Des pipelines faciles à réutiliser pour comparer les performances d'outils de reconnaissance d'entités nommées sur les textes cliniques en français

Thibault Hubert, Ghislain Vaillant, Olivier Birot, Camila Arias,
Antoine Neuraz, Bastien Rance et Adrien Coulet

Santé & IA @PFIA, puis à MIE 2024

HeKA

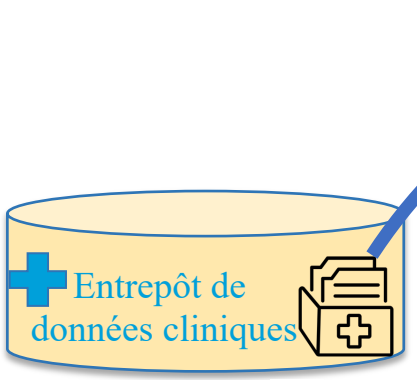
Inria

Inserm

Université Paris Cité

Motivation : faciliter l'extraction de connaissances à partir des textes cliniques (1/2)

- Intérêt des textes cliniques, notamment dans les études observationnelles
  - Pour inclure ou exclure les patients d'une étude
  - Pour capturer la variable réponse ou les co-variables



Textes cliniques (non structurés)

Outils de TAL

←Patients→

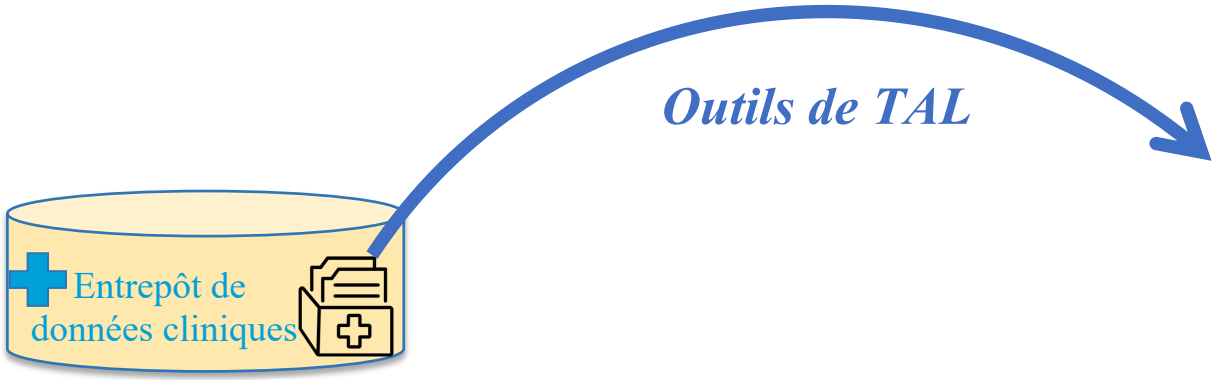
Heart failure	Chest pain	0	0	0	1.7	1	1	0
Heart failure	Hypoxia	21	0	0	3	0	0	0
Stroke	Headache	20	0	4	0	0	0	0
Diagnostics	**Symptoms**	**Prescriptions**			**Lab**		**Outcomes**	

Données structurées

Etudes observationnelles

Motivation : faciliter l'extraction de connaissances à partir des textes cliniques (2/2)

- Les textes cliniques sont complexes
  - Langue spécifique (en langue locale, vocabulaire spécialisé, acronymes, style télégraphique , etc.)
  - Très variables selon le pays, la structure de soin (hôpital, spécialité, auteur, etc.)
  - Difficilement partageables
- Les outils de Traitement Automatique des Langues (TAL)
  - Evoluent rapidement
  - Marchent mieux quand ils sont adaptés à la tâche et au type de textes utilisés
  - Peuvent nécessiter l'utilisation de calculateur hébergés par des tiers



Textes cliniques (non structurés)

←Patients→

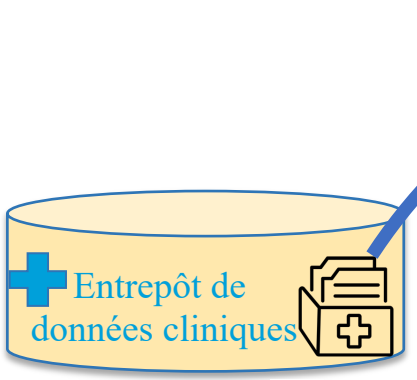
Heart failure	Chest pain	0	0	0	1.7	1	1	0
Heart failure	Hypoxia	21	0	0	3	0	0	0
Stroke	Headache	20	0	4	0	0	0	0
Diagnostics	**Symptoms**	**Prescriptions**			**Lab**		**Outcomes**	

