

# Retrieval Augmented Generation

Introduction à l'IA générative

*Theo Lopes Quintas*

BPCE Payment Services,  
Université Paris Dauphine

2025-2026

# Introduction

La bibliothèque de Babel, Jorge Luis Borges

Jorge Luis Borges est un écrivain argentin du 20e siècle. Il publie en 1941 la nouvelle **La Bibliothèque de Babel** qui y décrit une bibliothèque composé de tous les livres de 410 pages. Certains sont du charabias, d'autres contiennent du savoir. Dans sa recherche de savoir face à *l'infini* le narrateur conclut son récit par :

*Le désordre apparent, se répétant, constituerait un ordre, l'Ordre. Ma solitude se console à cet élégant espoir.*



# Introduction

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b><i>Retrieval</i></b>	<b>4</b>
2.1	Initialisation	4
2.2	Utilisation	6
2.3	Mesure de performance du <i>retrieval</i>	7
2.4	Modélisation probabiliste	8
<b>3</b>	<b>Mesures de performances d'un modèle de langage</b>	<b>10</b>
3.1	Évaluations automatique	11
3.2	Évaluations humaine	15
3.3	Évaluations par modèle de langage	17

# Introduction

Dans les grandes lignes

La méthode RAG adresse les limitations des modèles de langage : informations récentes non présente dans les données d'entraînement, informations spécifiques à un contexte peu ou pas disponible sur internet, réduit les hallucinations et gagne en tracabilité.

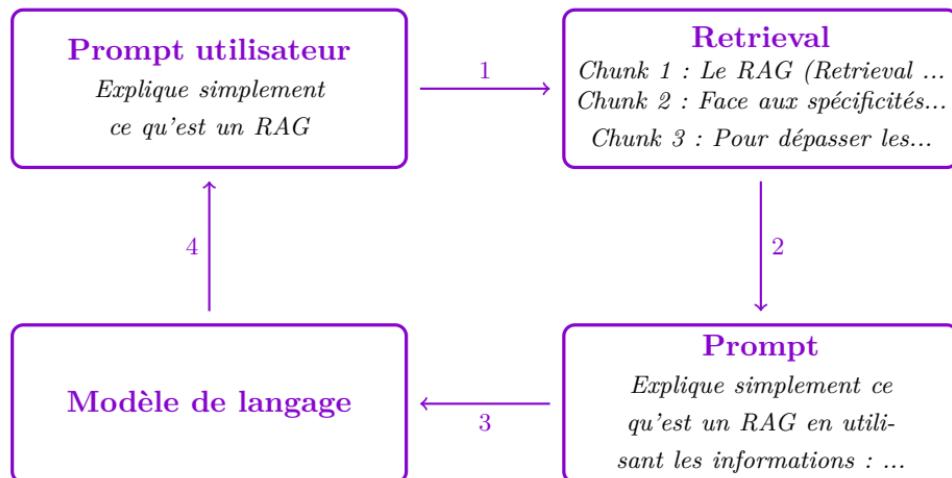
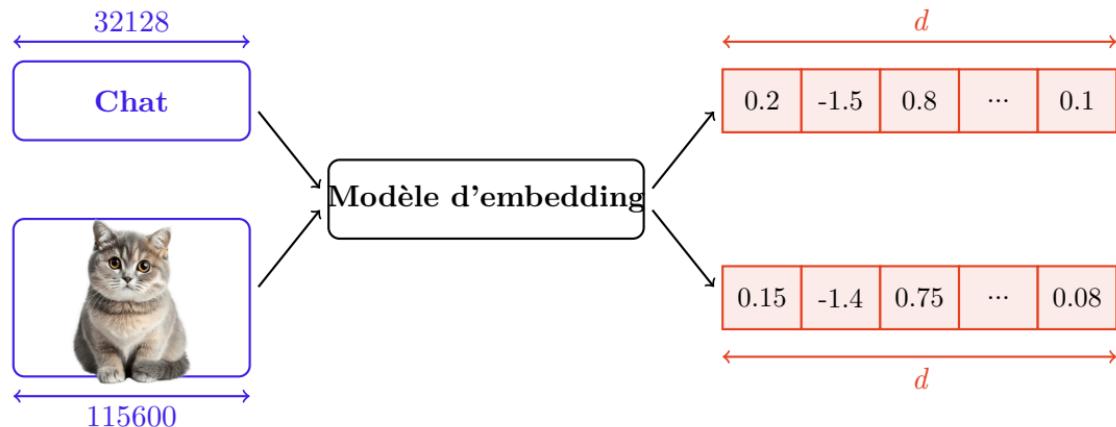


