Prédiction ou perdition de liens : les graines de l'instabilité dans les plongements de graphes de connaissances

Guillaume Méroué[1], Fabien Gandon[1], Pierre Monnin[1]

[1] Université Côte d'Azur, Inria, CNRS, I3S, France

prénom.nom@inria.fr

Résumé

Les modèles de plongements de graphe de connaissance (MPGC) constituent une des principales approches de prédiction de liens pour la complétion de graphes de connaissances. L'évaluation standard se concentre sur les métriques basées sur le rang de l'entité attendue, comme le MRR ou le Hits@K, mais néglige souvent l'influence des graines aléatoires sur la stabilité des résultats, masquant des instabilités potentielles dans les prédictions individuelles et l'organisation des espaces de plongement. Dans ce travail, nous menons une analyse systématique de la stabilité de plusieurs MPGC sur différents jeux de données. Nous constatons que les modèles performants produisent en réalité des prédictions divergentes au niveau des triplets et des espaces de plongement variables. En isolant les facteurs stochastiques (initialisation, ordre des triplets, échantillonnage négatif, dropout, hardware), nous montrons que chacun d'eux est suffisant pour induire une instabilité d'ampleur comparable. Par ailleurs, pour un modèle donné, les configurations d'hyperparamètres ayant un meilleur MRR ne garantissent pas plus de stabilité. Enfin, le vote, bien qu'étant un mécanisme de remédiation connu, n'apporte qu'une amélioration limitée à la stabilité. Ces conclusions soulignent les limites critiques des protocoles d'évaluation actuels et soulèvent des inquiétudes quant à la fiabilité des MPGC pour la complétion des graphes de connaissances.

Mots-clés

Graines Aléatoires, Apprentissage Automatique, Complétion de Graphes de Connaissances.

1 Introduction

Les Graphes de Connaissances (GC) [11] servent de structures fondamentales dans un large éventail d'applications académiques et industrielles, notamment la réponse aux questions [38], l'analyse sémantique [1], ou les systèmes de recommandation [34], entre autres [11]. Ils fournissent un réseau formel et explicite de faits dans de nombreux domaines d'application (par exemple, la biochimie, la géographie, la pharmacologie, la linguistique, etc.) [5, 11, 22]. Cela est permis par leur représentation structurée de l'information, généralement définie par des standards tels que RDF [16], où la connaissance est représentée par des triplets (h, r, t), indiquant qu'une relation r relie une entité sujet h à une entité objet t. Néanmoins, malgré leur large adoption, les GC du monde réel sont souvent incomplets en raison de leur processus de construction manuel ou semi-automatique [23]. Par exemple, en 2015, 58 % des scientifiques dans DBpedia n'avaient pas de lien vers leur domaine d'expertise [15]. Cette incomplétude a motivé diverses tâches d'enrichissement [11, 23]. Parmi elles, la *prédiction de liens* (PL) vise à déduire les entités manquantes dans des triplets incomplets de la forme $(h, r, ?)$ ou $(?, r, t)$. Au fil des années, divers modèles de plongement de graphes de connaissances (MPGC) ont été proposés pour la PL, s'imposant aujourd'hui comme les méthodes prépondérantes [2, 14].

Étant des modèles d'apprentissage automatique, les MPGC reposent intrinsèquement sur des éléments stochastiques lors de l'entraînement, qui peuvent être contrôlés par des graines aléatoires. Par conséquent, les performances d'un modèle peuvent varier d'une graine à l'autre. Une bonne pratique courante en apprentissage automatique consiste à reporter les performances en effectuant la moyenne des résultats sur plusieurs exécutions (*runs*) avec des configurations identiques mais des graines différentes, et de s'assurer que l'écart type reste faible, ce qui indiquerait des performances stables. Cependant, même en suivant cette pratique, deux exécutions entraînées avec des graines différentes peuvent obtenir des scores globaux similaires tout en produisant des prédictions top-K significativement différentes. Par exemple, dans la Figure 1, toutes les exécutions présentent le même MRR bien qu'elles affichent une variabilité substantielle, requête par requête, dans leurs prédictions.

Un tel comportement soulève un problème critique pour la tâche opérationnelle de complétion de GC : avec différentes graines, l'ajout des K meilleures prédictions au GC peut produire des graphes complétés substantiellement différents. Même dans les scénarios où les prédictions ne sont considérées que comme des candidats plutôt que d'être directement ajoutées au graphe (par ex., la découverte de traitements médicamenteux pour certaines maladies), la vérification de ces candidats peut engendrer des coûts importants. Dans de tels scénarios, il est crucial que les entités vaillent la peine d'être testées, ce qui nécessite une stabilité dans les prédictions. À titre d'illustration, dans la Figure 1, compléter le graphe avec les 2 meilleures prédic-

tions pour la requête (*Ribavirine, traitementPour, ?*) en utilisant trois instances de modèles aux performances identiques peut produire *Hépatite* et *Bronchite*, *Asthme* et *Bronchite*, ou *Anémie* et *Botulisme*, uniquement en fonction de la graine. La stabilité des MPGC est donc un enjeu critique pour la complétion opérationnelle des GC, où l'instabilité dépendante de la graine peut conduire à des complétions peu fiables et incohérentes, sapant ainsi la confiance accordée aux GC complétés en tant que structures de base pour les systèmes de prise de décision en aval.

Des travaux antérieurs [40] ont déjà montré que les métriques globales basées sur le rang, couramment utilisées pour la prédiction de liens, telles que le MRR ou le Hits@K, échouent à capturer la variabilité induite par les graines dans les prédictions des MPGC au niveau de chaque triplet. Ces auteurs ont étudié l'accord entre modèles sur la prédiction de l'entité de référence (*ground truth*) dans le top-K, ce qui ne s'aligne pas complètement avec le cadre opérationnel de la complétion de GC où la vérité terrain est inconnue. De plus, une fois qu'un MPGC a été entraîné, son espace de plongement appris est souvent réutilisé pour des tâches en aval au-delà de la prédiction de liens, telles que l'évaluation de la similarité entre entités [30], le regroupement conceptuel (*clustering*) [9], ou l'extraction de règles logiques [36]. Ces tâches en aval supposent la stabilité de l'espace de plongement pour des raisons de fiabilité, ce qui, à notre connaissance, n'a jamais été examiné au regard des facteurs d'aléatoire. Il est donc nécessaire de disposer de mesures de stabilité plus exhaustives ciblant les candidats du top-K et l'espace de plongement.

Cela a motivé ce travail [20], où nous interrogeons et quantifions la mesure dans laquelle l'aléatoire affecte à la fois le comportement prédictif et les représentations internes des MPGC. En particulier, nos questions de recherche sont les suivantes :

QR1. Les MPGC sont-ils stables lorsque l'ont fait varier les graines aléatoires ? Nous avons mené de multiples exécutions des mêmes modèles tout en faisant varier les sources d'aléatoire. Nous avons introduit des métriques (par ex., *Pred-Overlap*, *Space-Overlap*) pour évaluer la similarité des prédictions au niveau de chaque triplet et la similarité de l'espace de plongement appris. Notre analyse expérimentale révèle que, malgré une stabilité apparente des métriques agrégées telles que le MRR, les modèles présentent une variabilité prononcée tant au niveau des triplets que dans leurs espaces de plongement.