Etudes observationnelles

Données structurées

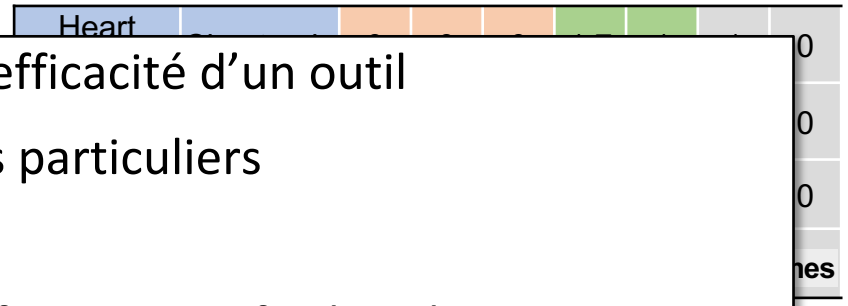
Motivation : faciliter l'extraction de connaissances à partir des textes cliniques (2/2)

- Les textes cliniques sont complexes
  - Langue spécifique (en langue locale, vocabulaire spécialisé, acronymes, style télégraphique , etc.)
  - Très variables selon le pays, la structure de soin (hôpital, spécialité, auteur, etc.)
  - Difficilement partageables
- Les outils de Traitement Automatique des Langues (TAL)
  - Evoluent rapidement
  - Marchent mieux quand ils sont adaptés à la tâche et au type de textes utilisés
  - Peuvent nécessiter l'utilisation de calculateur hébergés par des tiers



Textes cliniques (non structurés)

Outils de TAL



⇒Difficile de prévoir l'efficacité d'un outil sur un type de textes particuliers

⇒Besoin d'outils spécifiques pour faciliter la comparaison

→ *Etudes observationnelles*

Les approches de l'état de l'art

- Constitution de "gros entrepôts", ex : l'EDS de l'AP-HP
- Approches fédérées
- Entraînement sur données ouvertes et affinage sur données locales
- Dans tous les cas : besoin de reproduire et comparer les performance
  - vide dans le domaine de l'informatique médicale
  - *vs.* les outils de gestions de workflow en bioinformatique (Galaxy, SnakeMake)

Méthode : Encapsulation d'outils de TAL pour le développement de *pipelines* reproductibles

- Illustration de cette approche avec le développement de *pipelines* pour un banc d'essai de
  - 4 outils de reconnaissance d'entités nommées (REN)
  - 3 corpus de textes cliniques
- Avec la bibliothèque *med*kit

4 outils de REN

1. UMLSMatcher : l'approche dictionnaire
   - Input1 : le fichier MRCONSO.RRF de l'UMLS (ou le votre qui suit la même structure)
   - Input2 : les groups sémantiques de l'UMLS
   - Algo : similarité floue

2. DrBERT: l'approche Transformer v1 [1]
   - Architecture RoBERTa
   - Entraîné *from scratch* sur le corpus NACHOS (français, biomédical)

3. CamemBERT-bio : l'approche Transformer v2 [2]
   - Architecture RoBERTa
   - Entraîné sur le corpus OSCAR fr (français, domaine général), ajusté finement sur biomed-fr (qui inclut une partie de E3C)

4. GPTMatcher : l'approche générative
   - GPT3.5 Turbo
   - SpaCy-llm
   - *Chain of thoughts*
   - *Few shots* : 12 phrases annotés manuellement de QUAERO

[1] Labrak *et al*. DrBERT : A robust pre-trained model in French for biomedical and clinical domains. In ACL, 2023.
[2] Touchent *et al.* CamemBERT-bio : a tasty French language model better for your health. arXiv preprint, 2023

3 corpus de textes cliniques en français

1. QUAERO [3]
   - Titres d'article MEDLINE et notices de médicaments
   - Annotés manuellement
   - Types annotés : UMLS Semantic groups: Chemical and Drugs, Disorder, Procedures et d'autres
2. CAS-M2
   - Cas cliniques du corpus CAS [4]
   - Annotés manuellement par des étudiants en M2
   - Types annotés : problème, test et traitement
3. E3C [5]
   - Cas cliniques en cinq langues, ici seulement le sous-ensemble français
   - Annotés manuellement, semi- et automatiquement, ici seulement le sous-ensemble manuellement annoté
   - Type annoté : Disorder

[3] Névéol, *et al*. The QUAERO French medical corpus : A ressource for medical entity recognition and normalization. In *Proc. of BioText-Mining Work,* 2014.
[4] Grabar, *et al*. CAS : corpus of clinical cases in French. *Journal of Biomedical Semantics*, 2020.
[5] Magnini *et al.* The E3C project : European clinical case corpus. *Language*, 2021.

La bibliothèque *med*kit (1/2)

Why medkit ?