Figure – Schéma de la méthode *Retrieval Augmented Generation*

# Introduction

Rappel : Embedding

Un **embedding** est une représentation vectorielle d'objets complexe (texte, images, sons, ...) dans un espace de faible dimension par rapport aux dimensions d'origine. Dans l'espace de faible dimension les relations de similarité se traduisent par une proximité géométrique. La réduction de dimension ainsi réalisé permet d'accélérer la recherche de voisins.



**Figure** – Illustration d'un embedding multimodal

## Retrieval

Initialisation : Création de chunk

La méthode RAG nécessite :

- ▶ Une **base de connaissance** de qualité, éventuellement multimodale (texte, image, son...)
- ▶ Un modèle pour réaliser un embedding de la base de connaissance
- ▶ Une méthode pour indexer la base de donnée obtenu (l'embedding)

La qualité d'un modèle de langage est liée à celle de sa base d'entraînement, il en est de même pour la méthode RAG. La base de donnée est ensuite découpé en sous-parties nommées **chunks** pour pouvoir être vectorisé via un modèle d'embedding.

Plusieurs stratégies sont possibles pour créer ces chunks et doivent être adaptées aux documents constituant et au contexte d'utilisation. Des chunks long véhiculent plus de contexte mais *lisse* des spécificités qu'un chunk plus court peut conserver.

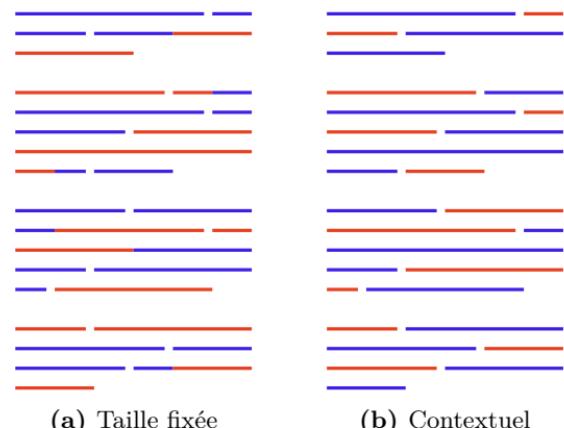


Figure – Exemple de deux méthodes de *chunking*

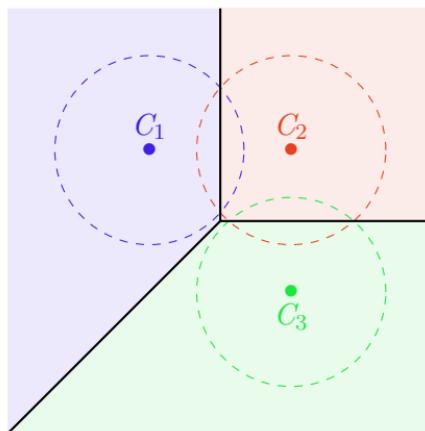
# Retrieval

Initialisation : Indexation

La méthode RAG nécessite :

- ▶ Une base de connaissance de qualité, éventuellement multimodale (texte, image, son...)
- ▶ Un modèle pour réaliser un embedding de la base de connaissance
- ▶ Une méthode pour **indexer** la base de donnée obtenu (l'embedding)

La méthode **FAISS** (Facebook AI Similarity Search) permet d'accélérer la recherche de voisin dans une base de donnée grâce à plusieurs optimisations. L'une d'elle consiste à segmenter les données avec un clustering K-Means pour ne chercher les voisins que dans la cellule ou les cellules dont le centroid est le plus proche du vecteur d'entrée.