QR2. Comment les différentes sources d'aléatoire contribuent-elles à l'instabilité ? Nous avons pris en compte la contribution de chaque source habituelle d'aléatoire lors de l'apprentissage d'un MPGC, c.-à-d. l'initialisation, l'ordonnancement des triplets, l'échantillonnage négatif, le *dropout*, et avons également identifié le matériel (c.-à-d. le GPU) comme une source supplémentaire d'aléatoire. Nous avons montré que la variation de ces sources, de manière isolée ou conjointe, induit une instabilité d'ampleur comparable.

QR3. La stabilité d'un modèle est-elle corrélée à ses performances en prédiction de liens ? Nous avons mesuré la stabilité de 3 configurations identifiées lors d'une recherche d'hyperparamètres : la plus performante, médiane et la moins performante. Si les modèles les moins performants sont clairement très instables, un MRR plus élevé ne garantit pas pour autant une plus grande stabilité qu'un MRR médian, ni dans l'espace de plongement, ni dans les triplets prédits.

Après avoir introduit les travaux connexes dans la Section 2, nous décrivons le formalisme que nous proposons pour répondre à ces questions dans la Section 3. Le protocole expérimental et les résultats sont présentés dans la Section 4. Nous résumons nos conclusions et perspectives dans la Section 5.

2 Travaux antérieurs

2.1 Modèles de plongement de graphes de connaissances pour la prédiction de liens

Les MPGC représentent les entités et les relations d'un graphe de connaissances dans un espace vectoriel de dimension d, afin de capturer les régularités structurelles du graphe sous une forme compacte. Dans le cadre de la prédiction de liens, une fonction de score $\varphi(h, r, t)$ évalue la plausibilité d'un triplet (h, r, t) à partir des vecteurs, ou plongements, de h, r et t, permettant ainsi d'inférer de nouveaux faits. Les MPGC peuvent être regroupés en trois grandes familles.

Les méthodes géométriques modélisent les relations comme des transformations dans l'espace d'embedding, par exemple une translation (TransE [4]) ou une rotation (RotatE [31]).

Les méthodes par factorisation de tenseurs reposent sur des produits bilinéaires entre entités sous une transformation linéaire associée à la relation, comme DistMult [37] ou ComplEx [33].

Les approches neuronales apprennent directement la fonction de score, par exemple via des réseaux convolutifs (ConvE [8]), des GNN exploitant le *message passing* (RGCN [27]) ou des Transformers (HittER [6]).

Malgré leurs différences architecturales, ces modèles partagent un point commun : leur entraînement implique intrinsèquement de l'aléatoire. Dans ce travail, nous étudions l'impact de cet aléatoire sur la stabilité des MPGC.

2.2 Impacts de l'aléatoire sur les modèles de plongement de graphes de connaissances

Les contextes d'apprentissage automatique impliquent plusieurs facteurs qui dépendent de l'aléatoire, tels que la séparation des jeux de données (d'entrainement, d'évaluation, de test), l'initialisation des paramètres, ou le *dropout*. En traitement automatique des langues (TAL), [17] montrent

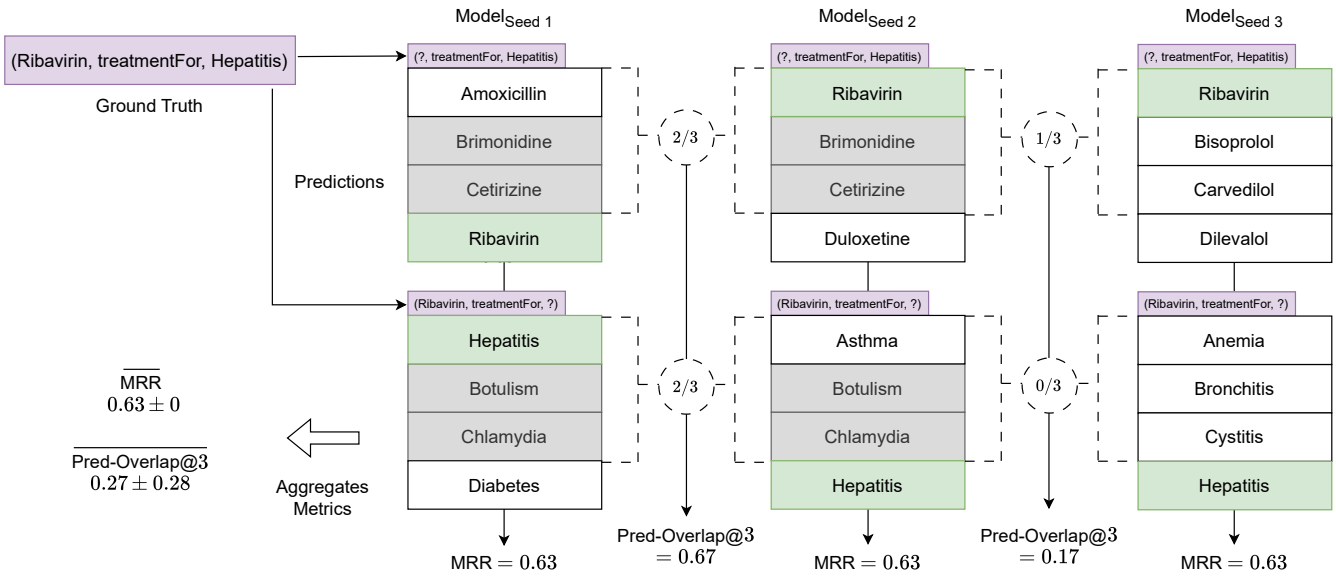


FIGURE 1 – Illustration de trois exécutions évaluées sur deux requêtes qui produisent le même MRR mais avec des différences substantielles dans les prédictions. Les exécutions 1 et 2 classent les entités de référence différemment mais prédisent tout de même des ensembles de candidats qui se chevauchent largement (Pred-Overlap@3 = 0.67). En revanche, les exécutions 2 et 3 attribuent le même rang à l'entité de référence, mais toutes les autres entités prédites diffèrent, ce qui conduirait à des graphes complétés entièrement différents (Pred-Overlap@3 = 0.17). Les cellules grises identifient les entités communes aux prédictions des différents modèles. Figure issue de [20].

que les graines aléatoires modifient les mécanismes d'attention ainsi que les mots jugés influents. De même, dans le cas des GNN, [29] mettent en évidence que l'entraînement sur différentes partitions des données conduit à des performances variables. Plus généralement, [18] définit la notion de multiplicité comme le fait que les métriques de performance sur l'ensemble de test dissimulent des variations substantielles au niveau local en raison des facteurs d'aléatoire. Ils proposent également deux métriques pour quantifier ce phénomène : la *Divergence* (*Discrepancy*) et l'*Ambiguïté*.

