

Rapport du Projet Ingénierie cognitive



RÉALISÉ PAR :

Alaoui Mdaghri Ahmed

ENCADRÉ PAR :

Mme. Maryem RHANOU

Mme. Mounia MIKRAM



Table des matières

03

Introduction

04

Description du sujet

05

Sources de données

06

L'annotation et son application

15

Conclusion

Introduction

La recherche d'emploi fait face à un changement avec l'apparition de réseaux sociaux professionnels à titre d'exemple LINKEDIN. Ces plateformes sont spécialement conçues pour la communauté des affaires, offrant aux utilisateurs la possibilité de documenter leur propre réseau de contacts en mettant en avant leurs compétences, leur expérience, leurs postes précédents et les opportunités de recrutement disponibles.

Description du sujet

Dans notre cas, nous utiliserons LinkedIn et Wikipédia comme sources de données pour analyser le marché du travail au Maroc, en se concentrant sur le domaine spécifique de la science des données. À cet effet, nous annoterons les termes spécifiques à ce domaine en identifiant les compétences requises à partir des données extraites de Wikipédia. De plus, chaque poste récupéré sur LinkedIn nous permettra d'obtenir la description du travail, ce qui nous aidera à mieux nous informer. En utilisant ces informations, nous pourrons ensuite nous former en tenant compte des compétences techniques actuellement demandées sur le marché.

Sources de données

Nos données seront scrappées en totalité du web :

-LinkedIn pour alimenter notre base de données relative au marché de travail au Maroc avec l'intitulé du job et sa

description	Job_title	Company	Location	Link	Description
0	Data Scientist	Société Générale Maroc	Casablanca, Casablanca-Settat, Morocco	https://ma.linkedin.com/jobs/view/data-scienti...	Le Data Scientist a pour mission la conception...
1	Data Scientist-(H/F)	Société Générale Maroc	Casablanca, Casablanca-Settat, Morocco	https://ma.linkedin.com/jobs/view/data-scienti...	Vos missions au quotidienLe Data Scientist a p...
2	AI Machine Learning Optimization Engineer H/F	Stellantis	Casablanca, Casablanca-Settat, Morocco	https://ma.linkedin.com/jobs/view/ai-machine-l...	Stellantis' AI team is expanding and o...
3	Data Scientist	Kenz'up	Casablanca, Casablanca-Settat, Morocco	https://ma.linkedin.com/jobs/view/data-scienti...	About Kenz'upJoin the new moroccan tech startu...

-Wikipédia pour les différentes annotations (langages de programmations, les plateformes, les bases de données et les frameworks).

languages.head()	
	name
0	A.NET (A#/A sharp)
1	A-0 System
2	A+ (A plus)
3	ABAP
4	ABC

frameworks_tools	
	name
0	TensorFlow
1	PyTorch
2	Spark MLlib
3	Torch
4	Transformers

platforms.head()	
	name
0	Alibaba Cloud (1 P)
1	Amazon Web Services (42 P)
2	Cloud databases (11 P)
3	Google Cloud (20 P)
4	IBM cloud services (13 P)

databases.head()	
	name
0	Oracle Database
1	MySQL
2	Microsoft SQL Server
3	PostgreSQL (free software)
4	IBM Db2

L'annotation et son application

Après extraction des données déjà scrapper on peut en construire des patterns et les ajouter au modèle nlp «en_core_web_sm » et l'enregistrer pour utilisation ultérieure.

```
languages = pd.read_csv('languages.csv')
platforms = pd.read_csv('platforms.csv')
databases = pd.read_csv('databases.csv')
frameworks_tools = pd.read_csv('frameworks_tools_etc.csv')

patterns = []
for x in languages.name.tolist():
    patterns.append({"label": "PROG_LANG", "pattern": x, "id": "SKILLS"})

for x in databases.name.tolist():
    patterns.append({"label": "DB", "pattern": [{"lower": w.lower()} for w in str(x).split()], "id": "SKILLS"})

for x in platforms.name.tolist():
    patterns.append({"label": "PLATFORM", "pattern": [{"lower": w.lower()} for w in str(x).split()], "id": "SKILLS"})

for x in frameworks_tools.name.tolist():
    patterns.append({"label": "FRAMEWORKS", "pattern": [{"lower": w.lower()} for w in str(x).split()], "id": "SKILLS"})

nlp = spacy.load("en_core_web_sm")
ruler = nlp.add_pipe("entity_ruler", before="ner")
ruler.add_patterns(patterns)
ruler.to_disk("patterns.jsonl")
```

L'application du modèle préétablie on peut extraire les entités d'une description.

