**Une image contenant dessin humoristique, croquis, art, robot

Description générée automatiquement**

****

**Une image contenant texte, Police, Graphique, capture d’écran

Description générée automatiquement**

**Document Technique : Création d'un Modèle de Reconnaissance d'Entités Nommées (NER) en Wolof**

**Réalisé par :**

* **Mouhamed Ndiaye**
* **Harona Kane**
* **El Hadji Issa Sall**
* **Mouhamed Lamine Mbaye**

**Introduction**

Ce document vise à fournir une feuille de route détaillée pour la création d'un modèle de reconnaissance d'entités nommées (NER) en langue Wolof, en utilisant les technologies suivantes :

* Spacy pour la mise en place du modele NER
* Selenium pour du scraping afin de collecter des articles du journal DEFU WAXU.
* Streamlit pour réaliser une application web du model

1. **Compréhension des Besoins**

L’objectif de ce projet est de réaliser un système de reconnaissance d'entités nommées (NER) en langue Wolof. La reconnaissance d'entités nommées est une tâche cruciale en traitement automatique du langage naturel (NLP) qui consiste à identifier et classer des entités spécifiques telles que des noms de personnes, d'organisations, de lieux, de quantités, etc., dans un texte.

1. **Acquisition des Données :**

Afin de réaliser ce projet nous avons utiliser les données mises à notre disposition par PAS qui ont du faire l’objet de traitement.

1. **Traitement des Données et Annotation**

Le pretraitement des données est une étape cruciale du projet. Les éléments des textes (Token) de nos jeux de données sont classé en :

'O', 'B-LOC', 'I-LOC', 'B-PER', 'I-PER', 'B-ORG', , 'I-ORG', 'B-DATE', 'I-DATE'

Lors du prétraitement nous avons regroupe les classes suivantes en une classe :

'B-PER', 'I-PER' 🡪 'PER'

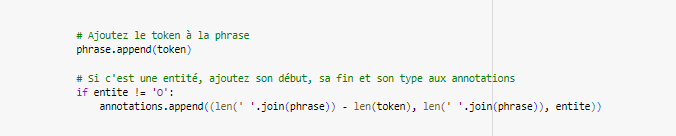
'B-LOC', 'I-LOC' 🡪 'LOC'

'B-DATE', 'I-DATE' 🡪 'DATE'

Une image contenant texte, Police, capture d’écran

Description générée automatiquement

Afin d’adapter les données a notre modele NER nous avons prétaité notre jeux de données en réalisant de la tokenization, suppression de caractères spéciaux, etc.



1. **Mise en place du Modèle NER**

Le modele NER a été initialiser grace spaCy pour entraîner un modèle NER de base avec l'ensemble de données annotées. L’étape que nous avons eu a suivre sont les suivantes :

1. **Importation des bibliothéques :**

* **import spacy** : Importe la bibliothèque spaCy, un outil puissant pour le traitement du langage naturel.
* **import random** : Importe le module random, qui sera utilisé pour mélanger les données d'entraînement.
* **from spacy.training.example import Example** : Importe la classe Example nécessaire pour créer des exemples à partir des données d'entraînement.

1. **Creation du Pipeline :**

* **nlp = spacy.blank("xx")** : Nous a permis de créer un modèle spaCy vide.offre une flexibilité essentielle pour créer des modèles NLP sur mesure, adaptés à des besoins spécifiques, sans être limité par les modèles pré-entraînés existants. C'est un outil précieux pour les tâches de NLP spécialisées et personnalisées.
* **ner = nlp.add\_pipe("ner")** : Nous a permis d’ajouter un composant (pipeline component) à un modèle spaCy. Cela permet de personnaliser le traitement du texte en ajoutant des étapes supplémentaires dans le pipeline de traitement. Les composants qui ont été ajoutés sont :

**ner.add\_label("PER")**

**ner.add\_label("ORG")**

**ner.add\_label("LOC")**

**ner.add\_label("DATE")** : Ceci est l’ajout des étiquettes d'entités personnalisées au NER. Dans notre projet, les entités sont :   
"PER" (Personne), "ORG" (Organisation), "LOC" (Lieu) et "DATE" (Date).

1. **Entrainement du modéle:**

* **optimizer = nlp.begin\_training**() : Nous a permis de prépare le modèle à l'entraînement en initialisant les paramètres et les mécanismes nécessaires pour ajuster les poids du modèle. C'est une étape essentielle pour garantir que le modèle est prêt à apprendre à partir des données d'entraînement.
* **loss\_values = []** : A été utilisé afin de créer une liste vide pour stocker les valeurs de perte au fil du temps. Ces valeurs de perte sont utilisées pour évaluer la performance du modèle pendant l'entraînement.
* **for itn in range(100)** : Débute une boucle pour l'entraînement. Dans notre travail, le modèle sera entraîné sur 100 itérations.
* **random.shuffle(TRAIN\_DATA)** : Mélange les données d'entraînement pour assurer une variabilité pendant l'entraînement.
* **losses = {} :** Initialise un dictionnaire pour stocker les pertes à chaque itération.
* **for batch in spacy.util.minibatch(TRAIN\_DATA, size=32):** Divise les données d'entraînement en mini-batch de taille 32 pour l'entraînement en lot (batch training).
* **texts, annotations = zip(\*batch) :** Dézippe les données du batch en textes et annotations.
* **example = [] :** Initialise une liste vide pour stocker les exemples.
* **for i in range(len(texts)):** Parcours les textes dans le batch.
* **doc = nlp.make\_doc(texts[i]) :** Crée un objet Doc à partir du texte.
* **example.append(Example.from\_dict(doc, annotations[i])) :** Crée un exemple à partir du Doc et des annotations, puis l'ajoute à la liste des exemples.
* **nlp.update(example, drop=0.5, losses=losses) :** Met à jour le modèle avec les exemples créés. Le paramètre drop contrôle le taux de dropout pendant l'entraînement (0.5 signifie un dropout de 50%). Les pertes sont enregistrées dans le dictionnaire.
* **Loss\_values.append(losses["ner"]) :** Ajoute la valeur de perte de la composante NER à la liste des valeurs de perte.
* **print(losses) :** Affiche les pertes à chaque itération.
* **nlp.to\_disk("NER\_fr\_wolof") :** Enregistrement du modele.

1. **Création de l'Interface Utilisateur :**

Pour mettre en application notre modele NER nous avons utilisé Streamlit. Grace à Streamlit nous avons conçu une interface conviviale permettant aux utilisateurs de saisir du texte en Wolof et de voir les entités nommées du texte.

1. **Extraction d’article dans Défu Waxu:**

Pour mettre aux utilisateurs d’avoir une meilleure expérience de l’utilisation de notre modèle, nous avons scrapper des articles depuis le journal Défu Waxu grâce à **Selenium**. Ceci nous a permis extraire différent article sur les thèmes suivants :

* XIBAR
* TAGGAT
* LIFESS

En combinaison avec Streamlit, l’utilisateur pour pourra choisir sur les articles de thème précédemment mentionnés et verra les entités nommées de chaque article.

**Phase 5 : Tests et Déploiement**

**Conclusion**

Ce projet permettra de développer un système de reconnaissance d'entités nommées robuste pour la langue Wolof, avec une interface utilisateur conviviale. Le modèle sera capable d'identifier et de classifier efficacement les entités nommées dans les textes en Wolof, offrant ainsi une solution précieuse pour de nombreuses applications NLP dans cette langue.

Haut du formulaire