Université Hassan II de Casablanca
Faculté des Sciences Ben M'Sick
Département de Mathématiques et Informatique

2024/2025

Rapport du projet De Fin De Module

Extracteur de caractéristiques architecturales dans des descriptions historiques

Licence d'excellence en intelligence artificielle

Encadré par :

Mr.KAICH Oussama

Réalisé par :

Aymen Sabiri
Iksi Badreddine
Sara maktabi

Année Universitaire : 2024-2025

Abstract

The richness of historical texts on architecture offers a valuable resource for understanding built heritage, but their automatic processing remains challenging due to their unstructured nature and multilingual content. This project proposes a simple and interpretable approach to automatically extract three key types of information — architectural elements, materials, and styles — from descriptive paragraphs in French and English.

The method combines classical NLP techniques: language detection, named entity recognition (NER), and extraction using manually defined keyword lists. The system is integrated into an interactive Streamlit interface.

Results show that even without supervised learning, a well-structured rule-based approach can produce relevant extractions in both languages.

Résumé

La richesse des textes historiques sur l'architecture constitue une ressource précieuse pour comprendre le patrimoine bâti, mais leur traitement automatique reste difficile à cause de leur structure non normalisée et de leur multilinguisme. Ce projet propose une approche simple et explicable pour extraire automatiquement trois types d'informations clés — les éléments architecturaux, les matériaux, et les styles — à partir de paragraphes descriptifs en français et en anglais. L'approche combine des techniques classiques de NLP : détection de langue, reconnaissance d'entités nommées (NER) et extraction à l'aide de listes de mots-clés définies manuellement. Le système est intégré dans une interface Streamlit interactive. Les résultats montrent que même sans apprentissage supervisé, une approche bien structurée permet d'obtenir une extraction pertinente dans les deux langues.

Table des matières

CHAPITRE 1 : Contexte Général

1. Introduction

1.1 Contexte et Motivation

La documentation et la préservation du patrimoine architectural sont essentielles pour comprendre l'évolution des sociétés et des identités culturelles. Les textes historiques — tels que les récits de voyage, les fiches patrimoniales ou les traités d'architecture — constituent des sources précieuses qui décrivent en détail les bâtiments, les matériaux, les styles et les éléments architecturaux. Cependant, ces descriptions sont souvent non structurées et rédigées dans différentes langues, ce qui rend leur traitement automatique difficile.

1.2 Objectif

Ce projet vise à développer une approche légère basée sur le Deep Learning et le Traitement Automatique du Langage (TAL ou NLP) pour extraire des informations architecturales clés (éléments, matériaux, styles) à partir de textes historiques en français et en anglais.

2. Travaux Connexes

L'extraction d'informations à partir de textes patrimoniaux et historiques est un domaine qui bénéficie des avancées en traitement automatique du langage naturel (TAL/NLP). Plusieurs approches ont été explorées dans la littérature, notamment dans le cadre des humanités numériques et de la valorisation du patrimoine culturel.

2.1 Reconnaissance d'entités nommées (NER)

La reconnaissance d'entités nommées (NER) est l'une des techniques les plus utilisées pour extraire automatiquement des informations structurées à partir de textes non structurés. De nombreux travaux ont cherché à entraîner des modèles de NER sur des corpus spécialisés issus du domaine culturel, incluant des descriptions de monuments, de sites archéologiques ou de documents historiques.

Ces modèles permettent de détecter des entités telles que des lieux, des dates, des personnalités historiques, ou des noms de monuments, facilitant ainsi l'indexation et l'analyse de contenus patrimoniaux. Toutefois, ces approches nécessitent souvent des corpus annotés manuellement et une phase d'entraînement coûteuse, ce qui limite leur déploiement dans des contextes multilingues ou à faibles ressources.