Dans la littérature sur les MPGC, l'impact de l'aléatoire pendant l'entraînement est rarement abordé. La section 4.3 de [3] étudie l'effet des graines d'initialisation et conclut que la plupart des modèles sont robustes car les métriques globales telles que le MRR présentent une faible variance. En revanche, [40] adapte la notion de multiplicité aux MPGC et montre que des modèles ré-entraînés avec des graines différentes peuvent produire des prédictions différentes au niveau des triplets, même lorsque leurs scores globaux (Hits@10) sont quasi identiques. Ils démontrent que cette instabilité peut être atténuée par un vote d'ensemble (*ensemble voting*). Cependant, leur évaluation se concentre uniquement sur le fait que l'entité cible (*gold entity*) apparaisse ou non dans le top-K des prédictions de chaque modèle, ignorant ainsi le reste des candidats classés. Nous soutenons qu'une telle perspective centrée sur la position prédite de l'entité de référence échoue à capturer toute l'ampleur du désaccord des modèles dans les contextes où la complétion de GC est utilisée de manière opérationnelle en ajoutant les prédictions du top-K au GC. Par exemple, les entrainements 2 et 3 de la Figure 1 ne montreraient aucun désaccord sous cette évaluation car, pour chaque requête, elles concordent systématiquement sur le fait que l'entité de référence se trouve à l'intérieur ou à l'extérieur du top-3. Pourtant, malgré cet accord apparent, les deux modèles prédisent des ensembles d'entités presque entièrement différents, ce qui donne des GC complétés radicalement différents. Cela souligne le besoin de mesures de stabilité qui révèlent la *divergence dans la complétion de GC prédite*, au-delà de l'accord sur la position prédite pour l'instance de référence, s'alignant sur le contexte opérationnel de la complétion de GC.

Outre la stabilité des prédictions, d'autres travaux ont étudié les impacts de l'aléatoire sur l'espace de plongement lui-même. Par exemple, [28] compare les topologies des espaces de plongement appris à travers différentes graines et signale une instabilité substantielle, conduisant à des prédictions individuelles instables même si la précision globale reste stable. Néanmoins, leur travail se concentre uniquement sur les modèles de plongement pour les graphes homogènes. Dans la littérature des MPGC, [13] et [12] étudient les propriétés des espaces de plongement, telles que la (non-)préservation de l'appartenance à une classe et la préservation de la proximité dans le graphe. Mais, à notre connaissance, aucun travail antérieur n'a examiné la stabilité de ces espaces par rapport à l'aléatoire.

Compte tenu du déploiement croissant des MPGC dans des contextes opérationnels, il y a un besoin pressant de protocoles d'évaluation de la stabilité qui reflètent l'utilisation dans le monde réel. Nous introduisons par conséquent des mesures qui quantifient la similarité des complétions de

GC produites par différents MPGC, ainsi que des métriques qui évaluent la cohérence de leurs espaces de plongement lorsque soumis à différentes graines aléatoires.

3 Mesurer l'(in)stabilité des MPGC

3.1 Préliminaires et notations sur l'entraînement des MPGC

Dans nos travaux, nous considérons la représentation formelle d'un GC comme un ensemble d'entités $\mathcal{E}$, un ensemble de relations $\mathcal{R}$, et une collection de triplets $\mathcal{T} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ encodant des faits. Chaque triplet (h, r, t) se compose d'une entité sujet h, d'une relation r, et d'une entité objet t, ex. : (France, aPourCapitale, Paris).

Nous nous concentrons sur les MPGC entraînés pour la PL. Dans ce cas, $\mathcal{T}$ est divisé en ensembles d'entraînement ($\mathcal{T}_{\text{train}}$), de validation ($\mathcal{T}_{\text{val}}$) et de test ($\mathcal{T}_{\text{test}}$), ce dernier étant utilisé pour l'évaluation du modèle. Étant donné un triplet de test $(h, r, t) \in \mathcal{T}_{\text{test}}$, la PL consiste à prédire les entités manquantes dans $(h, r, ?)$ ou $(?, r, t)$ en attribuant un score et un rang à toutes les entités du GC et en s'attendant à ce que l'entité de référence (h ou t) soit classée aussi haut que possible. La qualité du modèle est donc évaluée sur $\mathcal{T}_{\text{test}}$ avec des métriques basées sur le rang telles que le Rang Moyen (MR), le Rang Réciproque Moyen (MRR), et le Hits@K (généralement $K \in \{1, 3, 10\}$). L'entraînement des MPGC implique généralement une phase de recherche des hyperparamètres pour obtenir une **configuration d'hyperparamètres** $\boldsymbol{C}$ qui maximise le MRR ou le Hits@K de validation.

Un aspect moins exploré est celui des impacts de l'aléatoire pendant l'entraînement, associés au choix des graines aléatoires. Dans ce travail, nous étudions l'impact de ces facteurs aléatoires sur les résultats et les performances des modèles. Nous identifions quatre sources indépendantes d'aléatoire :

- **Échantillonnage négatif ($\mathcal{N}$)** corruption aléatoire des triplets positifs pour générer des exemples négatifs.
- **Ordonnancement des triplets ($\mathcal{O}$)** mélange (*shuffling*) aléatoire des triplets d'entraînement à chaque époque.
- **Initialisation des paramètres ($\mathcal{I}$)** initialisation aléatoire des paramètres du modèle.
- ***Dropout* ($\mathcal{D}$)** sous-ensemble aléatoire de neurones désactivé à chaque *batch*.

Nous définissons une **configuration de graines** $\mathfrak{S}$ comme un tuple qui assigne une valeur spécifique à chaque source d'aléatoire. Par exemple, $\mathfrak{S} = \{\mathfrak{S}_{\mathcal{N}} = s_a,\ \mathfrak{S}_{\mathcal{O}} = s_b,\ \mathfrak{S}_{\mathcal{I}} = s_c,\ \mathfrak{S}_{\mathcal{D}} = s_d\}$ fixe la graine $\mathfrak{S}_{\mathcal{N}}$ contrôlant l'échantillonnage négatif à s_a, et les autres graines suivent de manière analogue.

Une **instance** de modèle entraîné I est définie comme $I = \text{Instance}(\text{Dataset}, \text{MPGC}, \boldsymbol{C}, \mathfrak{S}, \mathcal{H})$, indiquant qu'un MPGC est entraîné sur le Dataset spécifié avec la configuration d'hyperparamètres $\boldsymbol{C}$, la configuration de graines $\mathfrak{S}$, et sur le matériel $\mathcal{H}$. Dans nos expériences, nous fixons $\mathcal{H}$ pour assurer des conditions d'exécution identiques à travers toutes les exécutions, et l'omettons donc de la notation. Nous omettons également le Dataset et le MPGC lorsque le contexte est clair, et désignons ainsi les instances de manière compacte par $I^{\boldsymbol{C}}_{\mathfrak{S}}$.

