Par Mohammed Bouayoun, Janvier 2025 (bouyao@gmail.com)

Chapitre 3 : Représentation et Modélisation des Entités d'Information

3.1. Introduction

Dans ce chapitre, nous nous concentrons sur la **représentation** et la **modélisation** des entités d'information au sein du DSL (Deep Synergy Learning). Tout système d'apprentissage repose sur une certaine façon de **décrire** ses données, dans le DSL, ce sont les entités et leurs propriétés qui déterminent la manière dont la synergie $\omega_{i,j}$ sera calculée et mise à jour. La présentation qui va suivre servira de **pierre angulaire** aux chapitres ultérieurs (notamment Chapitre. 4 sur la dynamique d'auto-organisation, Chapitre. 5 sur l'architecture SCN, etc.), qui tireront directement profit de la qualité de la modélisation initiale.

3.1.1. Contexte et Motivation

La représentation des entités est un **facteur crucial** dans le DSL, si elle est inadéquate, la fonction de synergie $S(i,j)$ ne peut pas refléter correctement les relations entre entités, risquant de compromettre la formation de clusters pertinents ou l'adaptation du réseau. À l'inverse, une bonne modélisation facilite la détection de liens inattendus et l'émergence de structures synergiques riches. L'enjeu est d'autant plus grand que les entités peuvent être de différentes **natures** (images, textes, signaux, concepts symboliques), obligeant à manier des représentations **hétérogènes**.

3.1.1.1. Rappel du Livre : Le DSL Repose sur des Entités d'Information Dont la Qualité de Représentation Conditionne le Calcul de la Synergie et la Dynamique $\omega_{i,j}$

Il a été souligné dans les chapitres précédents que le **Deep Synergy Learning (DSL)** organise la connaissance autour d'un réseau de pondérations $\omega_{i,j}$ ajustées par une règle d'auto-organisation, comme

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\left[\mathrm{S}(i,j) - \tau\,\omega_{i,j}(t)\right].$$

Cette règle ne prend tout son sens que si la **synergie** $\mathrm{S}(i,j)$ entre les entités $\mathcal{E}_i$ et $\mathcal{E}_j$ fournit un **indicateur fiable** de leur affinité ou de leur compatibilité. La structure entière du **Réseau de connexion synergétique** (**Synergistic Connection Network (SCN)** en anglais) repose ainsi sur la capacité à estimer correctement les liens potentiellement utiles, ce qui dépend étroitement de la **qualité** des **représentations** de chaque entité.

Pour comprendre l'importance de ce point, il suffit de noter que la **dynamique** de l'auto-organisation, fondée sur la formule ci-dessus, renforce ou diminue les liaisons $\omega_{i,j}$ en fonction de $\mathrm{S}(i,j)$. Dans le cas où la description des entités serait lacunaire ou bruitée, la **mesure** $\mathrm{S}(i,j)$ risque de **s'égarer**, renforçant des liens non pertinents ou affaiblissant des liens au contraire prometteurs. La conséquence peut être un **réseau** qui ne capture pas la structure réelle des données, des **clusters** peu significatifs ou des phénomènes oscillatoires ne reflétant plus la nature véritable du système.

À l'inverse, si chaque entité $\mathcal{E}_i$ est modélisée sous un format riche, cohérent et représentatif de ses caractéristiques internes, la **synergie** $\mathrm{S}(i,j)$ devient un détecteur fiable des regroupements logiques ou des similitudes profondes. Il en résultera une **auto-organisation** claire, faisant émerger des liens forts là où la complémentarité est avérée, et réduisant ceux qui n'apportent que peu de cohérence.

Dans l'optique du **DSL**, cette qualité de représentation peut être sub-symbolique (vecteurs, embeddings, distributions de probabilité) ou symbolique (règles logiques, concepts, graphes), ou encore un **mélange** de ces approches dans un cadre **hybride**.

