# Annalyse de la structure / architecture / mode de fonctionnement d’un chatbot

Analyse sur les chatbot et potentiel developpement d'un chatbot

Avant-propos : les principaux chatbots disponibles sur le marché, et leur domaine d’application

1. **ManyChat** (marketing et vente) *– chatbot simple*
2. **IBM Watson Assistant** (service client, assistance technique) *– chatbot IA*
3. **Google Dialogflow** (service client, automatisation des flux de travail) *– chatbot IA*
4. **Amazon Lex** (service client, automatisation des tâches) *– chatbot IA*
5. **Microsoft Bot Framework** (service client, automatisation des processus métiers) *– chatbot hybride*
6. **LivePerson** (commerce en ligne, service client) *– chatbot hybride*
7. **Zendesk Answer Bot** (gestion de tickets, support client) *– chatbot simple*
8. **Intercom** (service client, marketing) *– chatbot hybride*
9. **Drift** (marketing, vente) *– chatbot hybride*
10. **Kore.ai** (santé, finance) *– chatbot IA*
11. **HubSpot Chatbot Builder** (marketing, service client) *– chatbot hybride*
12. **Tars** (marketing digital) *– chatbot simple*
13. **Wysa** (santé mentale) *– chatbot IA*
14. **Replika** (compagnonnage, bien-être personnel) *– chatbot IA*
15. **Botnation** (marketing et service client) **[Français]** *– chatbot Hybride*
16. **SNCF OuiBot** (transport) **[Français]** *– chatbot simple*
17. **Doctolib Assistant** (santé) **[Français]** *– chatbot simple*
18. **Wysii** (assistance psychologique) **[Français]** *– chatbot IA*
19. **ChatGPT** (généraliste, création de contenu, assistance conversationnelle) *– chatbot IA*
20. **Gemini AI** (généraliste, IA conversationnelle avancée) *– chatbot IA*
21. **Claude AI** (généraliste, sécurité et éthique dans les interactions conversationnelles) *– chatbot IA*

Resumé de la présentation de Christophe Biernacki lors du Workshop FrugALIAS du 4 octobre 2024 : **MISE EN PERSPECTIVE HISTORIQUE DE LA FRUGALITÉ EN IA ET STATISTIQUE**  
  
La présentation traite de l'importance cruciale de la frugalité dans les domaines de l'intelligence artificielle et des statistiques, soulignant comment les innovations technologiques peuvent et doivent être réorientées vers une utilisation plus sobre des ressources. Le concept de frugalité en IA, qui implique une minimisation de la consommation d'énergie et des données, est mis en avant comme une réponse essentielle aux défis environnementaux, sociétaux et économiques actuels.

Christophe Biernacki retrace l'histoire des statistiques, depuis les premiers systèmes de stockage de l'information sur os et bois jusqu'à l'utilisation moderne d'ordinateurs et de logiciels avancés, pour illustrer comment la collecte et le traitement des données ont toujours été liés à la disponibilité et à l'efficacité des ressources. Face à des modèles de plus en plus gourmands en calcul, il propose des alternatives frugales qui maintiennent l'efficacité tout en réduisant les coûts, comme l'utilisation optimisée des données, le transfer learning, et la compression des modèles. Cette approche non seulement préserve les ressources mais favorise également une IA plus durable et accessible, adaptée aux limitations matérielles et énergétiques du monde réel.

L'effet rebond, abordé vers la fin de la présentation, souligne un paradoxe important dans l'adoption de technologies frugales en intelligence artificielle et statistique. Alors que les solutions frugales visent à réduire la consommation de ressources et les coûts, l'effet rebond se manifeste lorsque ces gains d'efficacité entraînent une augmentation de la consommation globale plutôt qu'une réduction. Cet effet peut survenir si les améliorations de l'efficacité rendent les technologies moins coûteuses et donc plus accessibles ou attrayantes, conduisant à une utilisation accrue qui annule les bénéfices environnementaux ou économiques initiaux.

Par exemple, en rendant les processus de calcul moins gourmands en énergie, les entreprises et les chercheurs peuvent être tentés d'exécuter plus fréquemment leurs algorithmes ou d'augmenter la complexité des tâches qu'ils n'auraient pas envisagées auparavant, augmentant ainsi la charge totale sur les ressources. Christophe Biernacki met en garde contre cet effet rebond, soulignant la nécessité d'une approche plus large qui ne se contente pas de réduire les coûts unitaires mais qui vise également à modérer la demande globale pour éviter une augmentation nette de la consommation.