Soit $S = \{s_1, \ldots, s_n\}$ un ensemble de valeurs de graines. Nous définissons un **groupe de comparaison** G comme un ensemble d'instances obtenues en faisant varier une graine de $\mathfrak{S}$ sur S, tout en gardant les autres fixées à s_1. Par exemple, le groupe d'instances où seul l'échantillonnage négatif varie est $G^{\boldsymbol{C}}_{\mathcal{N}} = \left\{ I^{\boldsymbol{C}}_{\{\mathfrak{S}_{\mathcal{N}}=s_i,\ \mathfrak{S}_{\mathcal{O}}=s_1,\ \mathfrak{S}_{\mathcal{I}}=s_1,\ \mathfrak{S}_{\mathcal{D}}=s_1\}} \;\middle|\; s_i \in S \right\}$. Nous définissons $G^{\boldsymbol{C}}_{\mathcal{I}}$, $G^{\boldsymbol{C}}_{\mathcal{O}}$, $G^{\boldsymbol{C}}_{\mathcal{D}}$ de manière similaire pour les autres sources d'aléatoire, et $G^{\boldsymbol{C}}_{\mathcal{A}}$ pour le cas où toutes les graines varient conjointement sur S. Notre objectif est de déterminer si les prédictions et l'organisation de l'espace de plongement des instances au sein du même groupe restent cohérentes, en utilisant les métriques spatiales et de prédiction introduites ci-dessous.

3.2 Métriques

3.2.1 Chevauchement sur les prédictions et les voisins de l'espace de plongement

Pour évaluer l'instabilité d'un groupe $G^{\boldsymbol{C}}_{\mathcal{X}}$ pour toute source $\mathcal{X}$ d'aléatoire, nous nous appuyons sur des métriques qui capturent l'accord soit dans les prédictions, soit dans l'espace de plongement. Nous adoptons le chevauchement (*overlap*) top-K, qui quantifie l'intersection entre deux ensembles non ordonnés de taille K, allant de 0 (ensembles disjoints) à 1 (ensembles identiques). Il permet une interprétation simple, couvrant à la fois les cas d'insertion automatique et les approches semi-automatiques avec des humains examinant les candidats. Il s'applique également lorsque qu'il n'existe pas forcément de valeur de référence (par ex., recommandation, découverte de médicaments).

En alignement avec les métriques classiques de prédiction de liens telles que Hits@1 et Hits@10, nous nous concentrons sur Overlap@1 et Overlap@10. Nous appliquons cette mesure à la fois aux prédictions du top-K (Pred-Overlap@K) et aux K plus proches voisins dans l'espace de plongement (Space-Overlap@K), comme détaillé ci-dessous.

Pred-Overlap@K. La complétion de GC par prédiction de liens consiste à traiter les requêtes incomplètes $(h, r, ?)$ ou $(?, r, t)$ en leur associant les K meilleures entités prédites, selon des métriques usuelles comme Hits@K. Dans ce contexte, la métrique *Pred-Overlap@K* permet de mesurer la proportion de triplets communs que deux instances distinctes d'un même modèle insèreraient dans le GC. Elle quantifie ainsi la stabilité des prédictions entre deux entraînements.

Notons $\mathcal{Q}_{\text{test}}$ toutes les requêtes de test qui peuvent être dérivées des triplets de test, c.-à-d. les requêtes $(h, r, ?)$ et $(?, r, t)$ pour chaque triplet de test (h, r, t). Soit $\nu(I, q, K)$ les K meilleures prédictions renvoyées par l'instance I pour la requête q. Pour deux instances I_1 et I_2, Pred-

Overlap@K est défini comme :

$$\text{Pred-Overlap@}K(I_1, I_2) = \frac{1}{|\mathcal{Q}_{\text{test}}|} \sum_{q \in \mathcal{Q}_{\text{test}}} \frac{|\nu(I_1, q, K) \cap \nu(I_2, q, K)|}{K} \quad (1)$$

Une valeur proche de 1 indique que I_1 et I_2 produisent presque les mêmes prédictions. Ainsi, dans la Figure 1, les exécutions 1 et 2 partagent deux de leurs trois entités prédites pour chaque requête et ont donc un Pred-Overlap@3 de 0.67. Les exécutions 2 et 3 ne partagent qu'une seule entité pour la première requête et aucune pour la seconde, ce qui donne un Pred-Overlap@3 de seulement 0.17.

Space-Overlap@K. Nous cherchons également à mesurer l'accord entre les espaces de plongement E_1 et E_2 appris par deux instances I_1 et I_2. Les espaces de plongement ne sont pas directement comparables, car ils peuvent différer par diverses transformations, par ex., translation, rotation ou mise à l'échelle. Plutôt que d'aligner les espaces, nous évaluons la stabilité à travers la préservation du voisinage. Pour chaque entité $e_i \in \mathcal{E}$, notons $\mathcal{N}_K^{E_j}(e_i)$ ses K plus proches voisins dans l'espace de plongement E_j, calculés à l'aide de la distance ℓ_2 [1]. Space-Overlap@K reflète le degré de partage des voisins locaux entre les instances :

$$\text{Space-Overlap@}K(I_1, I_2) = \frac{1}{|\mathcal{E}|} \sum_{e_i \in \mathcal{E}} \frac{\left|\mathcal{N}_K^{E_1}(e_i) \cap \mathcal{N}_K^{E_2}(e_i)\right|}{K} \quad (2)$$

Agrégation sur les groupes. Pour obtenir des scores de stabilité pour un groupe $G_{\mathcal{X}}^{\boldsymbol{C}}$, nous calculons les métriques ci-dessus à travers toutes les paires $(I_1, I_2) \in G_{\mathcal{X}}^{\boldsymbol{C}}$, et rapportons la moyenne et l'écart type. Nous notons ces versions agrégées $\overline{\text{Pred-Overlap@}K}(G_{\mathcal{X}}^{\boldsymbol{C}})$ et $\overline{\text{Space-Overlap@}K}(G_{\mathcal{X}}^{\boldsymbol{C}})$.

3.2.2 Autres métriques

Nous avons également exploré plusieurs métriques supplémentaires issues de l'état de l'art et dont les résultats sont disponibles dans le papier original [20] et sur notre GitHub[2]. Toutes ces mesures conduisent à la même conclusion : les MPGC présentent une instabilité substantielle à travers les exécutions.

4 Expérimentation

4.1 Cadre expérimental

Modèles. Nous sélectionnons des MPGC représentatifs des différentes familles : **TransE** [4] et **RotatE** [31] (basés sur la géométrie), **DistMult** [37] et **ComplEx** [33] (méthode de factorisation de tenseurs), **ConvE** [8] (réseaux de neurones convolutifs), **RGCN** [27] (réseaux de neurones sur graphes), et un **Transformer** (*transformer*) sans contexte de voisinage (Section 2.1 de [6]).

1. La distance cosinus a également été testée, donnant des résultats similaires ; voir notre dépôt GitHub.[2]

Jeux de données. Nous évaluons nos modèles sur des jeux de données de GC standards de la littérature. Ainsi, nous considérons **WN18RR** [8], un sous-ensemble lexical de WordNet caractérisé par sa structure hiérarchique ; **FB15k-237** [32] et **Codex-S** [26], qui sont des sous-ensembles de Freebase et Wikidata, respectivement, et contiennent des faits du monde réel. Par souci d'exhaustivité, nous rapportons en plus dans notre dépôt GitHub[2] les résultats sur les jeux de données plus petits **Kinship** et **Nations**, couramment utilisés dans les travaux antérieurs et décrivant des relations familiales et politiques, respectivement.