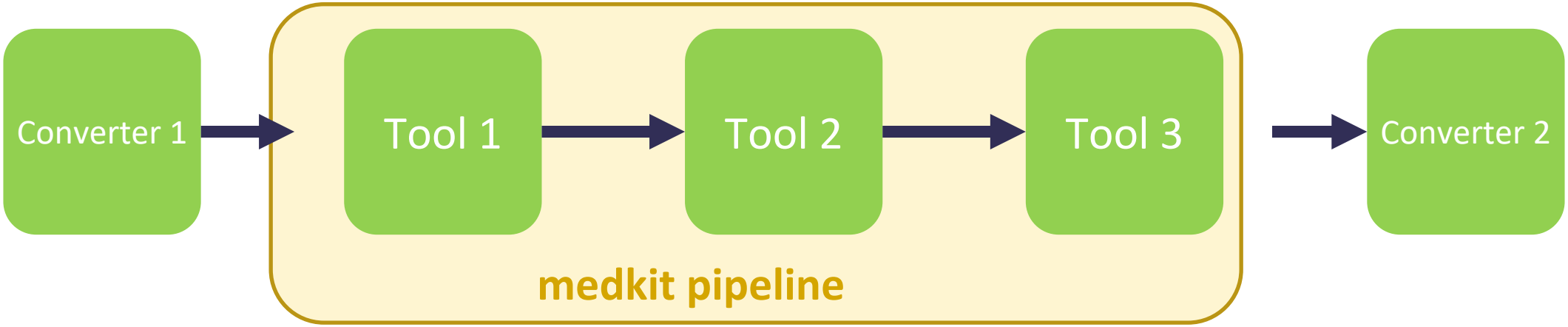
To encapsulate

and chain **tools**

…to facilitate reuse and comparison

What tools?

- Preprocessing
- Extraction
- Classification
- Evaluation …



La bibliothèque *med*kit (1/2)

Why medkit ?