Les **méthodes** d'apprentissage profond (transformers, autoencodeurs, réseaux génératifs, etc.) fournissent souvent des vecteurs latents puissants et expressifs pour $\mathcal{E}_i$, tandis que les systèmes symboliques (ontologies, logiques descriptives, axiomes) véhiculent une information formelle ou sémantique plus explicite. Chaque choix a un **impact** direct sur le calcul

$$\mathrm{S}(i,j) \;=\; f\big(\mathbf{r}(i),\ \mathbf{r}(j)\big),$$

où $\mathbf{r}(i)$ désigne la représentation de $\mathcal{E}_i$. L'**exactitude** et la **complétude** de $\mathbf{r}(i)$ s'avèrent alors cruciales pour guider le processus d'auto-organisation vers des **clusters** ou des **groupements** factuellement cohérents, révélant de manière fiable la structure cachée des données.

Du point de vue théorique, il apparaît donc que la **représentation** constitue un *pré-requis* indispensable à l'efficacité de l'ensemble du **DSL**. Même une très bonne règle de mise à jour $\omega_{i,j}$ échouera à produire une auto-organisation pertinente si $\mathrm{S}(i,j)$ est trompeuse, ce qui arrive lorsque $\mathbf{r}(i)$ et $\mathbf{r}(j)$ ne contiennent pas d'informations réellement discriminantes ou significatives. À l'inverse, une représentation soigneusement conçue amplifiera l'action de la synergie, rendant la **mise à jour** plus rapide et plus stable vers un **réseau** présentant des **structures** solides, qu'il s'agisse de clusters distincts ou de sous-réseaux de coopération thématique.

3.1.1.2. Besoin d'Aborder la Diversité des Entités : Sub-Symboliques (Vecteurs, Embeddings), Symboliques (Règles, Concepts), Hybrides, Multimédias…

Dans le **Deep Synergy Learning**, un **enjeu majeur** réside dans la prise en compte de **données** dont la nature dépasse le simple cadre vectoriel homogène. Les entités manipulées peuvent relever du **sub-symbolique** (des vecteurs numériques issus de l'apprentissage profond), du **symbolique** (des règles logiques ou des concepts formels) ou d'une **hybridation** de ces deux approches. Plus encore, elles peuvent provenir de sources **multimédias** (images, sons, textes, vidéos) que le **DSL** souhaite traiter de manière intégrée. Cette **diversité** impose donc de **définir** des modèles de représentation appropriés, car la fonction de **synergie** $\mathbf{S}(i,j)$ dépend directement de la manière dont $\mathbf{r}(i)$ et $\mathbf{r}(j)$ sont encodées.

Un premier registre rassemble les entités **sub-symboliques**, généralement décrites par des **vecteurs** ou des **embeddings**. Une telle approche s'avère particulièrement efficace pour modéliser des données continues ou massives, par exemple, un extrait textuel peut être projeté dans un espace latent à l'aide d'un modèle de langue, ou une image être convertie en un vecteur de caractéristiques via un **réseau neuronal convolutif**. En pratique, la **synergie** $\mathbf{S}(i,j)$ se ramène souvent à une **similarité vectorielle** (cosinus, euclidienne), notée

$$\mathbf{S}(i,j) \;=\; \mathrm{sim}\left(\mathbf{r}_{\mathrm{subsym}}(i),\ \mathbf{r}_{\mathrm{subsym}}(j)\right),$$

où $\mathbf{r}_{\mathrm{subsym}}(i)$ est un **embedding** associé à l'entité $\mathcal{E}_i$. De tels vecteurs favorisent la robustesse au bruit et la flexibilité face aux évolutions futures, mais peuvent se heurter à une **interprétabilité**

limitée, puisqu'il demeure malaisé de comprendre le contenu d'un embedding hautement dimensionnel.

