Alcatel·Lucent
Enterprise

Présentation Stage
Recherche Sémantique

Nathan Cerisara

03 Juin - 30 Août 2024

Alcatel·Lucent
Enterprise

Plan Global

I. Introduction du sujet
II. Explication du fonctionnement / Architecture du projet
III. Démos (WebApp puis Bot dans Sandbox Rainbow)
IV. Conclusion
V. Questions?
VI. Annexes (NER, découpe de conversation)

I. Introduction du sujet

- Recherche Syntaxique dans Rainbow
- Limitations: Fautes de frappes, pas les bons mots
  - Ex:
    - “La réunion de 10h” ↛ “La réu de 10h”
    - “J’ai du mal à travailler” ↛ “J’arrive pas à bosser”
- Solution
  - -> Recherche sémantique

II. Fonctionnement / Architecture - Plan

1) Formalisation d'une instance Rainbow
2) Structure de moteurs / Exemple d'un moteur de recherche
3) Optimisations / Stockage de cache de données
4) Intégration à Rainbow / Bot de recherche
5) Mentions supplémentaires
   a) Benchmarks
   b) Optimisation des hyper-paramètres des configurations

1) Formalisation d'une instance Rainbow - Éléments de base

Bubble

- id
- name
- members_ids
- messages_ids

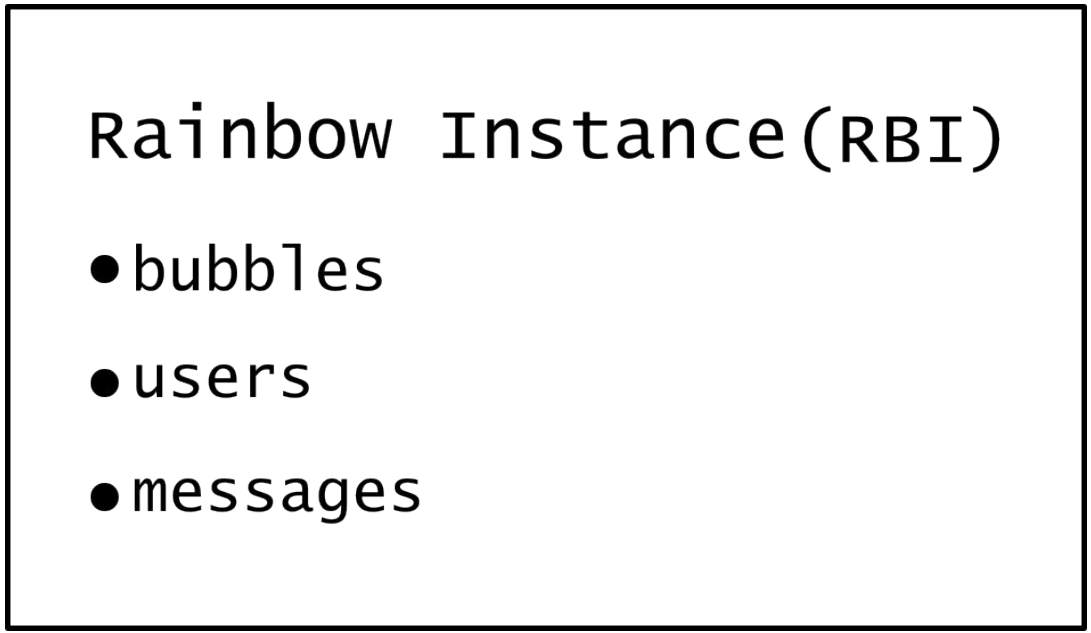
User

- id
- name
- bubbles_ids
- messages_ids

Message

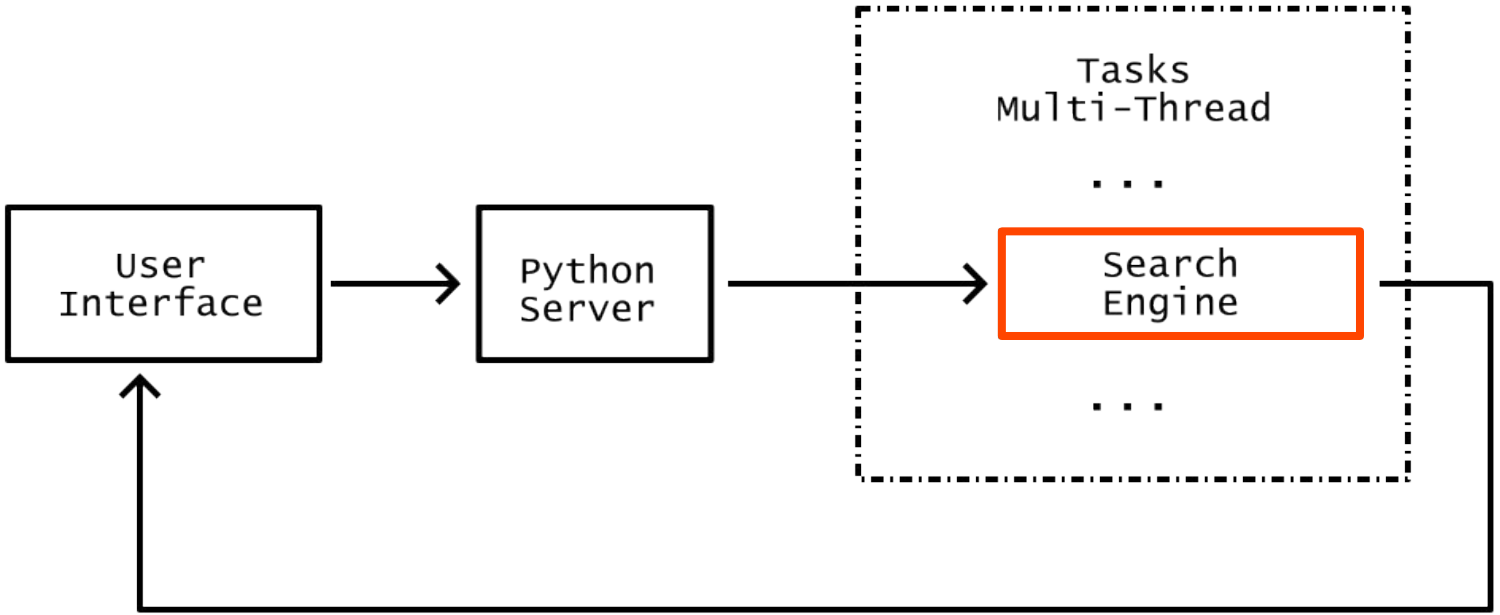
- id
- content
- author_id
- author_name
- date
- bubble_id
- answered_msg_id

1) Formalisation d’une instance Rainbow - Définition RBI



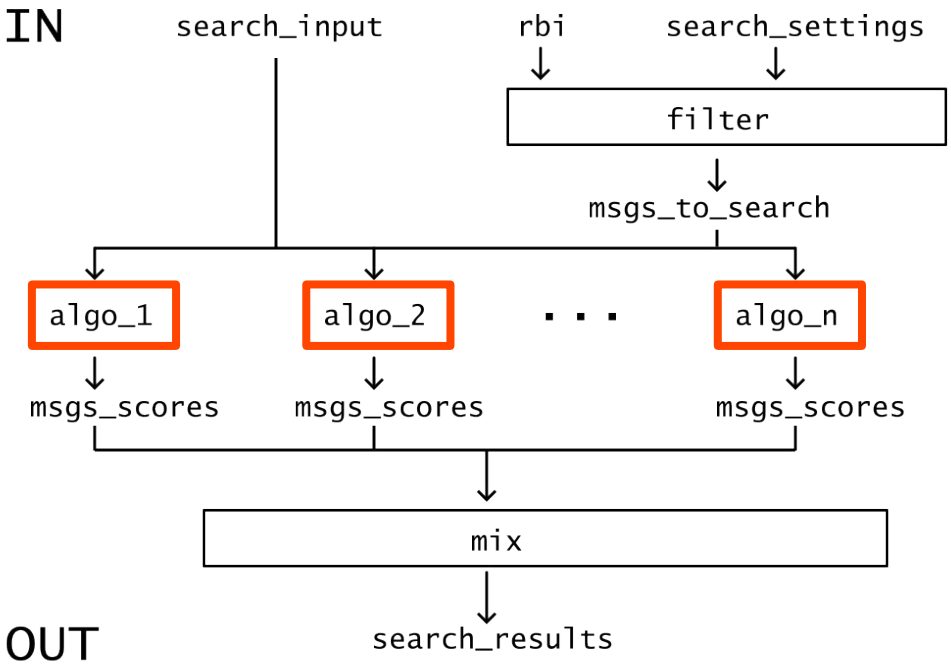
2) Structure de moteurs / Exemple d'un moteur de recherche