**Figure** – Exemple d'un clustering K-Means avec  $K = 3$  clusters

## Retrieval

### Similarité

La **similarité cosinus** est une mesure de similarité et non une distance. Elle identifie les vecteurs similaires comme ceux ayant un angle proche.

$$\text{Cosine}(\textcolor{blue}{u}, \textcolor{red}{v}) = \frac{\langle \textcolor{blue}{u}, \textcolor{red}{v} \rangle}{\|\textcolor{blue}{u}\|_2 \|\textcolor{red}{v}\|_2}$$

(Similarité cosinus)



**Figure** – Similarité cosinus

À partir de l'embedding de la question et de l'ensemble de connaissance, on peut identifier les *chunks* les plus proches.

## Retrieval

Mesure de performance du retrieval

Nous avons besoin de savoir si le *retrieval* trouve des éléments **pertinents** et si il les trouvent **tous**. Pour le faire, nous devons nous placer dans un contexte supervisé où on connaît par avance les questions et les chunks que le retrieval doit retrouver.

$$\text{Context Precision}@K = \frac{\sum_{k=1}^K \text{Precision}@k \times v_k}{\sum_{k=1}^K v_k}$$

$v_k = \begin{cases} 1 & \text{Chunk } k \text{ is useful} \\ 0 & \text{otherwise} \end{cases}$

Avec  $\text{Precision}@k$  définit comme la précision dans le cadre d'une classification, à partir des  $k$  premiers chunks récupérés.

$k = 1$   $v_k = 1$  Precision@ $k = 1$

$k = 2$   $v_k = 1$  Precision@ $k = 1$

$k = 3$   $v_k = 0$  Precision@ $k = \frac{2}{3}$

$k = 4$   $v_k = 1$  Precision@ $k = \frac{3}{4}$

$k = 5$   $v_k = 0$  Precision@ $k = \frac{3}{5}$

**Figure – Exemple**

Dans l'exemple, on obtient  $\text{Context Precision}@5 = \frac{1+1+\frac{3}{4}}{3} = 92\%$ . Pour s'assurer qu'on les obtient **tous**, le **recall** est la bonne métrique, *mutatis mutandis*.

# Retrieval

Modélisation probabiliste

On peut voir le RAG de manière probabiliste pour une requête  $x$  et une bonne réponse générée  $y$  à partir de  $(d_i)_{i \leq k}$  documents :

$$\mathbb{P}(y|x) \approx \sum_{i=1}^k \mathbb{P}(y|x, d_i) \mathbb{P}(d_i|x)$$

On peut alors identifier quatre possibilités d'échecs dans un RAG :

## Récupération

$$\mathbb{P}(d_i|x) \approx 0$$

- ▶ Réponse non spécifique

## Priorité

L'information n'est pas le premier chunk renvoyé

- ▶ Réponse partielle ou biaisée

## Chunking

L'information est séparées dans plusieurs documents

- ▶ Raisonnement incomplet

## Indépendance

$$\mathbb{P}(y|x, d) \approx \mathbb{P}(y|x)$$

- ▶ Réponse non spécifique

# En résumé

1	Introduction . . . . .	1
2	<i>Retrieval</i> . . . . .	4
2.1	Initialisation . . . . .	4
2.2	Utilisation . . . . .	6
2.3	Mesure de performance du <i>retrieval</i> . . . . .	7
2.4	Modélisation probabiliste . . . . .	8
3	Mesures de performances d'un modèle de langage . . . . .	10

# Mesures de performances d'un modèle de langage

Méthodologies de mesures de performances

## Évaluation automatique

Mesurer la performance de génération par rapport à une ou plusieurs métriques. Le plus souvent via un ensemble de prompts et réponses attendues en utilisant l'accuracy par exemple.