Technologies ORG Python PROG_LANG

PyTorch FRAMEWORKS TensorFlow FRAMEWORKS NumPy FRAMEWORKS Pandas FRAMEWORKS Keras FRAMEWORKS JAX GPE Julia PROG_LANG

PostgreSQL MySQL DB Snowflake DB GCP/AWS

Java PROG_LANG

Job requirements

3+ years DATE experience with applied Machine/Deep Learning ORG

Understanding of data structures, data modeling, and software architecture.

Strong coding experience and proven ability to implement ML PROG_LANG packages and neural network architectures using frameworks such as PyTorch FRAMEWORKS or Tensorflow FRAMEWORKS

Great communication skills and an ability to translate business context into data-oriented hypotheses to drive impact.

BSc or BA NORP degree in computer science, data science, applied math, or a related field.

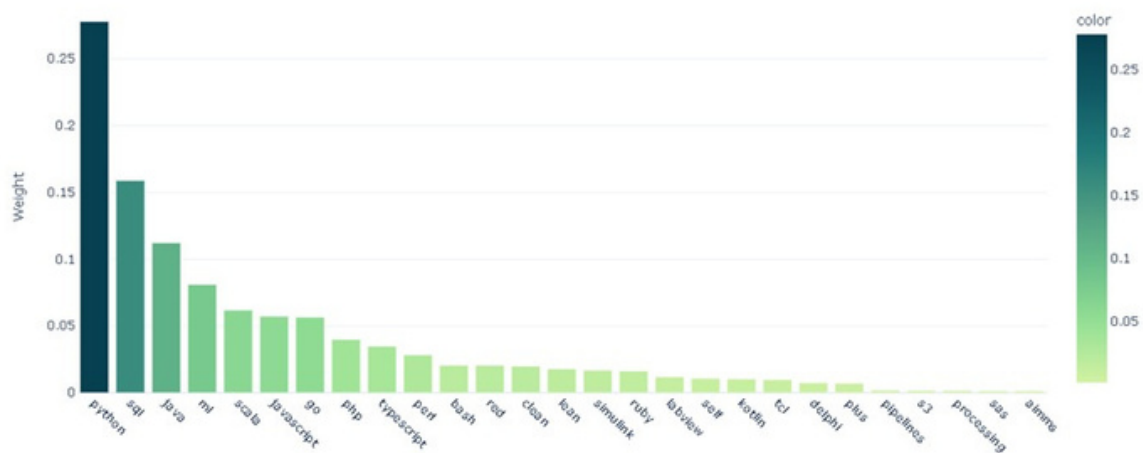
An appreciation for the connection between the product experience and models you build.

Avec une boucle for on pourra par la suite appliquer cela sur l'ensemble des descriptions déjà scrapper de Linkedin et stocker les résultats sous format de DataFrame comme suit :

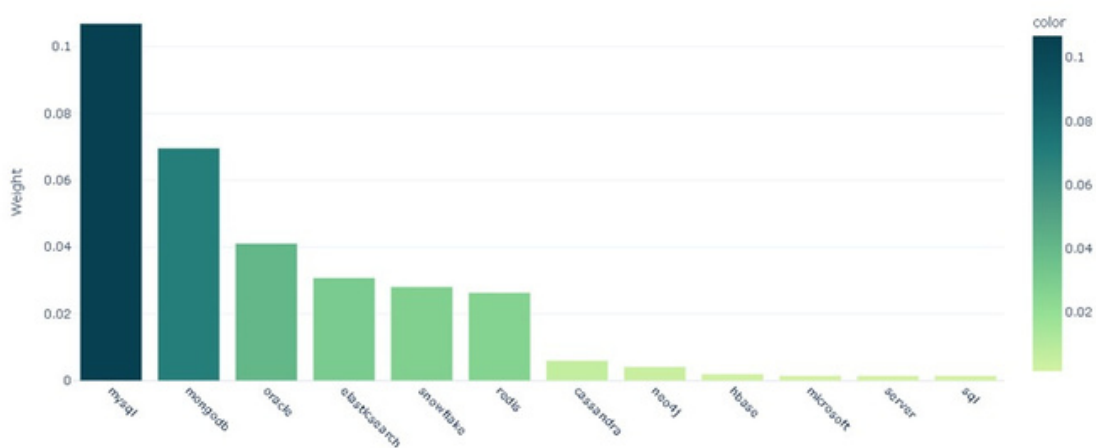
	skills_prog_langs	skills_platform	skills_databases	skills_frameworks
0				
1				
2	Python			Tensorflow Pytorch
3	Go ML SQL Python Scala B	Amazon Sagemaker		TensorFlow PyTorch
4	Python R SQL			
...
363	Lean Kotlin Ruby Go Python	Amazon Web Services		
364				
365	Lean Kotlin Ruby Go Python	Amazon Web Services		
366	Lean			
367	Delphi SQL			
368 rows × 4 columns				