Dans nos expériences, l'ensemble S de valeurs de graines a été fixé à $\{42, 283, 358, 698, 887\}$, et le matériel $\mathcal{H}$ a été fixé à un GPU *Nvidia GeForce RTX 2080 Ti (11 GiB)*, à l'exception de l'analyse spécifique de la variance matérielle menée à la Section 4.2.2.

PyKEEN et LibKGE ne séparent pas explicitement les différentes sources d'aléatoire, ce qui était nécessaire pour nos expériences contrôlées. Par conséquent, nous avons implémenté une bibliothèque de MPGC minimale qui fournit un contrôle fin et indépendant sur les quatre sources d'aléatoire. Elle est construite avec PyTorch, et les couches RGCN sont importées de PyTorch Geometric. Notre implémentation et les jeux de données utilisés sont accessibles publiquement. [2]

Recherche d'hyperparamètres. L'objectif de cette étape n'est pas d'identifier la configuration optimale du modèle, une tâche qui nécessiterait l'exploration d'un espace de recherche substantiellement plus vaste [25], mais plutôt de fournir un spectre significatif de configurations avec différents niveaux de performance en PL pour répondre à QR3. Nous avons fixé la configuration de graines à $\mathfrak{S} = \{\mathfrak{S}_{\mathcal{N}} = s_1,\ \mathfrak{S}_{\mathcal{O}} = s_1,\ \mathfrak{S}_{\mathcal{I}} = s_1,\ \mathfrak{S}_{\mathcal{D}} = s_1\}$. Nous avons fixé les taux de *dropout* pour les entités et les relations à 0.2, la taille du *batch* à 256, et le nombre d'échantillons négatifs à 500. Nous avons utilisé Xavier normal pour l'initialisation des poids, et nous employons la fonction de perte d'entropie croisée (*cross-entropy loss*), car [25] montre que ce choix donne systématiquement des résultats fiables. Plus de détails sont fournis sur GitHub.[2]

Puis, nous avons mené une recherche réduite d'hyperparamètres sur les dimensions de plongement $\{128, 256, 512\}$ et les taux d'apprentissage $\{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$. Les configurations résultantes ont été classées par MRR croissant et celles ne satisfaisant pas un seuil de performance minimum (MRR $<$ 0.05) ont été écartées. Nous avons ensuite sélectionné trois configurations représentatives : celle associée à la meilleure performance $\boldsymbol{C_B}$ (MRR le plus élevé), celle associée à la moins bonne performance $\boldsymbol{C_W}$ (MRR le plus bas), et enfin celle associée à la performance médiane $\boldsymbol{C_M}$, situé à la position médiane de la liste.

Notons que les meilleures configurations que nous avons identifiées obtiennent, presque toutes, des performances qui se situent dans une plage comparable à l'état de l'art (voir notre dépôt GitHub pour la comparaison détaillée[2]).

2. https://github.com/Wimmics/LinkPerdition

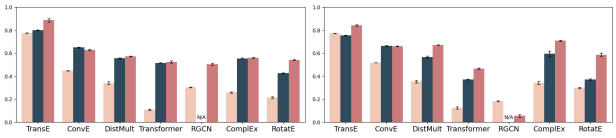
TABLE 1 – Moyenne et écart type du MRR des cinq exécutions dans $G_{\mathcal{A}}^{C_B}$, c.-à-d. la meilleure configuration de modèle avec toutes les graines variant conjointement sur S.[3]

Dataset	TransE	ConvE	DistMult	Transformer	RGCN	ComplEx	RotatE
WN18RR	.19 ± .00	.41 ± .00	.42 ± .00	.27 ± .01	.42 ± .00	.44 ± .00	.33 ± .04
FB15k-237	.32 ± .00	.32 ± .00	.31 ± .00	.30 ± .00	N/A	.32 ± .00	.29 ± .00
codex-s	.35 ± .00	.43 ± .00	.41 ± .00	.36 ± .01	.35 ± .02	.40 ± .00	.36 ± .00

4.2 Analyse des résultats

4.2.1 QR1 : stabilité par rapport à l'aléatoire

Si l'on ne considère que les scores agrégés tels que le MRR, les MPGC semblent être remarquablement stables. Le Table 1 reporte le MRR moyen et l'écart type sur les cinq exécutions de $G_{\mathcal{A}}^{C_B}$ (c.-à-d. la meilleure configuration de modèle avec toutes les composantes de graine variant conjointement). Les écarts apparaissent systématiquement faibles, suggérant que les instances entraînées avec la même configuration d'hyperparamètres mais avec des graines aléatoires différentes atteignent tout de même des performances prédictives globales comparables.



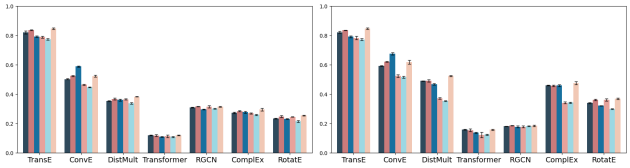
(a) $\overline{\text{Pred-Overlap@10}}$ (b) $\overline{\text{Space-Overlap@10}}$

FIGURE 2 – Analyse de stabilité sur tous les jeux de données et modèles.[3] Les couleurs représentent les jeux de données : WN18RR FB15k-237 CoDEx-S.

Cependant, nos résultats montrent que de faibles variations du MRR n'apportent aucune information sur la stabilité des prédictions individuelles. Les métriques agrégées tendent en effet à masquer une variabilité importante des prédictions, tant au sein d'une requête que d'une requête à l'autre. Dans la Figure 2a, la stabilité des prédictions est visualisée via $\overline{\text{Pred-Overlap@10}}$. Par exemple, deux instances de RotatE entraînées avec différentes graines ne partagent en moyenne que 4.3 des 10 meilleures entités prédites sur FB15k-237, chutant à 2.3 sur WN18RR. Un résultat similaire s'applique à la prédiction top-1[2], où les chevauchements sont proches ou inférieurs à 0.6 pour la plupart des paires (modèle, *dataset*), ce qui signifie que même en complétant le GC avec uniquement l'entité la mieux classée, environ 40 % des triplets différeraient d'une instance à l'autre. Les faibles écarts types suggèrent que l'ampleur de la divergence entre deux instances quelconques est elle-même remarquablement stable à travers toutes les paires.

Cette instabilité ne se limite pas aux prédictions, mais se manifeste également dans l'organisation de l'espace de plongement, visible via la métrique de $\overline{\text{Space-Overlap@10}}$ de la Figure 2b. Par example, lorsqu'on compare deux modèles DistMult entraînés séparément sur FB15k-237, on remarque qu'une même entité n'a, en moyenne, que 5.6 voisins en commun dans son top 10 des plus proches.