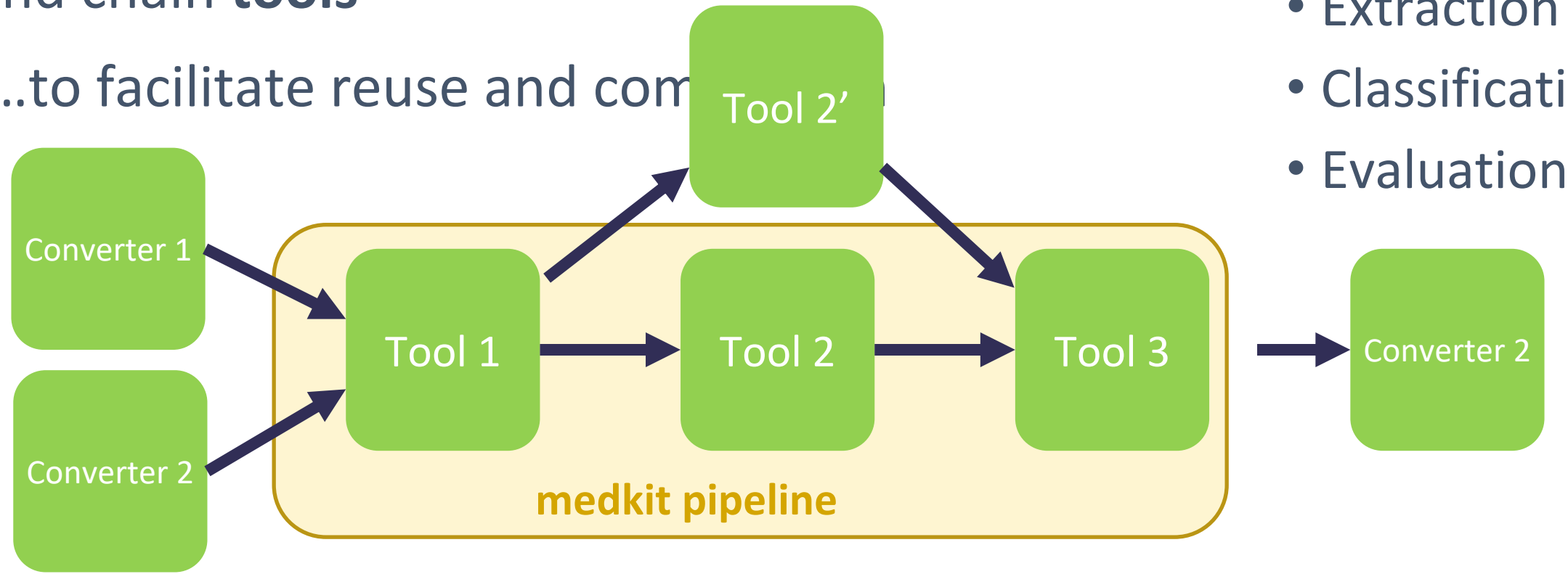
To encapsulate

and chain **tools**

…to facilitate reuse and com

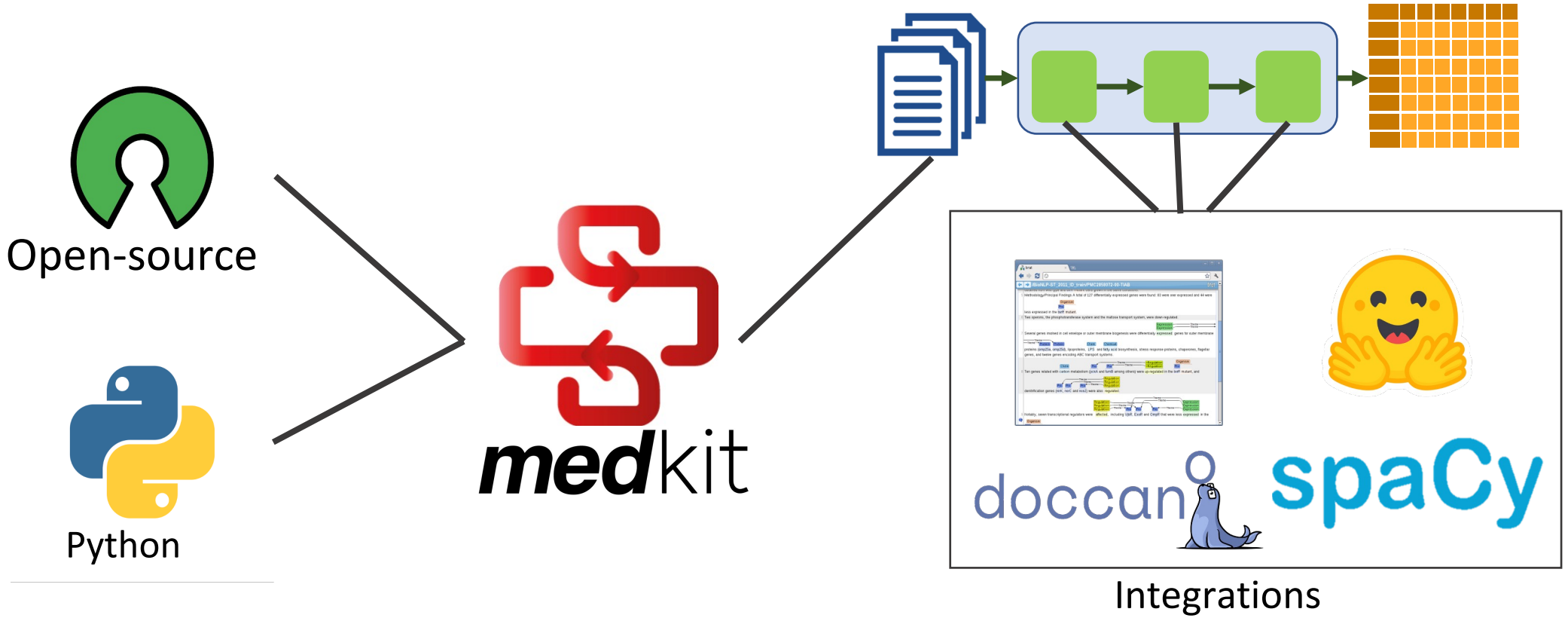
What tools?

- Preprocessing
- Extraction
- Classification
- Evaluation …



La bibliothèque *med*kit (1/2)