Pour finaliser notre but était de remonter les outils les plus demandé dans le marché de travail marocain et c'est ce qu'on va visualiser pour chaque entité comme suit :

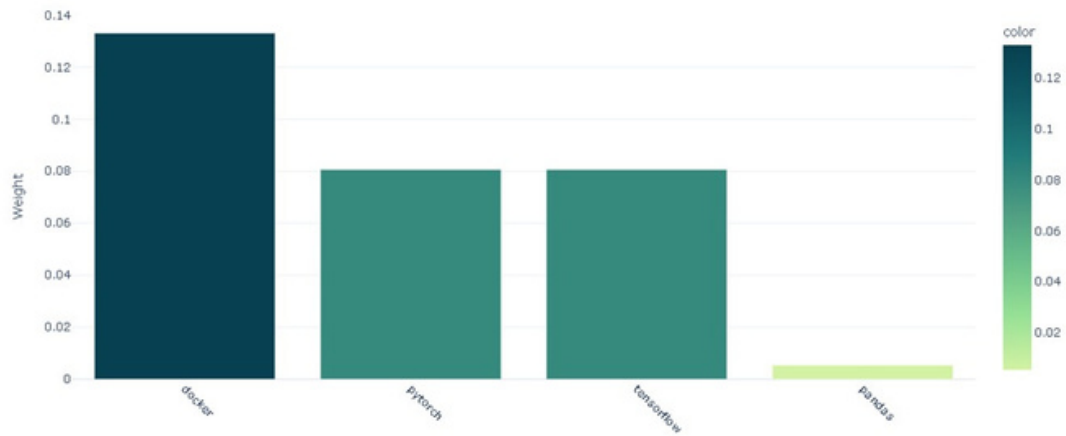
- Pour les langages de programmation :



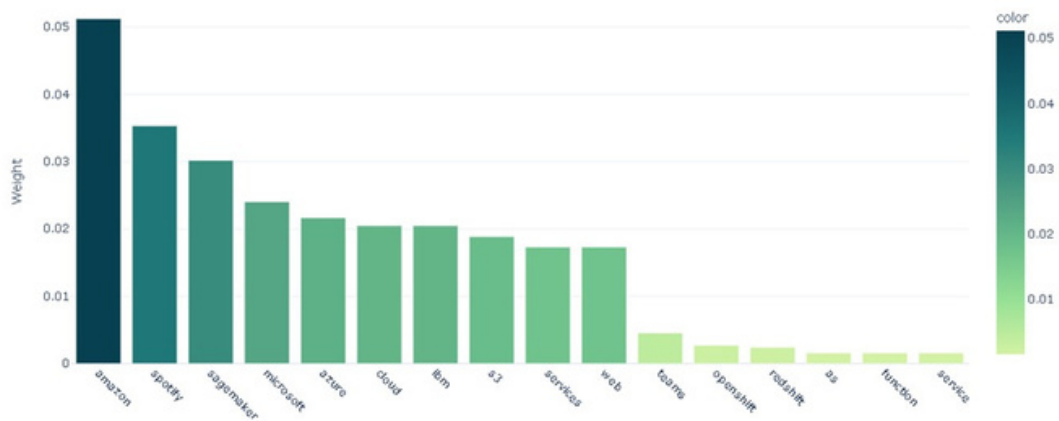
- Pour les bases de données :



-Pour Frameworks :



-Pour les plateformes :



-Vous pouvez consulter le code du projet sur ce lien !
https://github.com/AlaouiMdaghriAhmed/NER_data_jobs/

Ceci était une étape d'annotation automatique, on pourra par la suite passer par doccano pour enrichir notre annotation et entrainer le modèle pré entrainer "en-core-web-sem" .