Architecture Globale Simplifiée



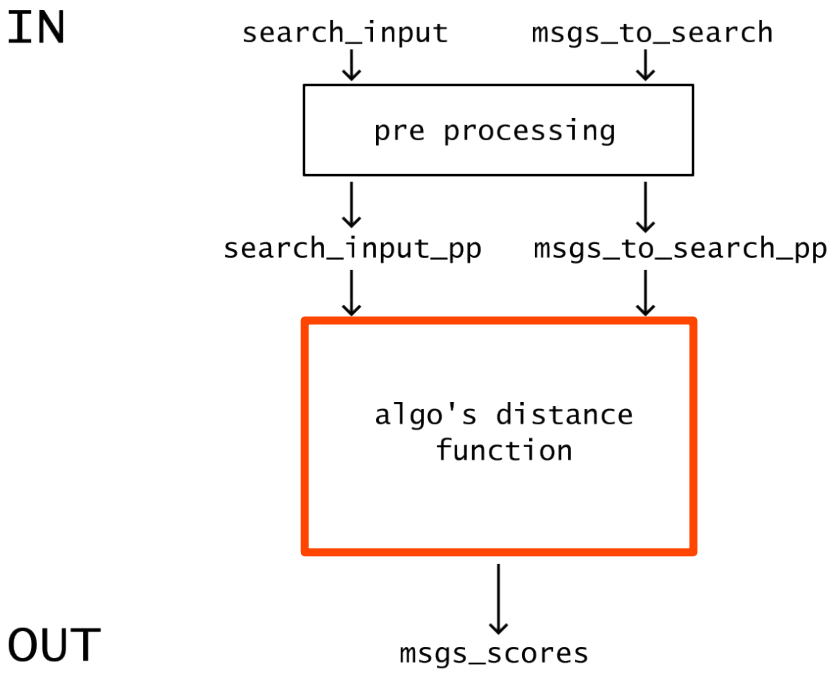
2) Structure de moteurs / Exemple d'un moteur de recherche

Structure d'un Moteur de Recherche



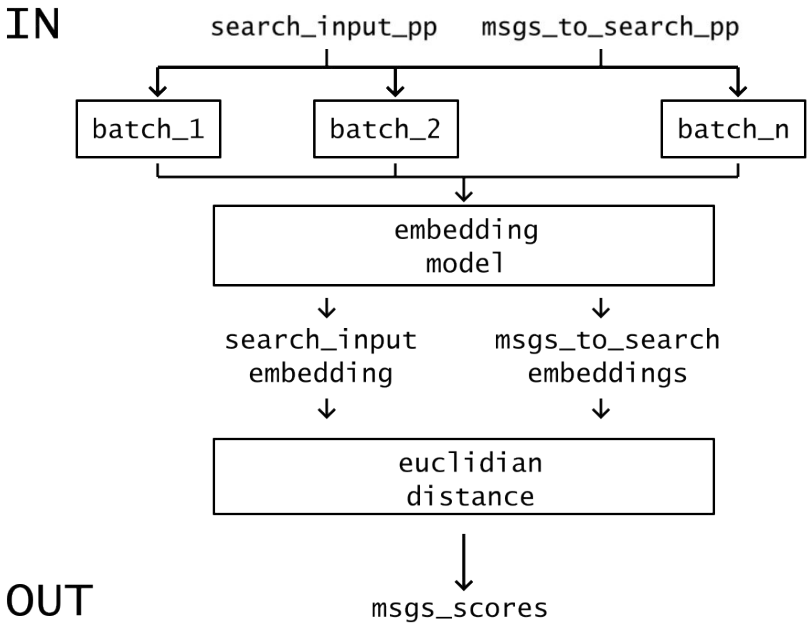
2) Structure de moteurs / Exemple d'un moteur de recherche