2.2 Approches par mots-clés et règles

Une alternative classique, particulièrement répandue dans le domaine des **humanités numériques**, repose sur l'utilisation de **listes de mots-clés** élaborées manuellement. Ces approches basées sur des règles permettent d'identifier des concepts ou des entités précises sans nécessiter de phase d'apprentissage. Par exemple, il est courant de construire des lexiques d'éléments architecturaux (comme *arc, dôme, vitrail*), de matériaux (comme *bois, pierre, brique*) ou de styles (comme *roman, gothique, baroque*). Ces méthodes offrent une **grande transparence** dans les résultats et sont facilement **adaptables** à différents contextes linguistiques ou patrimoniaux. Néanmoins, elles peuvent manquer de **souplesse sémantique** et ne gèrent pas bien les variations de formulation ou les ambigüités lexicales.

2.3 Modèles Transformers

L'avènement des modèles de type transformers, tels que BERT (Bidirectional Encoder Representations from Transformers) et ses variantes multilingues (comme mBERT ou XLM-RoBERTa), a profondément transformé les méthodes d'extraction d'information. Ces modèles capturent les relations contextuelles entre les mots, ce qui leur permet de mieux comprendre la structure linguistique et les nuances sémantiques. Appliqués à des tâches comme la classification de texte, la reconnaissance d'entités ou la réponse à des questions, ils offrent des performances nettement supérieures aux approches traditionnelles, y compris dans des textes anciens ou complexes.
Dans notre projet, ces approches ne sont pas mises en œuvre : **elles sont mentionnées ici uniquement à titre de comparaison avec des techniques plus avancées.**

En effet, leur utilisation nécessite des ressources computationnelles importantes et soulève des enjeux d'interprétabilité, peu compatibles avec notre objectif de simplicité et de transparence.

2.4 Positionnement de notre approche

Dans ce projet, nous avons choisi de privilégier une approche **simple, explicable et facilement déployable**, combinant :

- Des **modèles NER pré-entraînés** (SpaCy) pour détecter des entités nommées générales (personnes, lieux, organisations),
- Des **listes de mots-clés structurées** pour extraire des informations spécifiques au domaine architectural (éléments, matériaux, styles),
- Une **prise en charge bilingue (français/anglais)** sans recours à des corpus annotés ou à des modèles lourds comme BERT.

Ce choix permet une **mise en œuvre rapide**, tout en garantissant une **bonne couverture sémantique** des informations recherchées dans les deux langues. Il offre également une **interprétabilité** directe des résultats, essentielle dans le cadre des humanités numériques et de la recherche patrimoniale.

CHAPITRE 2 :
Méthodologie et Résultats

3. Collecte et préparation des données

3.1 Collecte des Données

Dans le cadre de ce projet, aucune base de données publique préexistante adaptée au domaine architectural et patrimonial n'a été trouvée. En conséquence, la collecte des données a été réalisée **manuellement**, mot par mot, à partir de sources fiables en ligne telles que :

- Des articles Wikipédia (FR/EN) traitant de l'architecture et du patrimoine bâti.
- Des lexiques spécialisés et des glossaires d'architecture.
- Des guides patrimoniaux ou documents descriptifs d'organismes culturels.

