

Extensions possible

- Ajouter une surcouche de modèle de langue dans la procédure afin de corriger encore d'avantages les erreurs → pour se rapprocher de l'état de l'art ?

- Tenter d'autres combinaisons de modèles ? (Transformer, FCN, etc.)