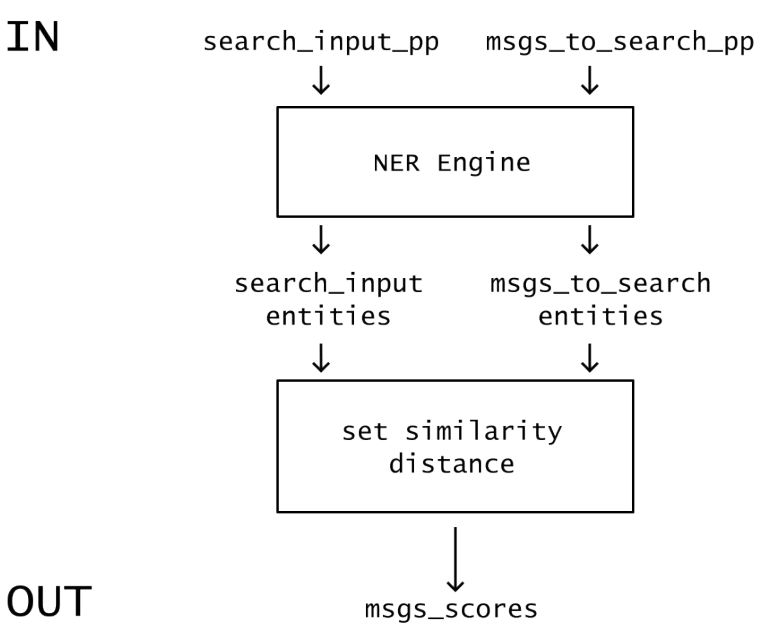
Structure d'un Algorithme de Recherche



2) Structure de moteurs / Exemple d'un moteur de recherche

Exemples: Algorithme d'embedding & Algorithme avec NER Engine





2) Structure de moteurs / Exemple d’un moteur de recherche
Détails sur le modèle d’embedding utilisé

Modèle d’embedding:

- Architecture Transformers
- AllMiniLM-l6-v2, Hugging Face, licence Apache 2.0
  - 22 millions de paramètres
  - Architecture de type Bert
  - Tourne en local sur ce Laptop

- Code Modulable -> Simple de changer de modèle si souhaité (config)

Plus de détails dans les questions pour ceux qui veulent.

3) Optimisations / Stockage de cache de données

- Compromis Vitesse - Stockage (Cache)

- Traduction (car modèles meilleurs en anglais)

- Stratégie actuelle: tout pré-calculer une seule fois à l'initialisation du serveur + mises à jours rapides à la réception de messages, tout côté serveur.

- Il faudra quand même bien étudier la mise à l'échelle dans l'architecture finale

4) Intégration à Rainbow / Bot de recherche

- Sandbox Rainbow
- Utilisation SDK C#
- Connexion socket entre script C# et serveur Python
- Limitations de la SDK
  - On n’a donc qu’un petit bot qui fonctionne par messages
    - Si intégré à Rainbow, ce sera directement dans l’interface de la recherche