À l'opposé, les entités **symboliques** sont décrites via des **blocs de règles**, des **concepts** structurés en **graphes sémantiques** ou en **logiques formelles**. Le calcul de la synergie $\mathbf{S}(i,j)$ se fonde alors sur des mesures de **compatibilité** entre axiomes, sur des **distances** dans une ontologie, ou sur des **critères** de cohérence dans un espace conceptuel. On obtient, par exemple,

$$\mathbf{S}(i,j) \;=\; \mathrm{compat}\big(\mathcal{C}_i,\, \mathcal{C}_j\big),$$

où $\mathcal{C}_i$ et $\mathcal{C}_j$ désignent des entités logiques. Cette représentation **symbolique** jouit d'une **lisibilité** élevée, car chaque entité est énoncée sous forme de règles ou de propriétés aisément contrôlables. Néanmoins, elle peut se révéler **rigide** et peu adaptée au traitement de données massives ou bruitées, nécessitant une maintenance stricte de la **consistance** logique.

Dans de nombreuses applications, un **format hybride** apparaît plus naturel, chaque entité combine des composantes **sub-symboliques** et **symboliques**. Par exemple, un document textuel peut être associé à un **embedding** qui résume le contenu statistique global, tout en étant pourvu d'un **ensemble de mots-clés** ou de relations logiques qui en précisent la structure ou la sémantique. Dès lors, la fonction de synergie $\mathbf{S}(i,j)$ doit agréger à la fois une **mesure vectorielle** et un **score symbolique**. On peut imaginer une formule

$$\mathbf{S}(i,j) \;=\; \alpha\, \mathrm{sim}\left(\mathbf{r}_{\mathrm{subsym}}(i),\, \mathbf{r}_{\mathrm{subsym}}(j)\right) \;+\; (1-\alpha)\, \mathrm{compat}\big(\mathcal{C}_i,\, \mathcal{C}_j\big),$$

la pondération $\alpha \in [0,1]$ modulant l'importance relative du versant sub-symbolique et du versant symbolique. L'attrait de cette solution **hybride** réside dans sa souplesse. Elle allie la **richesse expressive** des approches symboliques à la **robustesse** et à la **continuité** des embeddings profonds.

Enfin, le DSL vise fréquemment le traitement de **données multimédias**, telles que des **images**, des **sons**, ou des **clips vidéo**. Ces entités peuvent elles-mêmes être considérées comme **sub-symboliques**, au sens où chaque support est converti en un vecteur (ou un tenseur) issu d'un réseau de neurones approprié (par exemple, un **spectrogramme** pour l'audio, une **carte de caractéristiques** pour l'image). Cependant, des **métadonnées symboliques** (catégories, tags, règles de provenance) peuvent en outre accompagner ces vecteurs. De la sorte, la **synergie** inclut un terme de similarité visuelle ou auditive, et, simultanément, un terme de compatibilité sémantique. On formule alors,

$$\mathbf{S}(i,j) \;=\; \mathrm{sim}_{\mathrm{media}}\big(\mathbf{m}_i,\, \mathbf{m}_j\big) \;+\; \mathrm{sim}_{\mathrm{sym}}\left(\mathbf{r}_{\mathrm{sym}}(i),\, \mathbf{r}_{\mathrm{sym}}(j)\right).$$

Cette fusion multimodale encourage l'**émergence** de clusters plus riches, dotés d'une cohérence simultanément perceptive et conceptuelle.

La **réalité** des usages implique donc que le **DSL** accueille une vaste **diversité** de formats, depuis la pure vectorisation jusqu'à l'**ontologie** la plus stricte, en passant par des **mélanges** sub-symbolique et symbolique, et par des **données** clairement multimédias. Cette hétérogénéité, loin d'être un obstacle, constitue un **levier** où la synergie s'y nourrit d'interactions inattendues et des **clusterisations** inédites peuvent apparaître. La **condition** demeure néanmoins cruciale. Chaque entité doit être **décrite** de manière à dévoiler son essence ou sa fonction, faute de quoi la mesure $\mathbf{S}(i,j)$ ne sera qu'un miroir déformant qui freinera l'**auto-organisation** synergique.

3.1.2. Objectifs du Chapitre

Dans ce chapitre, nous allons détailler la **représentation** et la **modélisation** des entités d'information, un aspect central pour la réussite du **Deep Synergy Learning** (DSL). Il s'agit de comprendre :

Les principes fondamentaux de la représentation dans le DSL :

- Pourquoi et comment décrire une entité $\mathcal{E}_i$?
- Quels critères de qualité influent sur la synergie $S(i,j)$?