Annotation sur doccano :

```
knowledge, inference engine knowledge Proficient in Python,
•PROG_LANG

C++ Good communication skills in English languagePreferred
•PROG_LANG

Qualification Sound knowledge of at least one deep learning
framework such as Tensorflow, Pytorch Knowledge or prior
•FRAMEWORKS
•FRAMEWORKS
```

Et puis exporter notre données annoter comme fichier jsonl sous comme suit :

```
{
  "id": 375,
  "text": "Stellantis' AI team is expanding and offers exciting job opportunities for exceptional candidates to contribute to AI services and infrastructure knowledge, inference engine knowledge Proficient in Python, C++ Good communication skills in English languagePreferred Qualification Sound knowl",
  "entities": [
    {
      "start": 115,
      "end": 145,
      "label": "PROG_LANG",
      "score": 0.95
    },
    {
      "start": 155,
      "end": 185,
      "label": "PROG_LANG",
      "score": 0.95
    },
    {
      "start": 195,
      "end": 225,
      "label": "FRAMEWORKS",
      "score": 0.95
    },
    {
      "start": 235,
      "end": 265,
      "label": "FRAMEWORKS",
      "score": 0.95
    }
  ]
}
```

On pourra par la suite faire notre training sur cette dataset

```
nlp = spacy.load(model)
print("Loaded model '%s'" % model)

# Add entity recognizer to model if it's not in the pipeline
# nlp.create_pipe works for built-ins that are registered with spaCy
if "ner" not in nlp.pipe_names:
    ner = nlp.create_pipe("ner")
    nlp.add_pipe(ner)
    print("Created NER model")
# otherwise, get it, so we can add labels to it
else:
    ner = nlp.get_pipe("ner")
    print("Got NER model")

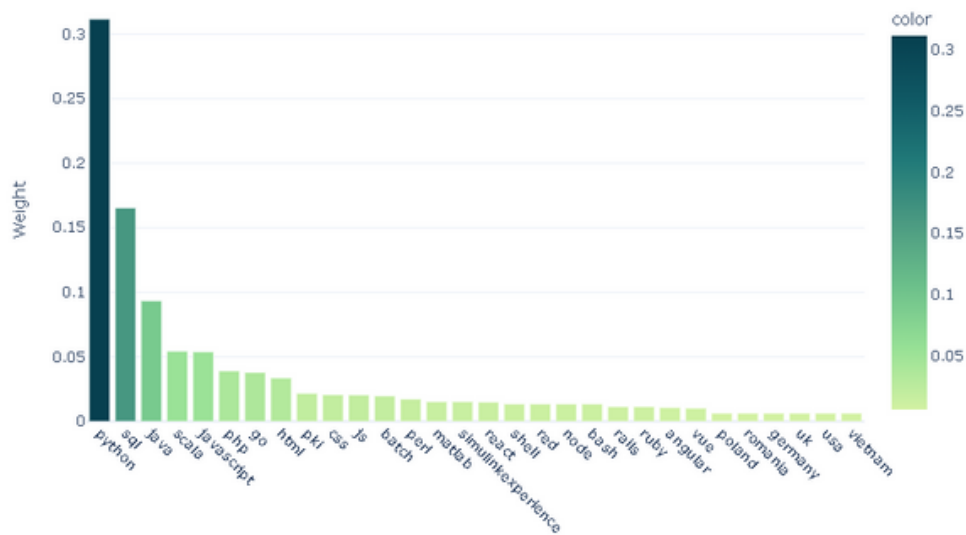
# Add new labels to entity recognizer
for label in LABELS:
    ner.add_label(label)

# We assume an existing model modification
optimizer = nlp.resume_training()

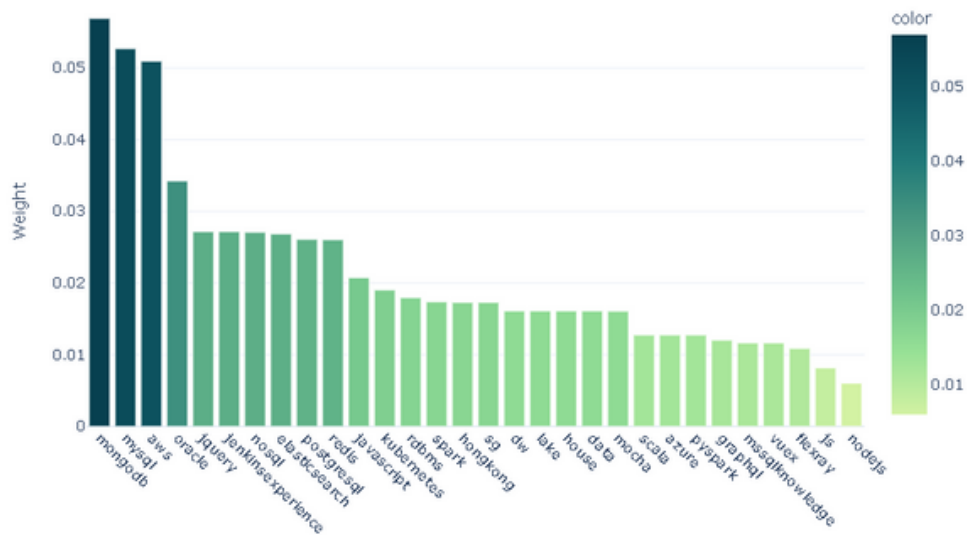
move_names = list(ner.move_names)
# Get names of other pipes to disable them during training
pipe_exceptions = ["ner", "trf_wordpiecer", "trf_tok2vec"]
```