5) Mentions supplémentaires
a) Benchmarks

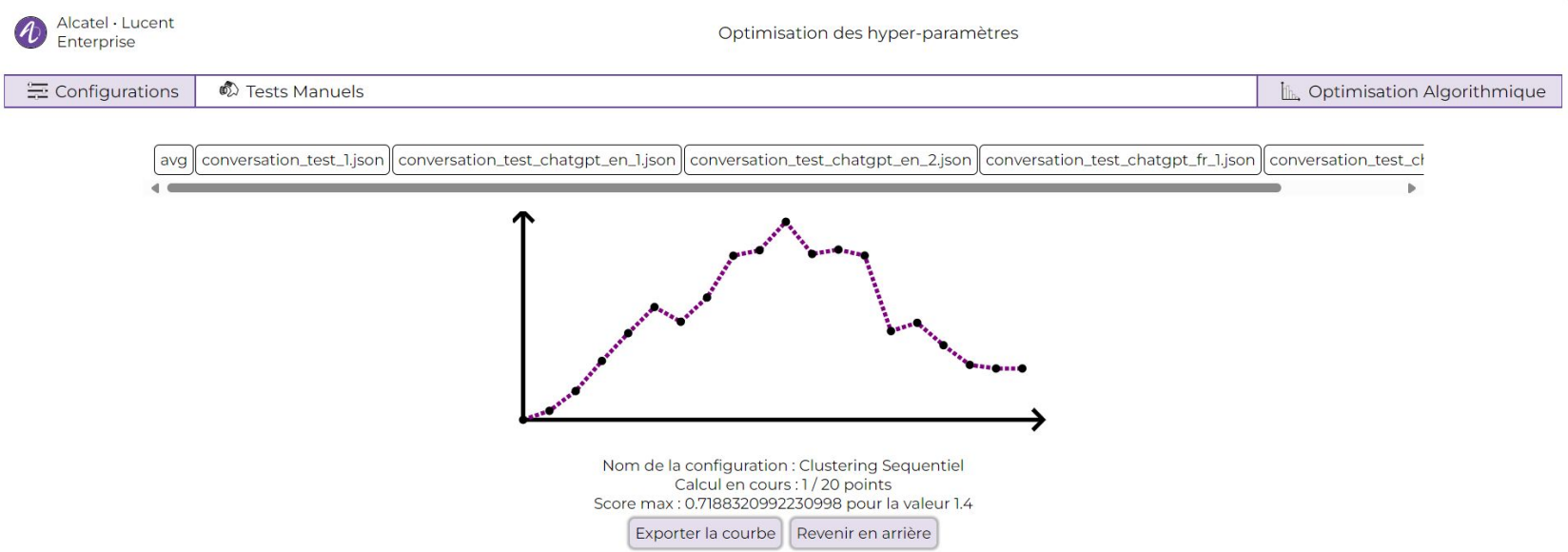
Alcatel · Lucent Enterprise

Benchmarks de Recherche - Tableau des benchmarks

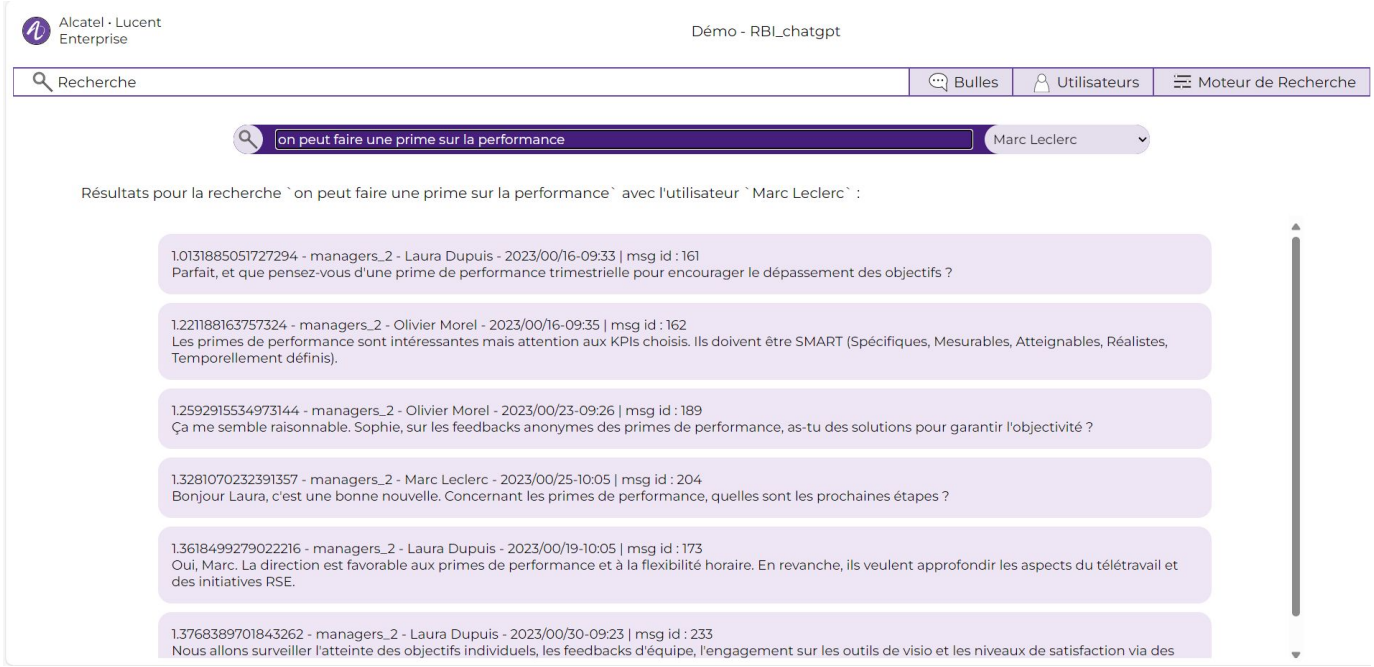
Machine Support : CPU: AMD Ryzen 5 PRO 4650U with Radeon Graphics - GPU:

	Moyenne	Evaluation Sémantique Basique en Anglais		Evaluation Sémantique basique en Français		Evaluation Sémantique simple en Anglais		Evaluation Sémantique simple en Français		Test Sémantique en Français qui se concentre sur la NER		Test d'usage qui va tester des recherches dans une réelle bulle extraite de Rainbow	
nom	score	score	vitesse	score	vitesse	score	vitesse	score	vitesse	score	vitesse	score	vitesse
Embeddings all-MiniLM-L6-v2 with NER replacement and translation	0.8995	1	1.9837 sec	1	1.5499 sec	0.9167	0.3058 sec	0.9167	0.3649 sec	0.87	3.7941 sec	0.6938	138.7033 sec
Embeddings all-MiniLM-L6-v2 with NER Engine and translation	0.8862	1	1.0069 sec	1	1.0615 sec	0.9167	0.1974 sec	0.9167	0.2059 sec	0.7146	2.4419 sec	0.7694	120.4834 sec
Embeddings all-MiniLM-L6-v2 with NER replacement	0.7662	1	0.8477 sec	0.7456	1.0317 sec	0.9167	0.2141 sec	0.5387	0.202 sec	0.7751	2.5621 sec	0.6211	89.1231 sec
Embeddings all-MiniLM-L6-v2	0.7204	1	0.9578 sec	0.7456	1.1697 sec	0.9167	0.309 sec	0.5387	0.3342 sec	0.5003	4.0981 sec	0.6211	116.0886 sec
Embeddings all-MiniLM-L6-v2 with NER Engine	0.7053	1	1.0966 sec	0.7456	0.8938 sec	0.9167	0.2013 sec	0.5387	0.2044 sec	0.4677	2.3278 sec	0.5632	103.5798 sec
Simple Syntaxic	0.3083	0.4852	0.003 sec	0.2767	0.002 sec	0.125	0 sec	0.0313	0 sec	0.5905	0.0081 sec	0.3408	0.5071 sec