Prises ensemble, la variabilité dans les voisinages des espaces de plongement et la variabilité dans les classements prédits indiquent que les modèles apprennent des structures latentes différentes et encodent donc différentes connaissances. Pourtant, aucune de ces variabilités n'est reflétée dans les métriques de PL standards telles que le MRR, qui ne dépendent que du rang prédit des entités de référence. En conséquence, des modèles avec un MRR similaire peuvent produire des GC complétés divergents. Cette observation souligne la nécessité de métriques complémentaires pour évaluer plus précisément le comportement des MPGC, une exigence particulièrement importante dans les applications où les GC soutiennent la prise de décision.



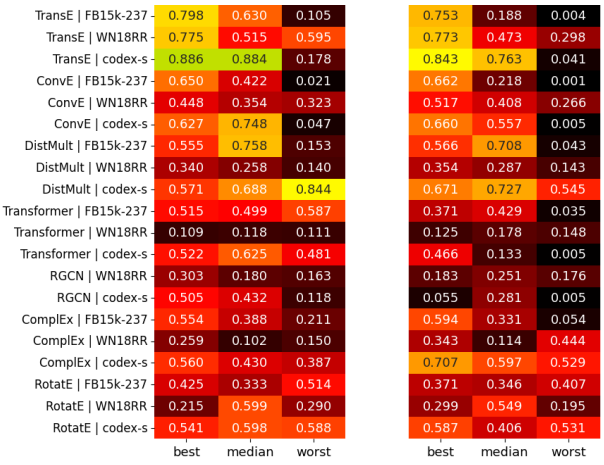
(a) $\overline{\text{Pred-Overlap@10}}$ (b) $\overline{\text{Space-Overlap@10}}$

FIGURE 3 – Impact des sources individuelles d'aléatoire sur WN18RR. Chaque source d'aléatoire (y compris le matériel) varie indépendamment tandis que toutes les autres sont maintenues fixes. Les couleurs indiquent le groupe de comparaison correspondant : $G_{\mathcal{I}}^{C_B}$ (init) $G_{\mathcal{D}}^{C_B}$ (*dropout*) $G_{\mathcal{N}}^{C_B}$ (neg) $G_{\mathcal{O}}^{C_B}$ (order) $G_{\mathcal{A}}^{C_B}$ (all) $G_{\mathcal{H}}^{C_B,\mathfrak{S}}$ (matériel). La catégorie "all" inclut toutes les sources d'aléatoire sauf le matériel.

4.2.2 QR2 : impact individuel des sources d'aléatoire

Nous examinons maintenant l'impact de chaque source d'aléatoire de manière isolée. En particulier, nous comparons la stabilité entre $G_{\mathcal{N}}^{C_B}$, $G_{\mathcal{I}}^{C_B}$, $G_{\mathcal{O}}^{C_B}$, $G_{\mathcal{D}}^{C_B}$ (en faisant varier les graines aléatoires indépendamment pour chaque source d'aléatoire sur un matériel fixe), et $G_{\mathcal{A}}^{C_B}$ (en faisant varier toutes les graines simultanément sur un matériel fixe). La Figure 3a et la Figure 3b rapportent, pour WN18RR, l'instabilité associée à chaque groupe, respectivement au niveau de la prédiction et dans l'espace de plongement. Les mesures d'instabilité des autres jeux de données considérés sont disponibles dans le dossier *Results* de notre dépôt GitHub[2]. Nos résultats indiquent que, pour chaque modèle, chaque source est indépendamment suffisante pour induire une instabilité d'ampleur comparable à la fois au niveau des plongements et de la prédiction. Cela met en évidence l'instabilité inhérente et persistante induite par les facteurs d'aléatoire dans l'entraînement des modèles. À travers tous les jeux de données et métriques, aucune graine particulière n'émerge comme un facteur d'instabilité dominant. Les variations d'instabilité observées semblent être plus fortement liées à l'architecture du modèle qu'à la source spécifique d'aléatoire. Notons que l'impact comparable de chaque source d'aléatoire valide le choix de conception de *frameworks* tels que Py-

3. Les résultats pour RGCN sur FB15k-237 sont omis ; sous notre pipeline, cette configuration dépasse le budget mémoire du GPU utilisé.



(a) $\overline{\text{Pred-Overlap@10}}$ (b) $\overline{\text{Space-Overlap@10}}$

FIGURE 4 – Impact des performances du modèle sur la stabilité.

Keen et LibKGE d'exposer uniquement une seule graine contrôlant tous les facteurs stochastiques.

De plus, nous définissons un groupe supplémentaire $G_{\mathcal{H}}^{\boldsymbol{C_B},\mathfrak{S}}$, dans lequel les configurations d'hyperparamètres et de graines $\boldsymbol{C_B}$ et $\mathfrak{S}$ sont fixes, tandis que le matériel varie. Chaque modèle dans ce groupe est donc entraîné sur un GPU différent. De manière frappante, cette configuration présente une instabilité de même ampleur que la variation des graines (en beige dans la 3b), soulignant le défi d'assurer la reproductibilité tant dans les environnements de recherche que de production, particulièrement dans les scénarios de *cloud computing* où le choix du matériel peut échapper au contrôle de l'utilisateur.

4.2.3 QR3 : relation entre stabilité et performances en prédiction de liens

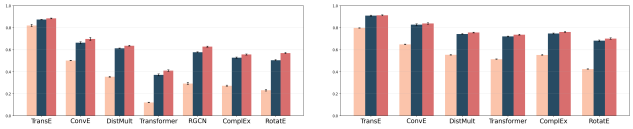
Nous étudions ici la relation, pour chaque modèle, entre les performances prédictives (MRR) d'une configuration et sa stabilité. Nous cherchons à identifier, pour chaque modèle, quelle configuration parmi $G_{\mathcal{A}}^{\boldsymbol{C_B}}$, $G_{\mathcal{A}}^{\boldsymbol{C_M}}$, et $G_{\mathcal{A}}^{\boldsymbol{C_W}}$, respectivement la meilleure, médiane, et la pire issues de la recherche d'hyperparamètres, présente la plus grande instabilité.

Nos observations (Figures 4a et 4b) montrent que les configurations de modèles peu performantes ($G_{\mathcal{A}}^{\boldsymbol{C_W}}$) ont tendance à être moins stables, ce qui correspond à l'intuition. Néanmoins, de manière contre-intuitive, nous n'observons aucune relation claire entre les performances et la stabilité des configurations de modèles les plus performantes ($G_{\mathcal{A}}^{\boldsymbol{C_B}}$) et de celles aux performances médianes ($G_{\mathcal{A}}^{\boldsymbol{C_M}}$), tant au niveau du plongement qu'à celui de la prédiction. La Figure 4 suggère donc que la stabilité est largement indépendante des performances de la configuration, à l'exception des instances mal entraînées, et qu'elle est plus fortement influencée par l'architecture du modèle.

4.3 Étude complémentaire : le vote comme remède à l'instabilité ?

Le vote a été proposé par Zhu et al. [40] pour atténuer l'instabilité, ce qui nous a motivés à évaluer son effet sur les métriques de stabilité que nous avons proposées. En particulier, nous avons utilisé le Vote Borda et le Vote par Score (*Range Vote*).