Contrairement à une collecte de paragraphes, notre objectif était ici de construire des **listes de mots-clés représentatifs**, organisées selon trois grandes catégories :


```
"en": {
  "Architectural Features": [
    "arch", "arcade", "vault", "dome", "façade", "pendentive", "staircase", "railing", "landing", "window", "bay", "dormer", "oculus",
    "balcony", "terrace", "loggia", "veranda", "roof", "roofing", "framework", "mansard", "facade", "gable",
    "curtain wall", "ornament", "frieze", "cornice", "pediment", "partition", "cloister", "atrium", "apse",
    "small apse", "pillar", "beam", "lintel", "niche", "alcove", "tribune", "portal", "door", "leaf",
    "patio", "interior garden", "forecourt", "monumental staircase", "panel", "movable partition", "grille",
    "guardrail", "balustrade", "mezzanine", "modillioned cornice", "corbel", "gargoyle", "stained glass rose window", "ionic entablature",
    "doric entablature", "corinthian entablature", "basket-handle vault", "carved tympanum", "architrave",
    "bow-window balcony", "ornamental fence", "pergola", "porch", "bartizan", "entrance hall",
    "enclosure wall", "wooden lattice screen", "metal lattice screen", "folding screen", "corbel", "modillion", "spandrel",
    "sloping dormer", "carved frieze", "decorative panel", "carved lintel", "sound baffle", "flying buttress",
    "buttress", "ridge", "weathervane", "main beam", "stone lintel", "swing", "shutter",
    "rib network", "base", "crowned cornice", "lancet window", "bell tower", "campanile",
    "continuous balcony", "caryatid", "pilaster", "glass lens", "barbican", "triumphal arch", "columns", "column"
  ],
  "Materials": [
    "stone", "marble", "granite", "limestone", "sandstone", "brick", "terracotta", "wood", "glulam wood",
    "metal", "steel", "cast iron", "wrought iron", "aluminum", "copper", "zinc", "glass", "stained glass", "concrete",
    "reinforced concrete", "raw concrete", "plaster", "stucco", "ceramic", "faience", "thatch", "wattle and daub", "rammed earth", "adobe",
    "PVC", "composite", "plastic", "CLT panel", "OSB panels", "corrugated sheet metal", "slate", "tile",
    "fiber reinforced concrete", "prestressed concrete", "laminated glass", "tempered glass", "fiberglass", "stainless steel",
    "patinated copper", "brass", "titanium", "sandwich panels", "raw earth", "hemp", "cork", "bamboo",
    "technical fabric", "self-cleaning glass", "recycled concrete", "translucent concrete", "charred wood", "MDF panels",
    "HDF panels", "high-density OSB panels", "recycled plastic", "anodized aluminum", "soft limestone",
```


- **Éléments architecturaux** (exemple : *arcade*, *tour*, *colonnade*)
- **Matériaux** (exemple : *brique*, *pierre*, *bois*)
- **Styles architecturaux** (exemple : *gothique*, *roman*, *baroque*)

Ces mots-clés ont été **saisis et vérifiés manuellement** en français et en anglais, puis stockés dans un fichier structuré au format JSON, nommé "Keywords.json". Ce fichier sert de base pour l'extraction par règles dans l'application développée.

3.2 Caractéristiques :

- **Nombre total de mots-clés** : environ 300 (répartis entre les deux langues et les trois catégories)
- **Langues** : français et anglais
- **Format** : fichier JSON contenant des listes structurées par langue et par catégorie
- **Annotation** : non annoté manuellement ; utilisé directement pour l'extraction de mots-clés via correspondance simple (avec complément NER)

Ce choix de données simplifiées et construites manuellement permet de garantir un contrôle précis du contenu analysé, tout en rendant l'approche plus légère et adaptable à d'autres langues ou domaines.

4. Méthodologie :

4.1 Architecture Générale

L'architecture du système repose sur un pipeline modulaire de traitement du texte, développé en Python et intégré dans une application interactive via la bibliothèque Streamlit. Chaque texte soumis est traité selon une séquence d'étapes successives incluant le prétraitement linguistique, l'extraction d'entités nommées (NER), l'analyse par mots-clés, puis l'agrégation et la visualisation des résultats dans trois catégories : **éléments architecturaux**, **matériaux** et **styles**.

Ces étapes sont orchestrées par la fonction centrale **main()** qui gère l'interaction utilisateur via Streamlit. Elle prend en charge la saisie du texte, déclenche les différentes phases de traitement (détection de la langue, extraction par mots-clés, NER) et affiche dynamiquement les résultats.

4.2 Prétraitement

La première étape du pipeline consiste en un prétraitement du texte destiné à garantir une analyse linguistique cohérente. Le texte est d'abord converti en minuscules et nettoyé de sa ponctuation pour uniformiser les données. Ensuite, la fonction **detect_language(text)** est utilisée pour identifier automatiquement la langue du texte grâce à la bibliothèque **langdetect**. Cette étape est cruciale, car elle permet de déterminer quel modèle linguistique SpaCy doit être utilisé pour les traitements ultérieurs.