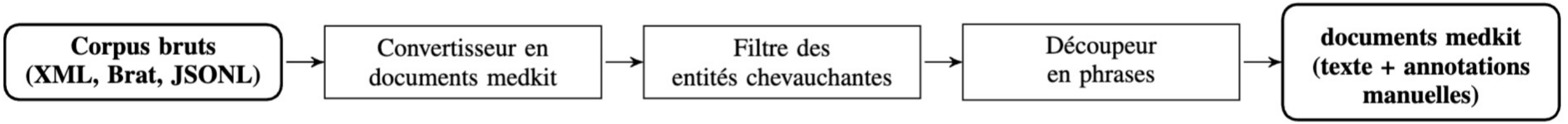
Reuse, reuse, reuse



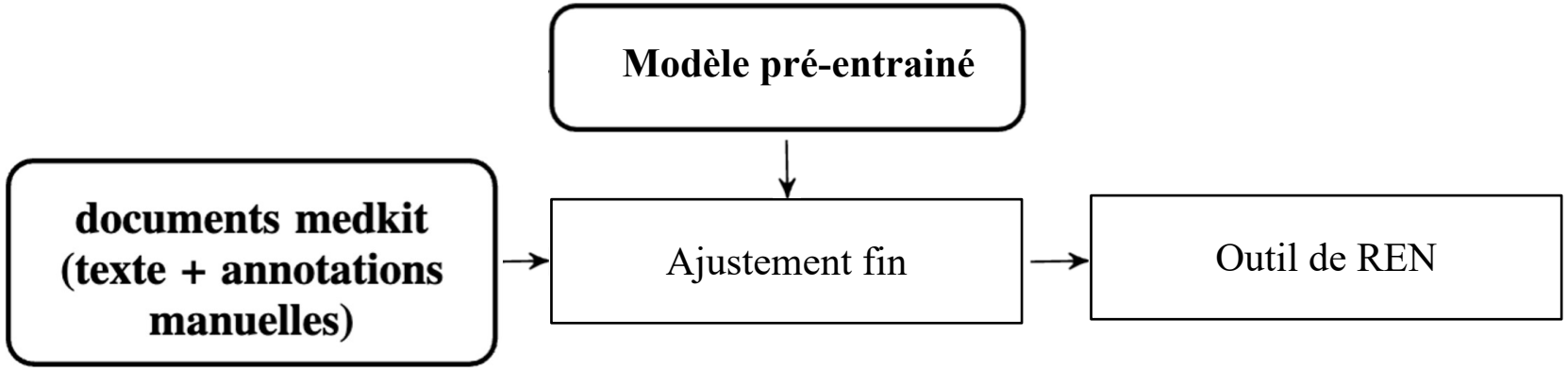
medkit-lib/medkit

Les *pipelines (1/3)*

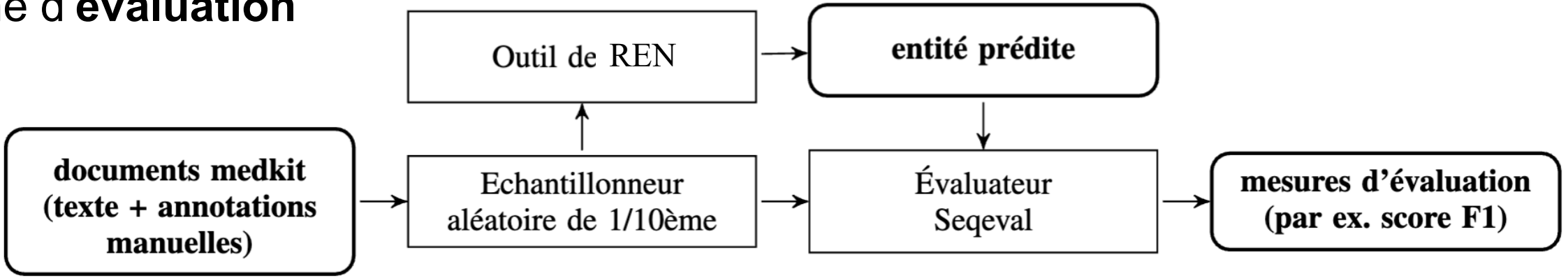
1. Pipeline de **prétraitement**



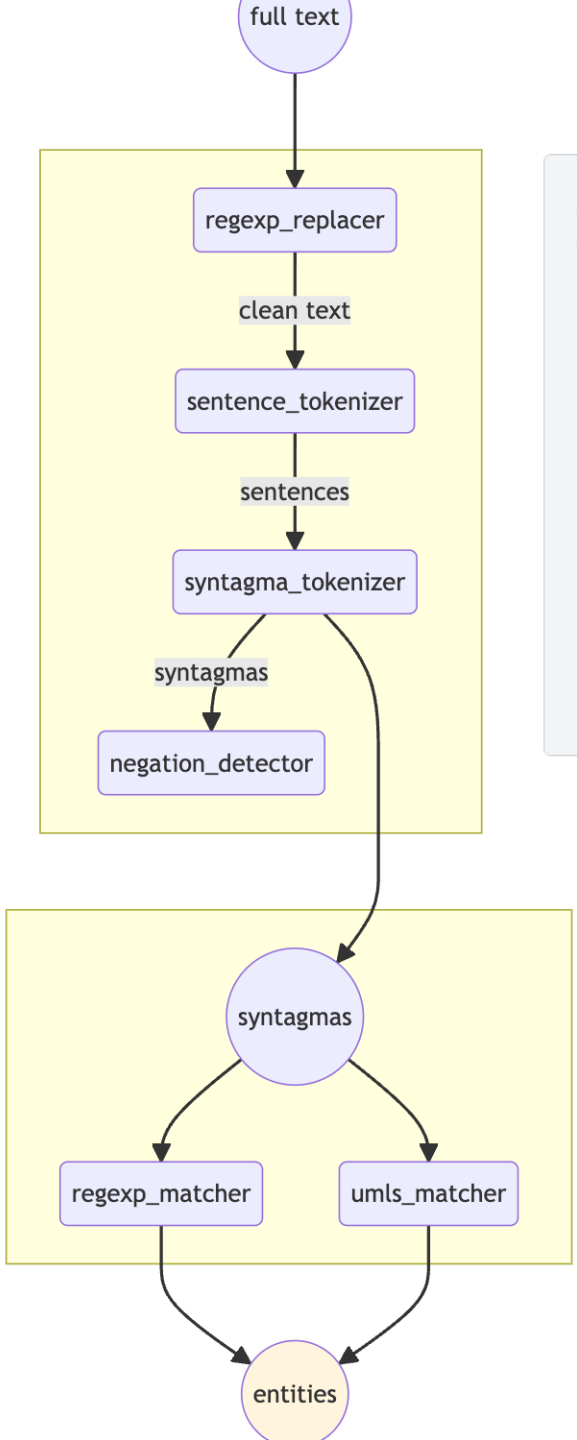
2. Pipeline d'**entraînement** (pour DrBERT et CamemBERT-bio)