Architecture d’un chatbot :

<https://www-ometrics-com.translate.goog/blog/a-simple-guide-to-chatbot-architecture/?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=rq>

**1. Initialisation**

**Composants :**

* **Base d'intentions** : Liste des intentions que le chatbot peut reconnaître.
* **Modèle de Machine Learning (ML)** : Modèle entraîné pour prédire les intentions à partir des messages des utilisateurs.
* **Seuils de confiance** : Valeurs définissant le niveau de confiance requis pour accepter une réponse basée sur des règles ou sur ML.
* **Historique de conversation** : Stocke les interactions passées pour le contexte.
* **Base de connaissances** : Ensemble d'informations et de données utilisées pour générer des réponses pertinentes.

**Fonctions :**

* charger\_base\_intentions()
* charger\_modele\_ml()
* charger\_base\_connaissances()

Lors de l'initialisation, le chatbot charge toutes les ressources nécessaires, y compris les intentions pré-définies, le modèle ML pour la compréhension du langage, et les bases de connaissances pour générer des réponses contextuelles. Les seuils de confiance déterminent quand utiliser les règles ou le ML pour répondre.

**2. Traitement des Messages Utilisateur**

**Composants :**

* **Prétraitement** : Nettoyage et normalisation du texte.
* **Approche basée sur les règles** : Recherche de correspondances avec des intentions définies.
* **Approche ML** : Utilisation du modèle ML (machine learning) pour prédire l'intention et générer une réponse.
* **Décision hybride** : Choix de la réponse finale basée sur les seuils de confiance.
* **Mise à jour de l'historique** : Enregistrement des interactions pour le contexte futur.

**Fonctions :**

* traiter\_message(message\_utilisateur)
* preprocesser\_texte(texte)
* appliquer\_regles(message)
* prediction\_ml(message)
* demander\_clarification()
* mettre\_a\_jour\_historique(message, reponse\_finale)

Lorsqu'un message utilisateur est reçu, il est d'abord prétraité pour le nettoyer et le normaliser. Ensuite, le chatbot tente de répondre en utilisant d'abord les règles basées sur des intentions pré-définies. Si aucune règle ne correspond avec un niveau de confiance suffisant, le modèle ML est utilisé pour prédire l'intention et générer une réponse. La décision finale est prise en comparant les niveaux de confiance des deux approches. Si aucune des approches ne répond avec un niveau de confiance suffisant, le chatbot demande une clarification à l'utilisateur.

**3. Prétraitement du Texte**

**Composants :**

* **Conversion en minuscules** : Uniformise le texte.
* **Tokenisation** : Divise le texte en unités plus petites (tokens).
* **Suppression des stop words** : Élimine les mots non significatifs.
* **Lemmatisation** : Réduit les mots à leur forme de base.

**Fonctions :**

* convertir\_minuscules(texte)
* tokeniser(texte)
* supprimer\_stop\_words(texte\_tokenise)
* lemmatiser(texte\_sans\_stop\_words)

Le prétraitement prépare le texte pour une analyse plus efficace en éliminant les bruits et en normalisant les données. Cela améliore la précision de la détection d'intentions et de l'extraction d'entités.

**4. Approche Basée sur les Règles**

**Composants :**

* **Patterns de correspondance** : Expressions régulières ou motifs définis pour reconnaître les intentions.
* **Génération de réponses** : Sélectionne et remplit des templates de réponses basées sur l'intention détectée.

**Fonctions :**

* appliquer\_regles(message)
* correspond\_pattern(message, patterns)
* generer\_reponse(intention)

Cette approche utilise des règles définies pour reconnaître des intentions spécifiques dans le message utilisateur. Si une correspondance est trouvée avec un niveau de confiance suffisant, une réponse prédéfinie est générée.

**5. Approche Machine Learning (ML)**

**Composants :**

* **Vectorisation du texte** : Transformation du texte en vecteurs numériques compréhensibles par le modèle ML.
* **Prédiction de l'intention** : Utilisation du modèle ML pour prédire l'intention basée sur le message vectorisé.
* **Estimation de la confiance** : Calcul du niveau de confiance de la prédiction.
* **Génération de réponses via ML** : Création de réponses basées sur l'intention prédite.