Encore une fois on pourra visualiser les différents outils pour ce modèle :

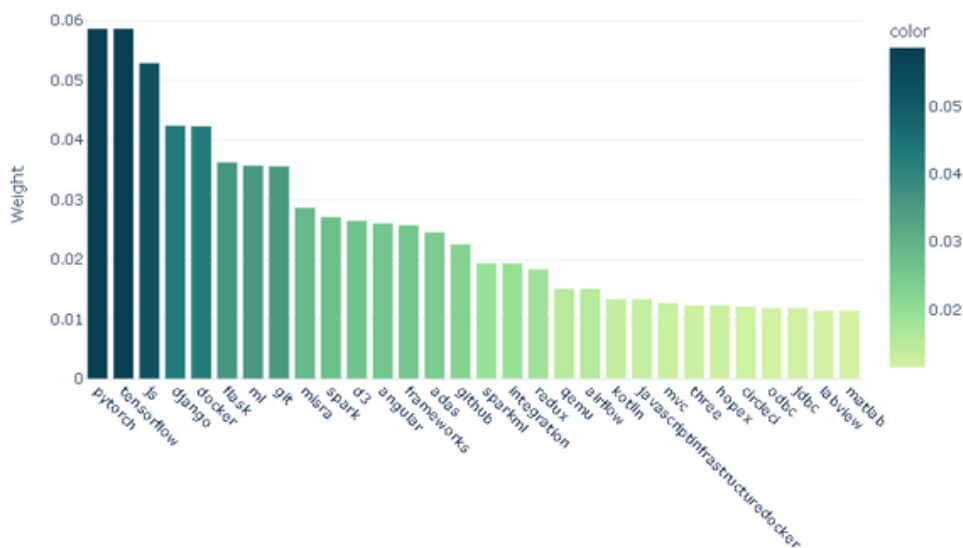
- Pour les langages de programmation :



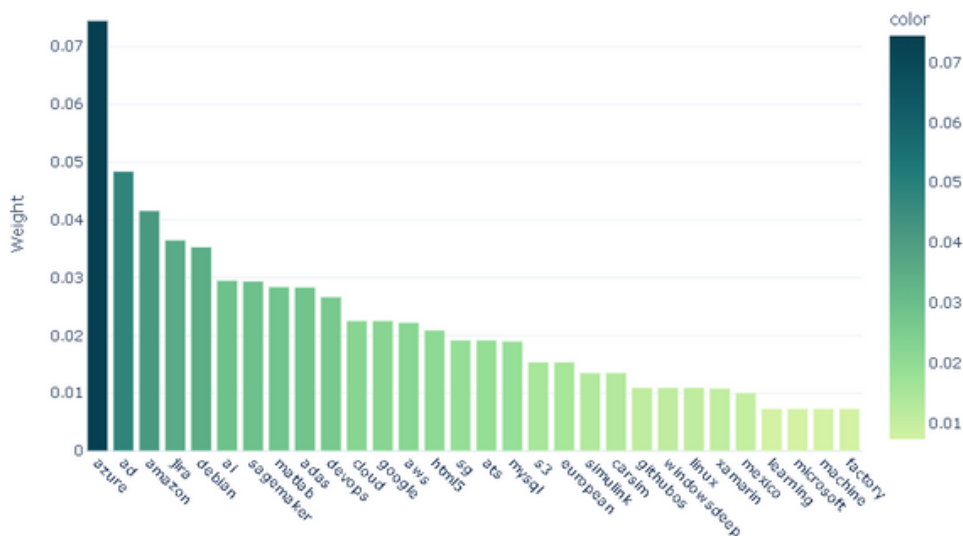
- Pour les bases de données :