3. Pipeline d'**évaluation**



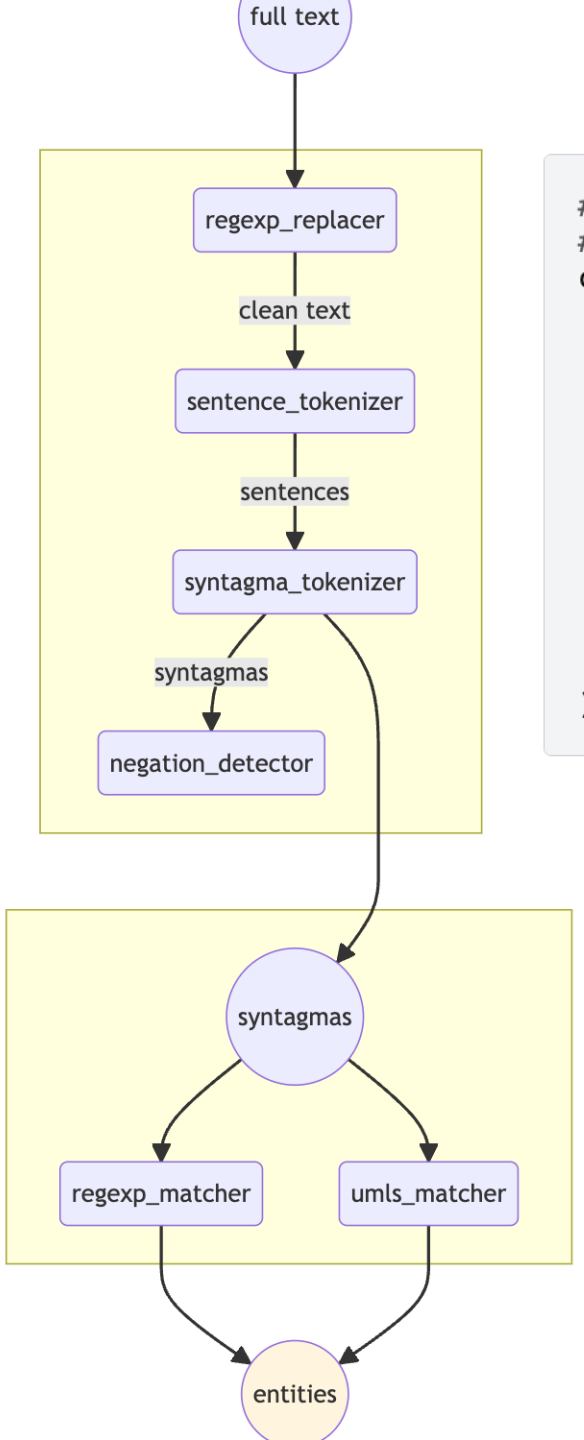
Les *pipelines (2/3)*




```python
# Context pipeline that receives full text segments
# and returns preprocessed syntagmas segments with negation attributes.
context_pipeline = Pipeline(
    # Optional name to indicate task performed by a pipeline
    # (will be used in provenance data)
    name="context",
    steps=[
        PipelineStep(regexp_replacer, input_keys=["full_text"], output_keys=["clean_text"]),
        PipelineStep(sentence_tokenizer, input_keys=["clean_text"], output_keys=["sentences"
        PipelineStep(syntagma_tokenizer, input_keys=["sentences"], output_keys=["syntagmas"]
        PipelineStep(negation_detector, input_keys=["syntagmas"], output_keys=[]),
    ],
    input_keys=["full_text"],
    output_keys=["syntagmas"],
)
```

```python
from medkit.text.ner import UMLSMatcher

umls_matcher = UMLSMatcher(
    umls_dir="../data/umls/2021AB/META/",
    language="FRE",
    cache_dir=".umls_cache/",
    attrs_to_copy=["is_negated"],
)

# NER pipeline that receives syntagmas segments
# and return entities matched by 2 different operations
ner_pipeline = Pipeline(
    name="ner",
    steps=[
        PipelineStep(regexp_matcher, input_keys=["syntagmas"], output_keys=["entities"]),
        PipelineStep(umls_matcher, input_keys=["syntagmas"], output_keys=["entities"]),
    ],
    input_keys=["syntagmas"],
    output_keys=["entities"],
)
```