Les différentes formes de modélisation (vecteurs, graphes, ontologies, symboliques, hybrides) et leur **impact** sur la dynamique d'auto-organisation :

- Des vecteurs ou embeddings pour les données sub-symboliques,
- Des blocs de règles ou concepts pour les entités logiques,
- Des structures mixtes pour combiner à la fois la puissance statistique et l'interprétabilité.

Les concepts avancés (embeddings profonds, transformers, représentations symboliques complexes, synergie n-aire, etc.) permettant de **pousser plus loin** la capacité d'un DSL à découvrir des patterns riches et adaptatifs.

En d'autres termes, l'objectif est de **baliser** l'espace des représentations possibles, afin d'équiper le lecteur des connaissances nécessaires pour modeler correctement chaque type d'entité ($\mathcal{E}_i$), en tenant compte de la diversité (images, sons, textes, règles logiques) et de la flexibilité (adaptation en temps réel, extensions multiformes). Ce faisant, on jette les bases permettant au DSL de tirer pleinement parti de la notion de **synergie** pour organiser et valoriser ces données multiples.

3.1.2.1. Rôle Fondamental de la Représentation

Dans le **Deep Synergy Learning (DSL)**, la **représentation** de chaque entité $\mathcal{E}_i$ constitue un **pivot** essentiel, car elle détermine la **qualité** de la **synergie** $S(i,j)$ et, partant, la **dynamique** de mise à jour des pondérations $\omega_{i,j}$. L'ensemble des mécanismes d'**auto-organisation**, qui aboutissent à la formation de **clusters** cohérents ou à l'émergence de **structures** plus complexes au sein du **Synergistic Connection Network (SCN)**, dépend donc étroitement de la façon dont chaque entité est encodée ou décrite.

Impact direct sur la synergie $S(i,j)$

La fonction $S(i,j)$ joue un rôle de **filtre** en évaluant dans quelle mesure deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ sont jugées **compatibles** ou **similaires**. La **définition** même de S varie selon la nature des entités, mais fait systématiquement intervenir leur **représentation**. Si celle-ci est sub-symbolique (vecteur d'**embedding**, par exemple), on peut recourir à une mesure de **similarité vectorielle** comme la cosinus ou la gaussienne :

$$\mathrm{S}(i,j) \;=\; \exp(-\alpha \;\|\, \mathbf{r}(i) - \mathbf{r}(j) \,\|) \quad \text{ou} \quad \frac{\mathbf{r}(i) \cdot \mathbf{r}(j)}{\|\, \mathbf{r}(i) \,\| \,\|\, \mathbf{r}(j) \,\|}.$$

Si la représentation est symbolique, S peut reposer sur un **calcul** de cohérence logique, de distance ontologique ou de correspondance conceptuelle :

$$\mathrm{S}(i,j) \;=\; f_{\text{logic}}\big(\mathbf{C}(i),\, \mathbf{C}(j)\big).$$

Dans tous les cas, la **pertinence** de $\mathrm{S}(i,j)$ dépend de la **capacité** des descripteurs $\mathbf{r}(i)$ ou $\mathbf{C}(i)$ à rendre compte des attributs substantiels de l'entité $\mathcal{E}_i$. En d'autres termes, une **incomplétude** ou un **bruit** excessif dans ces descripteurs entraîne une **approximation** inadéquate de la synergie, ce qui peut fausser la dynamique globale du réseau.

Conséquences sur l'auto-organisation

Le **principe** d'auto-organisation dans le DSL est de laisser chaque lien $\omega_{i,j}(t)$ évoluer selon une **règle** inspirée d'une descente d'énergie, telle que :

$$\omega_{i,j}(t+1) \;=\; \omega_{i,j}(t) \;+\; \eta\big[\mathrm{S}(i,j) \;-\; \tau\, \omega_{i,j}(t)\big].$$

La convergence ou la stabilité de cette équation dépend d'une part de **l'information** captée par $\mathrm{S}(i,j)$. Une bonne représentation garantit que la **force** $\mathrm{S}(i,j)$ saura distinguer avec finesse les entités véritablement proches (renforcement des liens) de celles sans réelle corrélation (faible synergie et diminution progressive de $\omega_{i,j}$). À l'inverse, si $\mathbf{r}(i)$ ne contient pas les caractéristiques déterminantes, le réseau peut s'orienter vers des **clusters** peu informatifs ou s'enliser dans des **oscillations** inutiles.