## Évaluation humaine

Annotations humaines sur des paires (prompt, réponse) d'un modèle. Peut être un avis d'influenceur, compte-rendu ou une compétition en mode arène sur des sites dédiés.

## Évaluation par modèle de langage

Exploiter un modèle de langage identifié comme plus performant pour noter, ou comparer, un ou plusieurs modèles de langage.

# Mesures de performances d'un modèle de langage

Évaluation automatique : GSM8K

**GMS8K** [Cobbe et al., 2021] est un dataset de problèmes mathématiques d'algèbre élémentaire en anglais.

## Exemple 1

**Question** : Weng earns \$12 an hour for babysitting. Yesterday, she just did 50 minutes of babysitting. How much did she earn ?

**Réponse** : Weng earns  $12/60 = \$\ll 12/60=0.2 \gg 0.2$  per minute. Working 50 minutes, she earned  $0.2 \times 50 = \$\ll 0.2*50=10 \gg 10$ . ##### 10

## Exemple 2

**Question** : James spends 40 years teaching. His partner has been teaching for 10 years less. How long is their combined experience ?

**Réponse** : His partner has been teaching for  $40-10=\ll 40-10=30 \gg 30$  years So together they have  $40+30=\ll 40+30=70 \gg 70$  years of experience ##### 70

# Mesures de performances d'un modèle de langage

Évaluation automatique : MMLU

**MMLU** [Hendrycks et al., 2020a, Hendrycks et al., 2020b] est un dataset de questions à choix multiples sur 57 domaines de connaissances en anglais.

## Exemple 1

**Question** : Let  $x_1 = 1$  and  $x_{(n+1)} = \sqrt{3+2x_n}$  for all positive integers  $n$ . If it is assumed that  $x_n$  converges, then  $\lim x_n$

=

**Choix** :

- ▶ 3
- ▶ e
- ▶  $\sqrt{5}$
- ▶ 0

**Réponse** : 0 A

## Exemple 2

**Question** : Statement 1| L2 regularization of linear models tends to make models more sparse than L1 regularization. Statement 2| Residual connections can be found in ResNets and Transformers.

**Choix** :

- ▶ "True, True"
- ▶ "False, False"
- ▶ "True, False"
- ▶ "False, True"

**Réponse** : 3 D

# Mesures de performances d'un modèle de langage

Évaluation automatique : HellaSwag

**HellaSwag** [Zellers et al., 2019] est un dataset de complétion de phrase en anglais.

## Exemple 1

**Question** : He also trims the back and sides of his head with the clippers. He uses scissors to trim the hair and give it a finished look.  
the model

**Choix** :

- ▶ poses with how to complete the look using the partial and then suggests a computer graphic tool.
- ▶ poses and begins to talk while holding up a piece of metal.
- ▶ uses some hair gel to style his hair after the haircut.
- ▶ uses rollers to take some of the hair off the brush and enjoy the ending.

**Réponse** : 2

## Exemple 2

**Question** : A man is completing a rubiks cube. a timer

**Choix** :

- ▶ begins showing the amount of time it will take it to complete the disk.
- ▶ is counting down on the side of the cube.
- ▶ is sitting on the table next to him.
- ▶ goes hand at the end of the board while the man continues to solve and figure out the cubes.

**Réponse** : 2

# Mesures de performances d'un modèle de langage

## Évaluations automatique

Une évaluation automatique permet d'avoir :

- ▶ Une **reproductibilité** forte, mis à part l'aléatoire machine et du modèle
- ▶ Une **compréhension** des résultats avec des métriques explicite
- ▶ Un **coût** faible sur des données de **qualité**

Pour réduire la **contamination** on peut :

- ▶ Ajouter un **caractère spécial** pour identifier le texte associé à un benchmark
- ▶ **Ne pas rendre publique** le dataset, mais cela amène des enjeux de transparence
- ▶ **Modifier** régulièrement le benchmark, en ajoutant la difficulté de s'assurer que benchmark reste *au même niveau*.