Les *pipelines (2/3)*




```
# Context pipeline that receives full text segments
# and returns preprocessed syntagmas segments with negation attributes.
context_pipeline = Pipeline(
    # Optional name to indicate task performed by a pipeline
    # (will be used in provenance data)
    name="context",
    steps=[
        PipelineStep(regexp_replacer, input_keys=["full_text"], output_keys=["clean_text"]),
        PipelineStep(sentence_tokenizer, input_keys=["clean_text"], output_keys=["sentences"
        PipelineStep(syntagma_tokenizer, input_keys=["sentences"], output_keys=["syntagmas"]
        PipelineStep(negation_detector, input_keys=["syntagmas"], output_keys=[]),
    ],
    input
    outpu
)
```

```
pipeline = Pipeline(
    steps=[
        PipelineStep(context_pipeline, input_keys=["full_text"], output_keys=["syntagmas"]),
        PipelineStep(ner_pipeline, input_keys=["syntagmas"], output_keys=["entities"]),
    ],
    input_keys=["full_text"],
    output_keys=["entities"],
)
```

```
from me

umls_ma
    uml
    lan
    cac
    attrs to copy=["is_negated"],
)

# NER p
# and r
ner_pip
    nam
    ste

    ],
    inp
    out
)
```

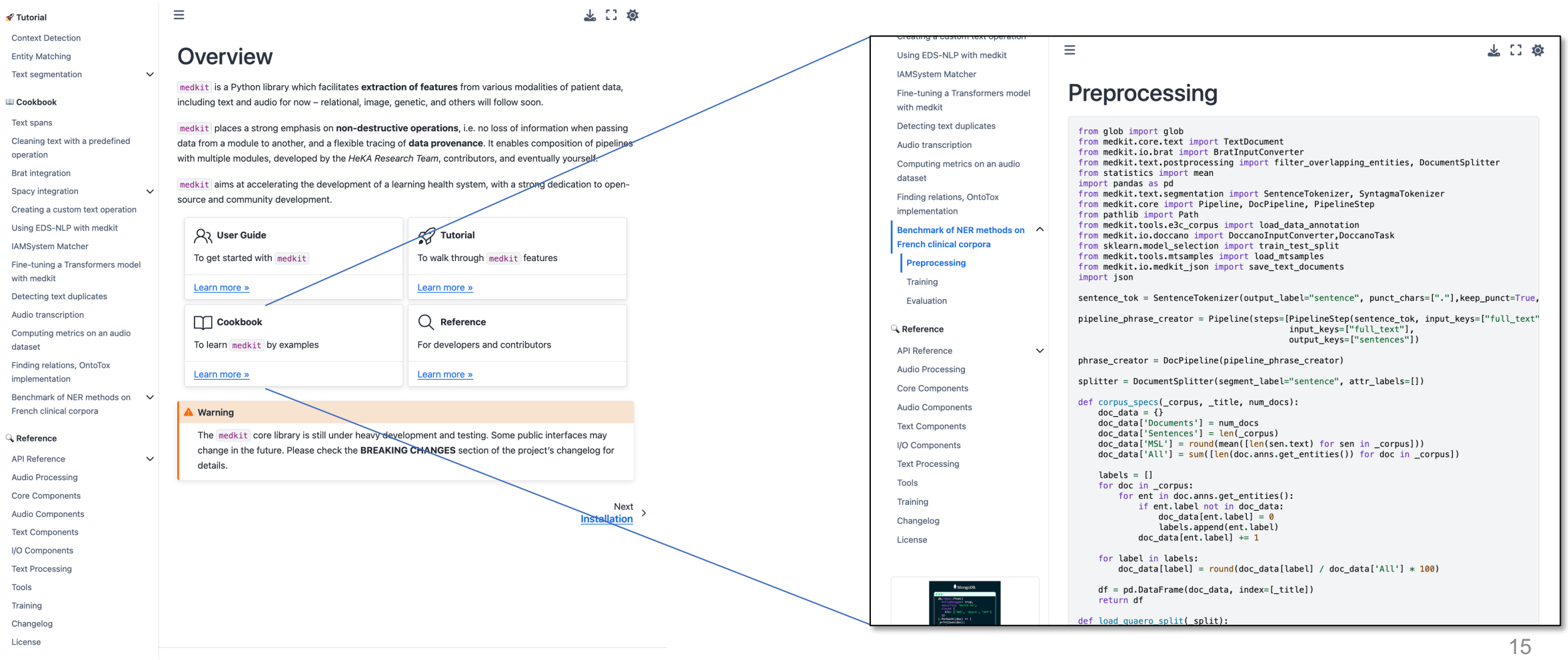
```
entities = pipeline.run([doc.raw_segment])

for entity in entities:
    neg_attr = entity.attrs.get(label="is_negated")[0]
    print(entity.label, ":", entity.text)
    print("negation:", neg_attr.value, end="\n\n")
```