Une fois la langue détectée, la fonction **load_model(language)** charge dynamiquement le modèle NLP correspondant. Deux modèles sont utilisés : **fr_core_news_sm** pour les textes en français et **en_core_web_sm** pour ceux en anglais. Ces modèles permettent la tokenisation ainsi que la reconnaissance d'entités nommées. Le texte est ensuite tokenisé à l'aide de SpaCy, rendant possible la détection fine de mots-clés ou d'entités spécifiques.

4.3 Extraction par mots-clés

L'analyse thématique du texte repose sur des **listes de mots-clés structurées** manuellement, stockées dans un fichier JSON (keywords.json). Ces listes couvrent trois catégories centrales du domaine architectural : **éléments architecturaux**, **matériaux**, et **styles**. La fonction **extract_keywords(text, keywords)** est chargée d'examiner le texte tokenisé et de détecter la présence de termes appartenant à ces listes.

La détection s'effectue via des correspondances exactes ou partielles entre les tokens du texte et les mots-clés prédéfinis. Cette approche par règles, bien que simple, permet une interprétation directe des résultats et une adaptation aisée à d'autres contextes linguistiques ou culturels.

4.4 Reconnaissance d'entités nommées (NER)

En complément de l'analyse lexicale par mots-clés, l'application intègre une reconnaissance d'entités nommées (NER) via la fonction **extract_named_entities(text, nlp)**. Cette fonction exploite les modèles SpaCy chargés précédemment pour détecter des entités telles que des **noms de monuments historiques**, **lieux géographiques**, **personnalités**, ou **organisations liées au patrimoine**.

L'extraction NER permet d'enrichir l'analyse du texte avec des éléments qui ne figurent pas dans les listes de mots-clés, tout en apportant une couche de compréhension contextuelle plus fine. Les entités extraites sont classées selon leur type (e.g., PERSON, GPE, ORG) et restituées à l'utilisateur.

4.5 Agrégation et Visualisation

L'ensemble des résultats issus de l'extraction par mots-clés et par NER est ensuite agrégé. Cette opération est intégrée dans le traitement principal orchestré par **main()**, qui fusionne les données extraites et les trie selon les trois catégories finales : **Éléments architecturaux**, **Matériaux**, et **Styles**. Ces résultats sont ensuite affichés dans une interface utilisateur claire et interactive grâce aux capacités de Streamlit.

Le système a été conçu pour être **modulaire** et **extensible**. Chaque fonction remplit un rôle clairement défini, ce qui facilite la maintenance et l'évolution future du projet (par exemple, l'intégration de nouveaux styles, de langues supplémentaires, ou même de modèles plus avancés comme BERT si les contraintes techniques le permettent à l'avenir).

5. Implémentation

5.1 Outils et Librairies

L'application a été développée en **Python** à l'aide de plusieurs bibliothèques spécialisées :

- **SpaCy** pour le traitement automatique du langage (tokenisation, analyse morphosyntaxique, reconnaissance d'entités nommées).
- **langdetect** pour la détection automatique de la langue du texte.
- **Streamlit** pour créer une interface web interactive permettant aux utilisateurs de tester le système facilement.

5.2 Architecture de l'Application

L'architecture repose sur un pipeline modulaire intégré dans une interface Streamlit. Le traitement suit les étapes suivantes :

Détection de la langue

Lorsqu'un utilisateur saisit un paragraphe dans l'application, la fonction **detect_language(text)** utilise **langdetect** pour identifier automatiquement la langue (français ou anglais). Par exemple, si l'utilisateur entre le texte suivant :

"*La cathédrale Notre-Dame de Paris, construite en pierre au XIIe siècle, est un chef-d'œuvre de l'architecture gothique. Sa façade est ornée de colonnes et de voûtes impressionnantes*."

le système détectera correctement la langue comme étant **français** (fr), ce qui conditionnera le reste du traitement.