Pour cette expérience, pour chaque modèle, nous avons réutilisé les instances de modèles entraînées dans les sections précédentes et les avons réparties aléatoirement dans cinq groupes de cinq instances chacun, indépendamment de leur groupe d'origine $G_{\mathcal{X}}^{\boldsymbol{C}}$. Au sein de chaque groupe, nous avons appliqué la procédure de vote, produisant un modèle agrégé par groupe pour chaque méthode de vote. Nous avons ensuite calculé les mesures de stabilité entre ces modèles agrégés, pour chacune des deux procédures de vote.



(a) WN18RR (b) FB15k-237

FIGURE 5 – Amélioration du $\overline{\text{Pred-Overlap@10}}$ apportée par le vote pour tous les modèles sur FB15k-237 et WN18RR. Les couleurs représentent la méthode de vote : Baseline sans vote (instabilité de $G_{\mathcal{A}}^{\boldsymbol{C}}$) Borda Score.

La Figure 5 affiche les résultats obtenus, ainsi que le $\overline{\text{Pred-Overlap@10}}$ obtenu sur $G_{\mathcal{A}}^{\boldsymbol{C}}$ comme instabilité de référence (*baseline*). Elle démontre que le vote améliore la stabilité à travers tous les modèles. Cependant, l'instabilité est réduite mais non éliminée, et le vote s'accompagne d'un coût de calcul supplémentaire puisqu'il nécessite l'entraînement de multiples instances du même modèle. De plus, bien que le vote puisse être utilisé de manière simple sur les prédictions, il n'est pas applicable aux voisinages de l'espace de plongement. Par conséquent, l'instabilité au niveau de l'espace de plongement reste un problème non résolu.

5 Discussion et conclusion

Nous avons mené une analyse systématique de la stabilité de plusieurs MPGC sur plusieurs jeux de données, en nous concentrant sur l'impact des graines aléatoires (initialisation, ordonnancement des triplets, échantillonnage négatif, et *dropout*) ainsi que du matériel (GPU) sur les performances globales des MPGC, leurs prédictions, et l'organisation de l'espace de plongement. Bien que les métriques agrégées telles que le MRR et le Hits@K paraissent stables, elles dissimulent des instabilités prononcées. Au niveau des prédictions, les modèles produisent des candidats top-K divergents pour des triplets incomplets identiques. Au niveau du plongement, les voisinages latents sont également dépendants de la graine. Nos observations rejoignent des travaux antérieurs [12, 13], qui ont averti que les plon-

gements ne préservent pas nécessairement la sémantique ou la proximité du graphe. Cela souligne les risques de traiter les MPGC comme des compagnons neuronaux fidèles des GC et remet en question la validité des opérations en aval sur l'espace de plongement, telles que la mesure de similarité, le *clustering*, ou le raisonnement basé sur l'espace ou le voisinage [21, 35, 39]. Ensemble, ces observations soulèvent des préoccupations pour la complétion de GC et les applications en aval. La variabilité dépendante de la graine pourrait saper la fiabilité des systèmes qui s'appuient sur des GC complétés ou des MPGC pour la prise de décision.

Notre analyse montre que la modification de n'importe lequel de ces facteurs est suffisante pour induire une instabilité comparable, rendant la reproductibilité très sensible à des choix stochastiques apparemment mineurs. De manière préoccupante, l'impact des types de GPU sur la stabilité soulève des inquiétudes supplémentaires concernant la reproductibilité, particulièrement dans les environnements de calcul partagés ou *cloud*, où les utilisateurs ont un contrôle limité sur le matériel utilisé. Nous montrons également que les performances et la stabilité ne sont pas des critères alignés, soulignant la nécessité de les traiter comme des dimensions d'évaluation distinctes pour la sélection de modèles. Cela motive la conception et l'adoption de protocoles d'évaluation plus larges pour la prédiction de liens, qui reflètent non seulement les performances sur les *benchmarks*, mais aussi la robustesse requise dans les applications pratiques.

Pour les travaux futurs, nos résultats indiquent que certains modèles, comme TransE, présentent une stabilité relativement plus élevée. Cela suggère une voie prometteuse pour concevoir de nouvelles architectures ou procédures d'entraînement (par ex., des fonctions de perte tenant compte de la stabilité) qui imposent explicitement la stabilité. Pour ce faire, il serait précieux d'analyser l'impact des hyperparamètres (tels que le taux d'apprentissage, le taux de *dropout* ou la dimension de plongement) sur la stabilité. Une autre direction consiste à explorer si d'autres méthodes de MPGC comme RDF2Vec [24], NBFNet [41], ou MINERVA [7], des systèmes d'extraction de règles tels qu'AMIE+ [10] ou AnyBURL [19], ou des approches hybrides comme IterE [39], fournissent des prédictions et des espaces plus stables.

Remerciements

Ce travail a été soutenu par le gouvernement français, par le biais des investissements 3IA Côte d'Azur dans le projet géré par l'Agence Nationale de la Recherche (ANR) sous la référence ANR-23-IACL-0001. Les expériences présentées dans cet article ont été réalisées en utilisant la plateforme Grid'5000, soutenue par un groupement d'intérêt scientifique hébergé par Inria et incluant le CNRS, RENATER et plusieurs Universités ainsi que d'autres organismes (voir https://www.grid5000.fr). Ce travail est basé sur les travaux de l'action COST CA23147 GOBLIN - *Global Network on Large-Scale, Cross-domain and Multilingual Open Knowledge Graphs*, soutenu par COST (Coopération européenne en science et technologie, https://www.cost.eu).

Références

[1] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[2] Russa Biswas, Lucie-Aimée Kaffee, Michael Cochez, Stefania Dumbrava, Theis E. Jendal, Matteo Lissandrini, Vanessa López, Eneldo Loza Mencía, Heiko Paulheim, Harald Sack, Edlira Vakaj, and Gerard de Melo. Knowledge graph embeddings : Open challenges and opportunities. *TGDK*, 1(1) :4 :1–4 :32, 2023.

[3] Stephen Bonner, Ian P Barrett, Cheng Ye, Rowan Swiers, Ola Engkvist, Charles Tapley Hoyt, and William L Hamilton. Understanding the performance of knowledge graph embeddings in drug discovery. *Artificial Intelligence in the Life Sciences*, 2 :100036, 2022.

[4] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26 : 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795, 2013.

[5] Jiaoyan Chen, Hang Dong, Janna Hastings, Ernesto Jiménez-Ruiz, Vanessa López, Pierre Monnin, Catia Pesquita, Petr Skoda, and Valentina Tamma. Knowledge graphs for the life sciences : Recent developments, challenges and opportunities. *TGDK*, 1(1) :5 :1–5 :33, 2023.

[6] Sanxing Chen, Xiaodong Liu, Jianfeng Gao, Jian Jiao, Ruofei Zhang, and Yangfeng Ji. HittER : Hierarchical transformers for knowledge graph embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10395–10407, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[7] Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. Go for a walk and arrive at the answer : Reasoning over paths in knowledge bases using reinforcement learning, 2018.