Volontairement
inspiré de `scikit learn`

Les *pipelines (3/3)*

Les pipelines sont disponibles :

- Code: https://github.com/medkit-lib/medkit
- Doc: https://medkit.readthedocs.io/



Résultats (1/2)

	QUAERO	CASM2	E3C	Moyenne
UMLS matcher	0,48 ± 0,02	0,31 ± 0,02	**0,61 ± 0,03**	0,41
DrBERT spécifique	0,42 ± 0,01	0,41 ± 0,02	0,4 ± 0,04	0,41
DrBERT générique	0,44 ± 0,02	0,42 ± 0,02	0,43 ± 0,04	0,43
CamemBERT-bio spécifique	0,57 ± 0,02	0,57 ± 0,0	0,56 ± 0,02	0,57
CamemBERT-bio générique	**0,59 ± 0,02**	**0,58 ± 0,01**	0,52 ± 0,04	**0,58**
GPT matcher	0,52 ± 0,03	0,34 ± 0,03	0,55 ± 0,04	0,43

TABLE 2 – Scores F1 pondérés, par outil de NER et par ensemble d'entraînement de corpus.

1. Relativement bas
2. CamemBERT-Bio > DrBERT (cf.[6])
3. Générique > spécifiques
4. Hétérogène

	Chemical		Disorder		Procedure	
	QUAERO	CASM2	QUAERO	CASM2	QUAERO	CASM2
UMLS matcher	0,55 ± 0,06	0,35 ± 0,05	**0,58 ± 0,05**	0,32 ± 0,04	0,29 ± 0,06	0,25 ± 0,05
DrBERT générique	0,67 ± 0,05	0,43 ± 0,06	**0,58 ± 0,03**	0,4 ± 0,03	0,58 ± 0,04	0,45 ± 0,05
CamemBERT-bio gén,	**0,69 ± 0,04**	**0,66 ± 0,07**	0,55 ± 0,05	**0,61 ± 0,04**	**0,6 ± 0,08**	**0,67 ± 0,03**
GPT matcher	0,62 ± 0,04	0,33 ± 0,05	**0,58 ± 0,03**	0,42 ± 0,04	0,4 ± 0,02	0,24 ± 0,02

TABLE 3 – Scores F1 par type d'annotations sur les corpus QUAERO et CASM2. Le corpus E3C a un seul type d'annotation et pour cette raison, les performance pour ce type unique sont celles rapportées dans le Tableau 2.

[6] Naguib *et al.* Few shot clinical entity recognition in three languages: Masked language models outperform LLM prompting, arXiv, 2024

Résultats (2/2)

- Les pipelines développés avec *med*kit

  https://medkit.readthedocs.io/en/stable/cookbook/ner_benchmark/index.html

- “Facile à réutiliser” :
  - Notre volonté, mais reste subjectif
  - 1 exemple (qui illustre seulement) : Thibault, étudiant en 4ème année de médecine, en stage pendant 6 semaines
  - Quelques retours des utilisateurs de medkit

Discussion

- Limites des résultats de pipelines
  - Ajustement des hyperparamètres
  - Portés de la comparaison : Différence entre *Chemical and Drugs* et traitement

- Ne remplace les bibliothèques et portails de NLP (HF, SpaCy, ens-nlp, etc.)
  - Se place à un niveau différent
  - Encapsule pour chaîner, évaluer, comparer

- 2 différences :
  - *Processing* non destructif
  - Tracabilité de la provenance

Want to try medkit ?

- Code: https://github.com/medkit-lib/medkit
- Doc: https://medkit.readthedocs.io/
- Contact: medkit-maintainers@inria.fr

- **Communité:**

  utilisé à l' *Hopital Georges Pompidou*, *Necker*, *CHU Strasbourg*, *Centre Eugène Marquis*, et testé au *CHU Nancy*, *AP-HM*, *Bichat*, *Fondation Rothshild*

Questions?

medkit

HeKA

Inria

Inserm

Université Paris Cité

Details des corpus

	QUAERO			CASM2			E3C		
	train	**val**	**test**	**train**	**val**	**test**	**train**	**val**	**test**
Documents	844	844	848	424	106	133	36	18	45
Phrases	1 569	1 514	1 454	6 320	1 613	2 173	509	241	642
LMP (cara.)	96	92	93	140	140	136	146	149	139
Entités (#)	**4 513**	**4 121**	**4 084**	**14 566**	**3 628**	**4 744**	**596**	**272**	**731**
Disorder (%)	29	25	24	48	46	47	100	100	100
Chemical (%)	21	25	25	23	26	23			
Procedure (%)	20	18	19	29	28	29			
Autres (%)	30	32	32						

TABLE 1 – Taille et contenu des trois corpus. LMP est la longueur moyenne des phrases en nombre de caractères.