Chargement dynamique du modèle NLP

Une fois la langue détectée, la fonction **load_model(language)** sélectionne le modèle SpaCy adapté :

- **fr_core_news_sm** pour le français,
- **en_core_web_sm** pour l'anglais.

Ce modèle est ensuite utilisé pour analyser le texte, en identifiant les **entités nommées** comme les noms de lieux ou de monuments. Dans notre exemple, le système extrait correctement les entités suivantes :

- Notre-Dame de Paris (MONUMENT)
- XIIe (DATE)
- Paris (LIEU)

Extraction par mots-clés

Le système utilise ensuite la fonction **extract_keywords(text, keywords)** pour extraire des termes spécifiques correspondant à des **éléments architecturaux**, **matériaux**, ou **styles** à partir de listes définies manuellement et stockées dans un fichier "keywords.json".

Dans notre exemple, les mots-clés suivants sont extraits :

- **Éléments architecturaux** : voûtes, façade
- **Matériaux** : pierre
- **Styles** : gothique

Cette approche par mots-clés permet de repérer des termes techniques, même s'ils ne sont pas identifiés automatiquement comme des entités nommées.

Affichage structuré

La fonction centrale **main()** orchestre l'ensemble du processus : détection de langue, chargement du modèle, extraction NER, recherche par mots-clés, puis **affichage dynamique** des résultats dans l'interface Streamlit, triés par catégories. Pour l'exemple précédent, l'interface présentera :

- Langage détectée : Français
- Entités nommées : Notre-Dame de Paris, XIIe, Paris
- Éléments architecturaux : voûtes, façade
- Matériaux : pierre
- Styles : gothique

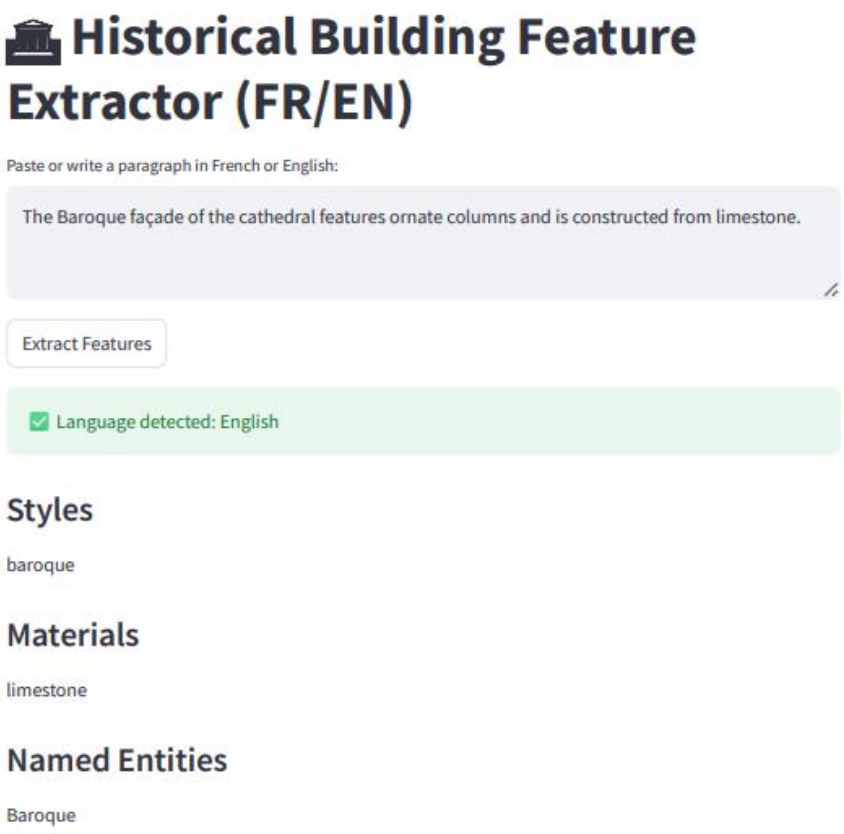
Ce cas d'usage illustre clairement l'intérêt d'un pipeline combinant techniques simples (recherche par mots-clés) et outils NLP avancés (SpaCy), le tout accessible via une interface web conviviale. Le système reste modulaire et extensible, permettant l'ajout futur de langues, catégories ou modèles plus performants (comme BERT ou CamemBERT).