Mais présente deux grandes limites :

- ▶ Les benchmarks sont du texte donc peuvent être présent dans les datasets d'entraînement des modèles : on parle de **contamination**
- ▶ Il est difficile de trouver une **métrique** juste pour des tâches complexe (résumés, génération de poèmes...)

## Mesures de performances d'un modèle de langage

Évaluations humaine : Score ELO

Dans une arène un évaluateur humain doit choisir une réponse préférée parmi deux proposées par deux modèles à une même question, sans connaître les modèles. Les modèles ont tous un rang nommé **score ELO**.

$$\mathbb{P}(A \text{ gagne face à } B) = \frac{1}{1 + 10^{-\frac{(R_A - R_B)}{400}}}$$

Rang de *B*

A l'issue de chaque *confrontation* les rangs du modèles *A* et *B* sont mis à jour :

$$\begin{cases} R_A &= K (\mathbb{1}_A \text{ gagne} - \mathbb{P}(A \text{ gagne face à } B)) \\ R_B &= K (1 - \mathbb{1}_A \text{ gagne} - \mathbb{P}(B \text{ gagne face à } A)) \end{cases}$$

↑  
Constante = 4

# Mesures de performances d'un modèle de langage

## Évaluations humaine - Limites

Une évaluation humaine permet d'avoir :

- ▶ Une évaluation sur des tâches **complexes**
- ▶ Une note qui reflète la **préférence** humaine
- ▶ Un **coût** faible pour avoir des résultats

Mais présente deux grandes limites :

- ▶ Les scores ELO sont sensible à **matchmaking** donc également à **l'ordre** des matchs
- ▶ Les résultats sont **biaisés** par la **population** qui répond : on parle de *culture bound*

# Mesures de performances d'un modèle de langage

## Évaluations par modèle de langage

Un modèle peut en évaluer un autre ! Cette méthodologie est principalement utilisée pour trois tâches :

1. Mesurer la performance d'une **génération** de texte : toxicité, utilité, cohérence...
2. Comparer deux réponses de modèles pour sélectionner la meilleure
3. Calculer la similarité entre une réponse et une référence

## Inconvénients

### Avantages

- Le **coût** est réduit par rapport à des annotateurs humains, et il peut être plus largement **mis à l'échelle**
- Les résultats sont **reproductible** et plus **objectifs** que les choix humains, en moyenne

- Des **biais** cachés peuvent compromettre le jugement, et ils sont souvent plus difficiles à détecter que ceux des annotateurs humains.
- Il est difficile de garantir la **qualité** : les quantités de texte produites lors de la mise à l'échelle doivent être vérifiées manuellement, et l'annotation humaine bien que plus onéreuse peut donner de meilleurs résultats.

# Mesures de performances d'un modèle de langage

## Évaluations par modèle de langage

Les modèles de langage en tant qu'évaluateur présentent quelques biais connus. Ils ont tendances à préférer :

- ▶ **Leur propre** réponses ([Panickssery et al., 2024]), on peut utiliser un ensemble de juge pour lisser ce biais
- ▶ Les **premières** réponses ([Zheng et al., 2023]), inverser l'ordre et ne conserver que les résultats cohérents est une possibilité
- ▶ Les réponses **longues** ([Dubois et al., 2024]), on peut pondérer les résultats par la longueur des réponses

# Mesures de performances d'un modèle de langage

Évaluations par modèle de langage - Évaluer l'évaluateur

Un modèle peut en évaluer un autre ! Cette méthodologie est principalement utilisée pour trois tâches :

1. Mesurer la performance d'une **génération** de texte : toxicité, utilité, cohérence...
2. Comparer deux réponses de modèles pour sélectionner la meilleure
3. Calculer la similarité entre une réponse et une référence

Puisque nous évaluons l'évaluateur, nous connaissons par avance la bonne réponse donc nous pouvons réutiliser les métriques de classification. Cependant, si on cherche à mesurer la corrélation, c'est une interprétation plus délicate.