**Fonctions :**

* prediction\_ml(message)
* vectoriser\_texte(message)
* modele\_ml.predire(vecteur\_message)
* modele\_ml.estimer\_confiance(prediction)
* generer\_reponse\_ml(prediction)

Le modèle ML analyse le message prétraité pour prédire l'intention de l'utilisateur. En fonction de la confiance de la prédiction, le chatbot décide d'utiliser cette réponse ou de recourir à une clarification.

**6. Décision Hybride**

**Composants :**

* **Seuils de confiance distincts** : Déterminent quand utiliser les réponses basées sur les règles ou sur ML.
* **Gestion des clarifications** : Demande des précisions à l'utilisateur lorsque les niveaux de confiance sont insuffisants.

Le chatbot compare les niveaux de confiance des réponses basées sur les règles et sur le ML. Il priorise les réponses des règles si la confiance est élevée. Sinon, il utilise le ML, et si nécessaire, demande une clarification à l'utilisateur.

**7. Génération de Réponses Contextuelles**

**Composants :**

* **Extraction du contexte** : Utilise l'historique des conversations pour comprendre le contexte actuel.
* **Sélection et remplissage de templates** : Adapte les réponses en fonction du contexte.

**Fonctions :**

* generer\_reponse(intention)
* extraire\_contexte(historique\_conversation)
* selectionner\_template(intention, contexte)
* remplir\_template(reponse\_template, contexte)

Les réponses générées tiennent compte du contexte conversationnel en utilisant l'historique des interactions. Cela permet au chatbot de fournir des réponses plus pertinentes et personnalisées.

**8. Clarification**

**Composants :**

* **Messages de clarification prédéfinis** : Listes de phrases pour demander des précisions à l'utilisateur.

**Fonctions :**

* demander\_clarification()
* choisir\_aleatoirement(clarifications)

Lorsque le chatbot n'est pas sûr de la demande de l'utilisateur, il utilise des messages de clarification pour obtenir plus d'informations, améliorant ainsi l'expérience utilisateur et la précision des réponses futures.

**9. Apprentissage Continu**

**Composants :**

* **Collecte des nouvelles conversations** : Rassemble les interactions récentes pour l'entraînement.
* **Entraînement incrémental du modèle ML** : Met à jour le modèle ML avec de nouvelles données.
* **Mise à jour de la base de connaissances** : Intègre les nouvelles informations apprises.

**Fonctions :**

* apprentissage\_continu()
* collecter\_historique\_journee()
* modele\_ml.entrainer(nouvelles\_conversations)
* mettre\_a\_jour\_base\_connaissances(nouvelles\_conversations)

Le chatbot améliore continuellement ses performances en réentraînant le modèle ML avec de nouvelles données collectées quotidiennement. Cela permet d'adapter le chatbot aux nouvelles tendances et aux changements dans les demandes des utilisateurs.

**10. Programme Principal**

**Composants :**

* **Boucle de conversation** : Gère les interactions en continu avec l'utilisateur.
* **Gestion des commandes de sortie** : Permet à l'utilisateur de quitter la conversation.

**Fonctions :**

* attendre\_message\_utilisateur()
* afficher(reponse)

Le programme principal initialise le chatbot et entre dans une boucle continue où il attend les messages des utilisateurs, traite chaque message, et affiche les réponses générées. Il inclut également une condition pour quitter la boucle lorsque l'utilisateur le demande.

**Diagramme de Flux Simplifié**

[Utilisateur]

|

v

[Message Utilisateur]

|

v

[Prétraitement]

|

v

[Détection d'Intention]

/ \

/ \

[Approche [Approche

Basée sur ML]

Règles] |

| |

v v

[Réponse Règles] [Réponse ML]

\ /

\ /

[Décision Hybride]

|

v

[Réponse Finale]

|

v

[Utilisateur]

**Détails des Composants**

**1. Prétraitement**

Le prétraitement est essentiel pour nettoyer le texte brut et le transformer en une forme que le système peut analyser efficacement. Cela inclut :

* **Conversion en minuscules** : Assure que la casse n'affecte pas la reconnaissance des intentions.
* **Tokenisation** : Divise le texte en mots ou tokens individuels.
* **Suppression des stop words** : Élimine les mots courants qui n'apportent pas de valeur sémantique (comme "le", "et", "ou").
* **Lemmatisation** : Réduit les mots à leur forme de base, facilitant la reconnaissance des intentions et des entités.