6. Résultats

6.1 Évaluation Qualitative

L'évaluation du système a été réalisée de manière qualitative à partir de plusieurs textes représentatifs en français et en anglais. Les extraits suivants illustrent la capacité du système à extraire les informations ciblées.

Exemple en anglais :



"The Baroque façade of the cathedral features ornate columns and is constructed from limestone."

Résultat obtenu :

- **Matériaux** : *limestone*
- **Styles** : *baroque*
- **Entités nommées :** *Baroque*

Exemple en français :

"L'église gothique en pierre possède un clocher imposant et des vitraux colorés."

Résultat obtenu :

- **Éléments architecturaux** : *clocher*
- **Matériaux** : *pierre, vitrail*
- **Styles** : *gothique*

Ces exemples montrent que le système est capable d'identifier les principales entités pertinentes dans des textes courts, avec une bonne précision à la fois pour les mots-clés spécifiques et les entités nommées.

6.2 Observations

L'analyse des performances à travers plusieurs textes a permis de dégager les observations suivantes :

- **Bonne gestion du bilinguisme** : le système parvient à adapter automatiquement le traitement selon la langue détectée. Les résultats sont globalement cohérents en anglais comme en français, notamment grâce au chargement dynamique des modèles linguistiques adaptés.
- **Extraction pertinente des informations** : les éléments architecturaux, matériaux et styles sont bien identifiés dans la majorité des cas. Cela s'explique par la qualité des listes de mots-clés utilisées et la simplicité du domaine étudié (terminologie architecturale bien définie).
- **Limites du NER** : les modèles SpaCy, bien qu'efficaces, extraient parfois des entités hors sujet comme des dates ou des noms communs mal classifiés. Par exemple, certains adjectifs ou noms ambigus comme *"Roman"* sont parfois identifiés comme des noms propres ou mal catégorisés.
- **Ambiguïtés lexicales** : certains mots comme *"roman"* (peut désigner un style architectural ou une langue) ou *"basilica"* (peut désigner une typologie ou un bâtiment spécifique) posent des problèmes de désambiguïsation. Le système, basé sur des règles simples, ne peut pas toujours interpréter correctement ces cas selon le contexte.
- **Influence du style rédactionnel** : les textes descriptifs bien structurés (comme ceux de Wikipédia) donnent de meilleurs résultats que les textes plus libres ou subjectifs. Cela suggère que la robustesse du système dépend partiellement de la régularité lexicale et grammaticale des textes.
- **Aucune annotation manuelle nécessaire** : l'approche sans supervision (basée sur des règles et des dictionnaires) permet une extraction rapide sans phase de labellisation, ce qui constitue un avantage pour des projets à faible ressources.

En somme, cette évaluation montre que l'approche adoptée est fonctionnelle pour une première version, avec des résultats satisfaisants sur des textes simples. Toutefois, pour

une application à grande échelle ou sur des corpus plus variés, l'intégration de modèles plus avancés ou de techniques de désambiguïsation contextuelle serait à envisager.

7. Perspectives

7.1 Intégration de modèles contextuels avancés

L'approche actuelle repose sur des méthodes légères, transparentes et facilement déployables, telles que l'extraction par mots-clés et la reconnaissance d'entités nommées avec SpaCy. Une piste d'amélioration consisterait à intégrer des modèles de traitement du langage plus avancés, tels que **BERT multilingue** ou **XLM-RoBERTa**, capables de mieux interpréter le contexte linguistique.

Bien que ces modèles n'aient pas été utilisés dans le cadre de ce projet, leur utilisation future pourrait renforcer la précision de l'analyse, notamment dans des contextes plus complexes ou ambigus.