- Pour les frameworks :



- Pour les plateformes :



On peut observer que particulièrement pour les frameworks, nous avons amélioré le modèle en ajoutant plusieurs autres annotations à l'aide de Doccano. De plus, nous avons effectué un entraînement en utilisant les données déjà annotées. Cette approche a considérablement contribué à la création d'un modèle amélioré, voire performant, bien que des améliorations supplémentaires soient envisagées en utilisant d'autres modèles pré-entraînés tels que BERT.

Dans cette partie on va essayer d'implémenter un Bert based model fournit par simpletransformers sur notre dataset post annotation avec doccano.

Un défi rencontré à ce stade est de formater nos données pour les insérer dans le modèle. Selon la documentation de NERmodel de SimpleTransformers, nous devons fournir au modèle des données d'entraînement dans le format suivant : (id_sentence, words, labels).

On instancie dans un premier lieu les arguments du modèle et les labels à utiliser :

```
[146] label = df["labels"].unique().tolist()
      args = NERArgs()
      args.num_train_epochs = 1
      args.learning_rate = 1e-4
      args.overwrite_output_dir = True
      args.train_batch_size = 32
      args.eval_batch_size = 32
```

▶ label

↳ ['PROG_LANG', 'FRAMEWORKS', 'DB', 'PLATFORM']

On passe après à l'entraînement

```
model = NERModel('bert', 'bert-base-cased', labels=label, args=args, use_cuda=False)
model.train_model(train_data, eval_data=test_data, acc=accuracy_score)
```

Downloading: 100% ██████████ 436M/436M [00:11<00:00, 52.7MB/s]

Some weights of the model checkpoint at bert-base-cased were not used when initializing BertForTokenClassification
- This IS expected if you are initializing BertForTokenClassification from the checkpoint of
- This IS NOT expected if you are initializing BertForTokenClassification from the checkpoint
Some weights of BertForTokenClassification were not initialized from the model checkpoint at
You should probably TRAIN this model on a down-stream task to be able to use it for predictions

Downloading: 100% ██████████ 213k/213k [00:00<00:00, 1.47MB/s]

Downloading: 100% ██████████ 29.0/29.0 [00:00<00:00, 587B/s]

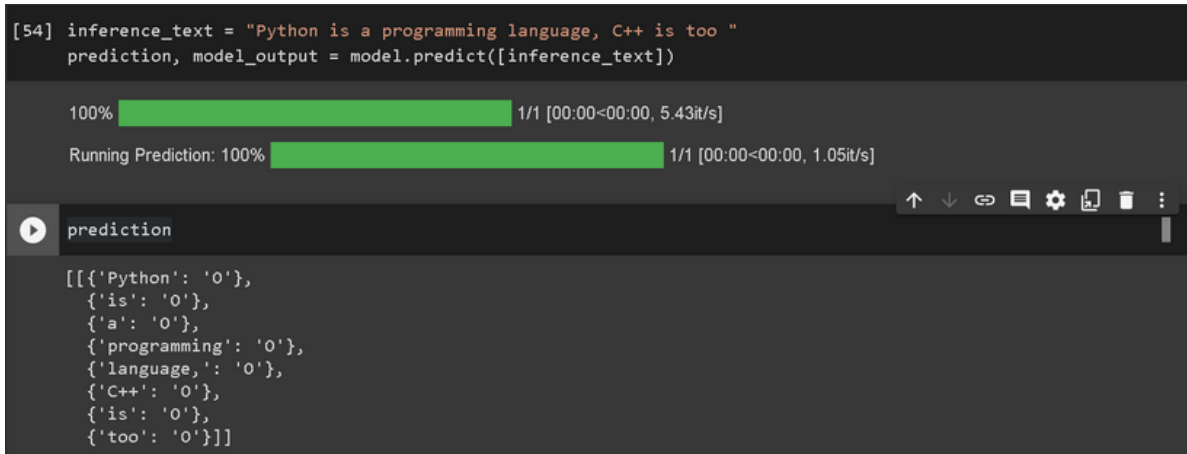
100% ██████████ 1/1 [00:00<00:00, 5.49it/s]