## $\kappa$ de Cohen

Mesure l'accord entre deux annotateurs sur des données catégorielles. Il est plus conservateur par rapport aux autres métriques.

## $\tau$ de Kendall

Mesure la force et la direction de l'association entre deux classements. Il est plus robuste face aux outliers que le  $\rho$  de Spearman.

## $\rho$ de Spearman

Mesure la force et la direction de l'association entre deux classements. Il est plus sensible au magnitude d'écart entre les rangs que le  $\tau$  de Kendall.

## En résumé

1	Introduction . . . . .	1
2	<i>Retrieval</i> . . . . .	4
<b>3</b>	<b>Mesures de performances d'un modèle de langage . . . . .</b>	<b>10</b>
3.1	Évaluations automatique . . . . .	11
3.2	Évaluations humaine . . . . .	15
3.3	Évaluations par modèle de langage . . . . .	17

# Bibliographie I

- 📄 Adler, B., Agarwal, N., Aithal, A., Anh, D. H., Bhattacharya, P., Brundyn, A., Casper, J., Catanzaro, B., Clay, S., Cohen, J., et al. (2024).  
**Nemotron-4 340b technical report.**  
*arXiv preprint arXiv :2406.11704.*
- 📄 Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. (2021).  
**Training verifiers to solve math word problems.**  
*arXiv preprint arXiv :2110.14168.*
- 📄 Dubois, Y., Galambosi, B., Liang, P., and Hashimoto, T. B. (2024).  
**Length-controlled alpacaeval : A simple way to debias automatic evaluators.**  
*arXiv preprint arXiv :2404.04475.*
- 📄 Hendrycks, D., Burns, C., Basart, S., Critch, A., Li, J., Song, D., and Steinhardt, J. (2020a).  
**Aligning ai with shared human values.**  
*arXiv preprint arXiv :2008.02275.*
- 📄 Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. (2020b).  
**Measuring massive multitask language understanding.**  
*arXiv preprint arXiv :2009.03300.*

## Bibliographie II

-  Kusupati, A., Bhatt, G., Rege, A., Wallingford, M., Sinha, A., Ramanujan, V., Howard-Snyder, W., Chen, K., Kakade, S., Jain, P., et al. (2022).  
**Matryoshka representation learning.**  
*Advances in Neural Information Processing Systems*, 35 :30233–30249.
-  Panickssery, A., Bowman, S., and Feng, S. (2024).  
**Llm evaluators recognize and favor their own generations.**  
*Advances in Neural Information Processing Systems*.
-  Wang, Z., Dong, Y., Delalleau, O., Zeng, J., Shen, G., Egert, D., Zhang, J., Sreedhar, M. N., and Kuchaiev, O. (2024).  
**Helpsteer 2 : Open-source dataset for training top-performing reward models.**  
*Advances in Neural Information Processing Systems*.
-  Wang, Z., Dong, Y., Zeng, J., Adams, V., Sreedhar, M. N., Egert, D., Delalleau, O., Scowcroft, J. P., Kant, N., Swope, A., et al. (2023).  
**Helpsteer : Multi-attribute helpfulness dataset for steerlm.**  
*arXiv preprint arXiv* :2311.09528.  
On recommande de regarder la figure 1 de l'article.
-  Weller, O., Boratko, M., Iftekhar, N., and Lee, J. (2025).  
**On the theoretical limitations of embedding-based retrieval.**  
*arXiv preprint arXiv* :2508.21038.