7.2 Constitution d'un corpus annoté

L'absence de données annotées a conduit à adopter une approche fondée sur des règles. Pour permettre une **évaluation quantitative rigoureuse**, il serait utile de constituer un **corpus bilingue annoté manuellement**, contenant des descriptions patrimoniales associées à des balises correspondant aux éléments architecturaux, matériaux et styles. Ce corpus servirait à mesurer les performances de l'outil et à entraîner ou affiner d'éventuels modèles supervisés.

7.3 Désambiguïsation sémantique

Certaines erreurs observées sont dues à l'ambiguïté lexicale de certains termes, par exemple *roman* pouvant désigner un style architectural ou une langue. L'introduction d'une **analyse syntaxique plus fine** ou de **techniques de désambiguïsation contextuelle** pourrait réduire ces confusions. Des approches basées sur des graphes de dépendances syntaxiques ou sur l'analyse des cooccurrences pourraient s'avérer utiles dans ce contexte.

7.4 Enrichissement de l'interface utilisateur

L'interface Streamlit actuelle offre une interaction simple et fonctionnelle. Des améliorations pourraient être apportées par l'ajout de **visualisations interactives**, telles que :

- Des nuages de mots (Word Clouds) pour représenter les éléments extraits.
- Des graphes de relations entre entités.
- Ou encore des cartes géographiques situant les monuments cités.

Ces enrichissements permettraient de **mieux valoriser les résultats** de l'analyse et de renforcer l'aspect pédagogique et exploratoire de l'outil.

8. Conclusion

Ce projet a démontré la faisabilité d'une approche simple mais efficace pour l'extraction d'informations architecturales à partir de textes historiques en français et en anglais. En combinant deux techniques complémentaires — la **recherche par mots-clés** et la **reconnaissance d'entités nommées (NER)** via SpaCy — il a été possible d'identifier trois catégories clés : **les éléments architecturaux**, **les matériaux** et **les styles**.

Malgré l'absence d'un corpus annoté ou de modèles sophistiqués d'apprentissage profond, les résultats obtenus sont pertinents, notamment grâce à l'utilisation de **listes de mots-clés bien structurées** et à une segmentation linguistique adaptée. Le recours à une interface Streamlit a permis de rendre le système accessible et interactif, facilitant l'expérimentation et la visualisation des résultats par l'utilisateur final.

Cette solution, bien que fondée sur des règles simples, constitue une base solide pour des applications futures dans le domaine de la **valorisation numérique du patrimoine architectural**. Elle offre un point de départ pour des extensions possibles, telles que l'intégration de modèles contextuels, l'utilisation de données annotées, ou encore l'optimisation de l'interface utilisateur à des fins pédagogiques, exploratoires ou scientifiques.

En somme, le projet illustre comment des outils légers de traitement du langage naturel peuvent être mis à profit pour explorer et structurer des connaissances patrimoniales multilingues, ouvrant la voie à des travaux plus complexes dans le champ des humanités numériques.

Références

1. **Matthew Honnibal, Ines Montani.** *spaCy 2: Natural Language Understanding with Bloom Embeddings, Convolutional Neural Networks and Incremental Parsing.* https://spacy.io
2. **Google.** *langdetect – Language detection library ported from Google's language-detection.* https://pypi.org/project/langdetect
3. **Streamlit Inc.** *Streamlit: The fastest way to build and share data apps.* https://streamlit.io
4. **Wikipédia.** Articles sur les monuments historiques, l'architecture gothique, romane, baroque, etc. https://fr.wikipedia.org / https://en.wikipedia.org
5. **Patrick Juola.** *Authorship Attribution*. Foundations and Trends in Information Retrieval, 2006.
6. **S. Bird, E. Klein, E. Loper.** *Natural Language Processing with Python*. O'Reilly Media, 2009.
7. **Projet de l'utilisateur.** Données collectées manuellement à partir de sources patrimoniales et historiques, rapports et guides d'architecture.