Epoch 1 of 1: 100% ██████████ 1/1 [02:00<00:00, 120.94s/it]

Epochs 0/1. Running Loss: 0.8658: 100% ██████████ 3/3 [01:53<00:00, 34.39s/it]

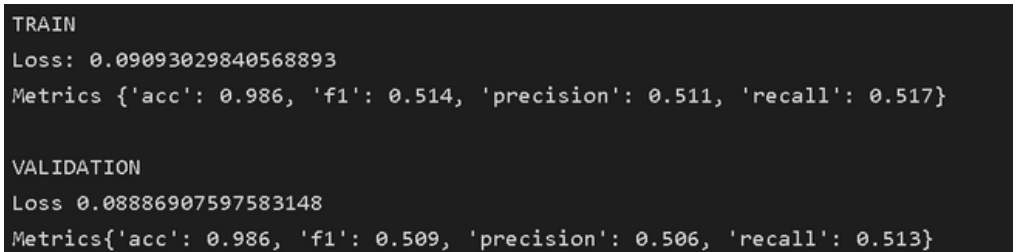
(3, 1.2429930567741394)

Les résultats obtenus lors de l'utilisation de cette méthode se révèlent très insatisfaisants, ne permettant pas de prédire que Python est le langage de programmation le plus utilisé.



```
[54] inference_text = "Python is a programming language, C++ is too "  
prediction, model_output = model.predict([inference_text])  
  
100%  1/1 [00:00<00:00, 5.43it/s]  
Running Prediction: 100%  1/1 [00:00<00:00, 1.05it/s]  
  
prediction  
[[{'Python': '0'},  
 {'is': '0'},  
 {'a': '0'},  
 {'programming': '0'},  
 {'language': '0'},  
 {'C++': '0'},  
 {'is': '0'},  
 {'too': '0'}]]
```

D'où l'initiative d'essayer d'utiliser les modèles fournis par Hugging Face, en optant pour un modèle spécifique qui pourrait prendre en compte la diversité linguistique de notre ensemble de données, à savoir DistilBERT. Le modèle entraîné est disponible dans le fichier `DistilBERT_Training&Tuning.ipynb`, avec les scores suivants :



```
TRAIN  
Loss: 0.09093029840568893  
Metrics {'acc': 0.986, 'f1': 0.514, 'precision': 0.511, 'recall': 0.517}  
  
VALIDATION  
Loss 0.08886907597583148  
Metrics{'acc': 0.986, 'f1': 0.509, 'precision': 0.506, 'recall': 0.513}
```

Sous contrainte de temps, nous n'avons pas pu retester le modèle sur nos données pour révisualiser les différentes entités. L'obstacle principal rencontré lors de cette phase était l'espace RAM limité et la capacité du processeur qui ne pouvait pas maintenir les conditions nécessaires pour exécuter le code.

Pour surmonter cette limitation, nous avons choisi de réduire le nombre d'epochs et le nombre de batches, permettant ainsi à l'algorithme de s'exécuter dans un laps de temps spécifique.

Conclusion

Ce projet nous a permis d'appliquer les connaissances acquises durant notre cursus à l'ESI, telles que l'utilisation de Python dans différents contextes, depuis l'extraction de données sur le Web (scraping) jusqu'à l'établissement d'un modèle pour réaliser la reconnaissance d'entités dans un texte.

Le projet nous apporte également une idée sur la distribution des différentes compétences dans le marché du travail marocain. Cette information pourra nous être utile pour approfondir nos connaissances dans le domaine de la data science, ou encore acquérir les compétences les plus demandées, dans le but de décrocher un entretien.

GITHUB LINK : https://github.com/AlaouiMdaghriAhmed/NER_data_jobs