[8] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence,*

(AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018, pages 1811–1818. AAAI Press, 2018.

[9] Mohamed H. Gad-Elrab, Vinh Thinh Ho, Evgeny Levinkov, Trung-Kien Tran, and Daria Stepanova. Towards utilizing knowledge graph embedding models for conceptual clustering. In Kerry L. Taylor, Rafael S. Gonçalves, Freddy Lécué, and Jun Yan, editors, *Proceedings of the ISWC 2020 Demos and Industry Tracks : From Novel Ideas to Industrial Practice co-located with 19th International Semantic Web Conference (ISWC 2020), Globally online, November 1-6, 2020 (UTC)*, volume 2721 of *CEUR Workshop Proceedings*, pages 281–286. CEUR-WS.org, 2020.

[10] Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. Fast rule mining in ontological knowledge bases with AMIE+. *VLDB J.*, 24(6) :707–730, 2015.

[11] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. Knowledge graphs. *ACM Computing Surveys (Csur)*, 54(4) :1–37, 2021.

[12] Nicolas Hubert, Heiko Paulheim, Armelle Brun, and Davy Monticolo. Do similar entities have similar embeddings ? In *The Semantic Web - 21st International Conference, ESWC 2024, Hersonissos, Crete, Greece, May 26-30, 2024, Proceedings, Part I*, volume 14664 of *Lecture Notes in Computer Science*, pages 3–21. Springer, 2024.

[13] Nitisha Jain, Jan-Christoph Kalo, Wolf-Tilo Balke, and Ralf Krestel. Do embeddings actually capture knowledge graph semantics ? In *The Semantic Web : 18th International Conference, ESWC 2021, Virtual Event, June 6–10, 2021, Proceedings*, page 143–159, Berlin, Heidelberg, 2021. Springer-Verlag.

[14] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. A survey on knowledge graphs : Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, 33(2) :494–514, 2022.

[15] Denis Krompaß, Stephan Baier, and Volker Tresp. Type-constrained representation learning in knowledge graphs. In *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I*, volume 9366 of *Lecture Notes in Computer Science*, pages 640–655. Springer, 2015.

[16] Markus Lanthaler, Richard Cyganiak, and David Wood. RDF 1.1 concepts and abstract syntax. W3C recommendation, W3C, February 2014. https ://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/.

[17] Pranava Madhyastha and Rishabh Jain. On model stability as a function of random seed, 2019.

[18] Charles T. Marx, Flavio du Pin Calmon, and Berk Ustun. Predictive multiplicity in classification, 2020.

[19] Christian Meilicke, Melisachew Wudage Chekol, Patrick Betz, Manuel Fink, and Heiner Stuckenschmidt. Anytime bottom-up rule learning for large-scale knowledge graph completion. *VLDB Journal*, 33(1) :131–161, 2024.

[20] Guillaume Méroué, Fabien Gandon, and Pierre Monnin. Link Prediction or Perdition : the Seeds of Instability in Knowledge Graph Embeddings. In *The Semantic Web - 23rd European Semantic Web Conference, ESWC 2026, Dubrovnik, Croatia, May 10-14, 2026, Proceedings*, The Semantic Web - 23rd European Semantic Web Conference, ESWC 2026, Dubrovnik, Croatia, May 10-14, 2026, Proceedings, Dubrovnik, Croatia, May 2026.

[21] Pierre Monnin, Chedy Raïssi, Amedeo Napoli, and Adrien Coulet. Discovering alignment relations with graph convolutional networks : A biomedical case study. *Semantic Web*, 13(3) :379–398, 2022.

[22] Natalya Fridman Noy, Yuqing Gao, Anshu Jain, Anant Narayanan, Alan Patterson, and Jamie Taylor. Industry-scale knowledge graphs : lessons and challenges. *Commun. ACM*, 62(8) :36–43, 2019.

[23] Heiko Paulheim. Knowledge graph refinement : A survey of approaches and evaluation methods. *Semantic Web*, 8(3) :489–508, 2017.

[24] Petar Ristoski and Heiko Paulheim. Rdf2vec : Rdf graph embeddings for data mining. In Paul Groth, Elena Simperl, Alasdair Gray, Marta Sabou, Markus Krötzsch, Freddy Lecue, Fabian Flöck, and Yolanda Gil, editors, *The Semantic Web – ISWC 2016*, pages 498–514, Cham, 2016. Springer International Publishing.

[25] Daniel Ruffinelli, Samuel Broscheit, and Rainer Gemulla. You CAN teach an old dog new tricks ! on training knowledge graph embeddings. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[26] Tara Safavi and Danai Koutra. CoDEx : A Comprehensive Knowledge Graph Completion Benchmark. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8328–8350, Online, November 2020. Association for Computational Linguistics.

[27] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer, 2018.

[28] Tobias Schumacher, Hinrikus Wolf, Martin Ritzert, Florian Lemmerich, Martin Grohe, and Markus Strohmaier. *The Effects of Randomness on the Stability of Node Embeddings*, page 197–215. Springer International Publishing, 2021.

[29] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation, 2019.

[30] Zequn Sun, Qingheng Zhang, Wei Hu, Chengming Wang, Muhao Chen, Farahnaz Akrami, and Chengkai Li. A benchmarking study of embedding-based entity alignment for knowledge graphs. *Proceedings of the VLDB Endowment*, 13(11) :2326–2340, 2020.

[31] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate : Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[32] Kristina Toutanova and Danqi Chen. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality, CVSC 2015, Beijing, China, July 26-31, 2015*, pages 57–66. Association for Computational Linguistics, 2015.

[33] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org, 2016.

[34] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. Ripplenet : Propagating user preferences on the knowledge graph for recommender systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 417–426, New York, NY, USA, 2018. Association for Computing Machinery.

[35] Bo Xiong, Mojtaba Nayyeri, Ming Jin, Yunjie He, Michael Cochez, Shirui Pan, and Steffen Staab. Geometric relational embeddings : A survey. *CoRR*, abs/2304.11949, 2023.

[36] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[37] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[38] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. QA-GNN : Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 535–546, Online, June 2021. Association for Computational Linguistics.

[39] Wen Zhang, Bibek Paudel, Liang Wang, Jiaoyan Chen, Hai Zhu, Wei Zhang, Abraham Bernstein, and Huajun Chen. Iteratively learning embeddings and rules for knowledge graph reasoning. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 2366–2377. ACM, 2019.

[40] Yuqicheng Zhu, Nico Potyka, Mojtaba Nayyeri, Bo Xiong, Yunjie He, Evgeny Kharlamov, and Steffen Staab. Predictive multiplicity of knowledge graph embeddings in link prediction. In *Findings of the Association for Computational Linguistics : EMNLP 2024*, pages 334–354, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[41] Zhaocheng Zhu, Zuobai Zhang, Louis-Pascal Xhonneux, and Jian Tang. Neural bellman-ford networks : A general graph neural network framework for link prediction. *Advances in Neural Information Processing Systems*, 34, 2021.