**2. Détection d'Intention**

**Approche Basée sur les Règles**

Cette méthode utilise des motifs prédéfinis pour reconnaître des intentions spécifiques. Elle est rapide et efficace pour des scénarios bien définis mais manque de flexibilité face à des variations non anticipées.

* **Correspondance de patterns** : Compare le message utilisateur avec des motifs définis pour chaque intention.
* **Génération de réponse** : Si une correspondance est trouvée, une réponse appropriée est générée en utilisant des templates.

**Approche Machine Learning**

Cette méthode utilise des algorithmes de ML pour prédire l'intention basée sur des exemples d'entraînement. Elle est plus flexible et capable de gérer des variations complexes dans les messages utilisateurs.

* **Vectorisation** : Convertit le texte en vecteurs numériques.
* **Prédiction** : Le modèle ML prédit l'intention la plus probable.
* **Estimation de la confiance** : Calcule la probabilité associée à la prédiction pour évaluer sa fiabilité.

**3. Décision Hybride**

Le cœur du chatbot hybride réside dans la décision de quelle approche utiliser pour générer la réponse finale. Les seuils de confiance déterminent si la réponse basée sur les règles ou sur le ML est suffisamment fiable. Si aucune approche ne dépasse le seuil, une clarification est demandée à l'utilisateur.

**4. Génération de Réponses Contextuelles**

En utilisant l'historique des conversations, le chatbot peut adapter ses réponses pour qu'elles soient plus pertinentes dans le contexte actuel. Cela inclut la sélection de templates de réponse qui prennent en compte les interactions passées.

**5. Apprentissage Continu**

Pour maintenir et améliorer la performance du chatbot, il est crucial d'intégrer un mécanisme d'apprentissage continu. En collectant régulièrement de nouvelles données de conversation et en réentraînant le modèle ML, le chatbot peut s'adapter aux nouvelles demandes et tendances.

**Avantages de l'Architecture Hybride**

1. **Flexibilité** : Combine les forces des approches basées sur les règles (précision pour des cas spécifiques) et du ML (adaptabilité et compréhension contextuelle).
2. **Robustesse** : Permet de gérer une large gamme d'interactions utilisateur, même celles non anticipées lors de la conception initiale.
3. **Amélioration Continue** : L'apprentissage continu permet au chatbot de s'améliorer avec le temps, augmentant ainsi sa pertinence et sa précision.
4. **Gestion du Contexte** : L'utilisation de l'historique des conversations permet des interactions plus naturelles et cohérentes.

### Liste résumée des principaux tests statistiques : **1. Tests paramétriques :**

* **t-test** :
  + t-test à un échantillon
  + t-test à deux échantillons indépendants
  + t-test apparié
* **ANOVA** :
  + ANOVA à un facteur
  + ANOVA à deux facteurs
* **Corrélation** :
  + Test de corrélation de Pearson
* **Régression linéaire** :
  + Régression linéaire simple
  + Régression linéaire multiple

**2. Tests non paramétriques :**

* **Test de Wilcoxon** :
  + Test des rangs signés
  + Test de Wilcoxon-Mann-Whitney
* **Kruskal-Wallis** (équivalent non paramétrique de l'ANOVA)
* **Corrélation** :
  + Test de corrélation de Spearman
* **Test du Chi-carré** :
  + Chi-carré d’indépendance
  + Chi-carré de conformité

**3. Tests de proportion :**

* Test d'une proportion
* Test de proportions pour deux échantillons

**4. Tests d’hypothèse spécifiques :**

* Shapiro-Wilk (test de normalité)
* Levene (homogénéité des variances)
* Kolmogorov-Smirnov (test de distribution)

**5. Tests multivariés :**

* PERMANOVA (ANOVA multivariée sur matrices de distance)
* MANOVA (ANOVA multivariée)

Documentation pour l’utilisation de l’api chatgpt :   
<https://platform.openai.com/docs/api-reference/introduction>

Frugalité -> dirigé vers un chatbot de stat