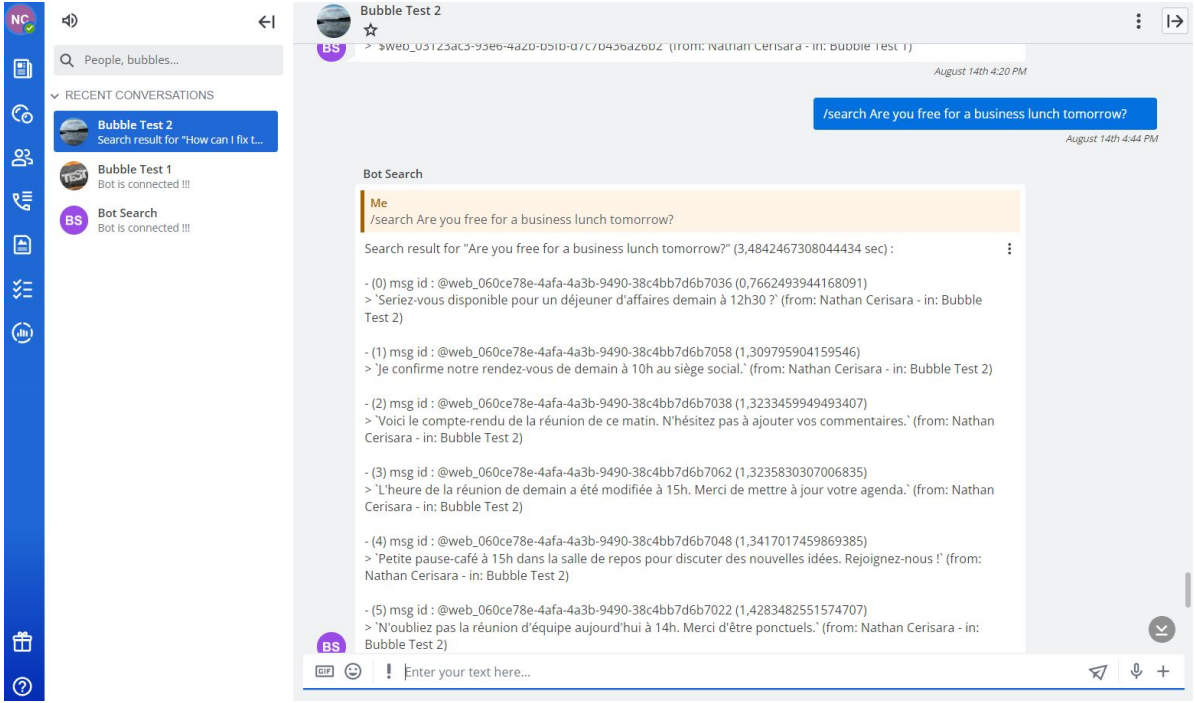
5) Mentions supplémentaires
b) Optimisation des hyper-paramètres des configurations



III. Démo - WebApp



III. Démo - Bot dans Sandbox Rainbow



Alcatel·Lucent Enterprise

IV. Conclusion

- Meilleure Recherche
- Architecture modulable -> flexibilité et améliorations
- Avec meilleurs modèle d'embeddings pour le Français
  - pas besoin de traduction, donc plus de problèmes de cache de traduction
- Il reste quand mêmes quelques limites sur "l'intelligence" du modèle.
- Tout tourne en local sur le laptop Thinkpad
  - + rapide avec meilleurs CPU / GPU
- Il faudra quand même bien étudier la mise à l'échelle, pour voir quelle puissance serveur sera requise, ou bien déléguer certaines tâches côté client?

Questions ?

VI. Annexes

Annexe 1 - Reconnaissance d'Entités Nommées (NER)

Dictionnaires d'entités nommées

Annexe 2- Découpe de conversation

Annexe 1 - Reconnaissance d'Entités Nommées (NER)

Dictionnaires d'entités nommées


```
{
    "EDT": "emplois du temps",
    "JIRA": "système de suivi de bugs, de gestion des incidents et de gestion de projets",
    "LinkedIn Learning": "online learning platform that provides video courses taught by industry experts",
    "Coursera": "entreprise numérique proposant des formations en ligne ouvertes à tous",
    "KPI": "indicateur clé de performance",
    "KPIs": "indicateurs clés de performance",
    "RSE": "responsabilité sociétale des entreprises",
    "GAN": "réseau adversaire génératif",
    "GANs": "réseaux adversaires génératifs",
    "SurveyMonkey": "site de sondage en ligne gratuit avec sondages personnalisables",
    "ViT": "Vision Transformer",
    "DETR": "Detection Transformer, Detectron"
}
```

Annexe 2 - Découpe de conversation

Algorithme séquentiel:

- Calcul d'une matrice de distances entre tous les messages
- On lit les messages dans l'ordre
- Pour chaque message:
  - Si conversation trouvée qui convient
    -> On l'y ajoute
  - Sinon, nouvelle conversation