## Bibliographie III

-  Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. (2019).  
**Hellaswag : Can a machine really finish your sentence?**  
*arXiv preprint arXiv :1905.07830.*
-  Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E., et al. (2023).  
**Judging llm-as-a-judge with mt-bench and chatbot arena.**  
*Advances in neural information processing systems.*

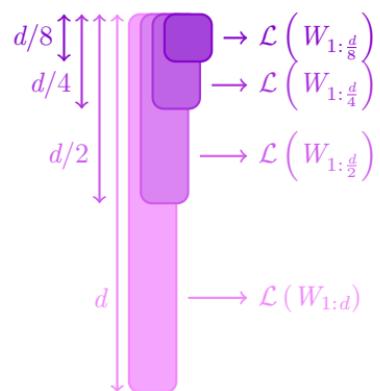
## Annexe

### Matryoshka Representation Learning

Différents usages peuvent nécessiter des tailles d'embedding différent, qui doivent être appris indépendamment. La méthode **Matryoshka Representation Learning**<sup>1</sup> [Kusupati et al., 2022] propose d'apprendre en une fois plusieurs embedding de manière *imbriqué*.

La méthode force l'embedding à hiérarchiser les informations à l'aide d'une fonction de perte  $\mathcal{L}$  composée de plusieurs fonction de perte, chacune traitant d'un segment du vecteur d'embedding. Elles sont finalement sommées et pondérées. Dans l'article les pondérations sont toutes fixées à 1.

OpenAI publie en janvier 2025 des embeddings exploitant cette méthodologie, compétitif avec des embeddings issues de modèle de langage performant comme Qwen3 par exemple.



**Figure** – Schéma du principe

1. *Matryoshka* désigne les poupées russes.

## Annexe

### Reward as a Judge

[Wang et al., 2023] puis [Wang et al., 2024] propose des datasets de paire (prompt, réponse) annoté selon cinq critères :

1. **Correctness** : inclusion de tous les faits pertinents sans erreurs
2. **Coherence** : consistance et expression claire
3. **Complexity** : profondeur de l'analyse
4. **Verbosity** : somme des détails inclus dans la réponse
5. **Helpfullness** : utilité de la réponse, notée indépendamment des autres critères bien que l'analyse suggère une corrélation forte avec *correctness* et *coherence*

[Adler et al., 2024] introduit la méthode **Reward as a Judge** pour la partie d'alignement du modèle avec la préférence humaine. En modifiant la dernière couche du modèle de langage, remplacer la couche softmax par une couche linéaire de 5 neurones, chaque paire (prompt, réponse) est scoré selon les cinq critères.

L'intérêt est d'avoir des scores **rapidement** et **reproductible**, sans repasser sur un prompt. On obtient également un jugement plus nuancé qu'une réponse catégorielle. Cependant cette méthode nécessite un **fine-tuning délicat**, qui peut être long et coûteux.

## Annexe

### Limite théorique du retrieval

L'ensemble du cours repose sur l'hypothèse que l'embedding est représenté par un unique vecteur. [Weller et al., 2025] montre que ce type de système ont une limite théorique de représentation, même pour des requêtes simple. Par exemple, considérons la requête suivante :

*Récupère des documents sur A et B, mais pas C*

Puisque le système de retrieval a pour objectif de trouver les vecteurs géométriquement les plus proches, il n'est pas capable de **raisonnement** : il est donc très probable que le système renvoie des vecteurs concernant C. La conclusion de [Weller et al., 2025] est qu'un embedding ne peut couvrir toute les combinaisons de retrieval qu'à partir d'une dimension d'embedding donnée souvent très grande, limitant donc l'utilité d'un embedding.

Avec l'utilisation progressive des agents, des requêtes complexes sont de plus en plus fréquente : il faudrait soit changer d'architecture<sup>2</sup> soit faire plusieurs retrieval et sélectionner après cette étape.

---

2. Mais nous n'avons pas encore de résultat de capacité de représentation.