Cette relation entre **représentation** et **dynamique** d'auto-organisation se révèle cruciale dans la mise en œuvre pratique du DSL. En effet, une représentation solide permet de poser un **pas d'apprentissage** η plus élevé, accélérant la consolidation des liens. Au contraire, une représentation frêle ou sujette au **bruit** impose un réglage prudent de η afin d'éviter des ajustements erratiques de $\omega_{i,j}$.

Représentation comme "passeport commun"

Le **DSL** se veut généraliste et vise à traiter des entités issues d'une **variété** de domaines (images, langage, signaux biomédicaux, règles logiques, etc.). Pour pouvoir **comparer** ces entités au sein d'un même **SCN**, il faut établir un **passeport** ou un **langage commun**, c'est-à-dire une représentation dans laquelle $\mathrm{S}(i,j)$ devient calculable même si $\mathcal{E}_i$ et $\mathcal{E}_j$ proviennent de modalités différentes. Ainsi, un document texte peut être encodé par un **embedding** (BERT, GPT, etc.) tandis qu'une image l'est par un CNN, et l'on projette ces descriptions dans un espace latent partagé permettant de définir, par exemple,

$$\mathrm{S}(i,j) \;=\; \exp(-\,\|\, \Phi\left(\mathbf{r}_{\text{image}}(i)\right) \;-\; \Phi\big(\mathbf{r}_{\text{text}}(j)\big) \,\|^2)$$

où Φ représente la fonction d'encodage multimodale. Cette approche assure la **cohérence** des liens de synergie, puisqu'on manipule des "vecteurs" homogènes malgré l'hétérogénéité initiale.

Robustesse et sensibilité au bruit

Une représentation de **qualité** se caractérise en outre par sa capacité à **absorber** le bruit des données et à **conserver** les informations essentielles pour le calcul de S. Un embedding textuel doit par exemple demeurer stable face à de légères variations lexicales, tandis qu'une représentation d'image doit tolérer des perturbations d'éclairage ou de perspective. Si l'on ne parvient pas à extraire ces invariances, l'estimation de la synergie est parasitée par des fluctuations indues, et la mise à jour de $\omega_{i,j}$ devient chaotique.

Il s'ensuit que la **stabilité** des descripteurs est un facteur-clé pour la réussite de l'auto-organisation. Dans un cadre "online", où les représentations peuvent elles-mêmes être affinées au cours du temps, les fluctuations trop rapides de $\mathbf{r}(i)$ risquent de semer la confusion dans les pondérations $\omega_{i,j}$, engendrant des **oscillations** durables ou une absence de convergence stable.

Dans les sections suivantes (3.2 à 3.5), on examinera plus en détail la manière dont on peut modéliser une entité $\mathcal{E}_i$ en recourant à des approches **sub-symboliques** (vecteurs, embeddings neuronaux), **symboliques** (règles, ontologies, structures logiques) ou **hybrides** (combinaison de descripteurs neuronaux et de blocs de connaissances expertes). On illustrera également comment cette variété de représentations se traduit dans la fonction S, influençant l'ensemble de la dynamique synergique. Ainsi, le **DSL** se dote d'un cadre flexible où le **choix** de la représentation devient un levier majeur pour révéler des **synergies** pertinentes dans des univers de données de plus en plus hétérogènes.

3.1.2.2. Détails des Formes de Modélisation (Vecteurs, Graphes, Ontologies…) et Leur Impact sur la Synergie

Le **Deep Synergy Learning (DSL)** manipule des entités $\mathcal{E}_i$ dont la **représentation** peut relever de plusieurs paradigmes. On peut décrire chaque entité par un **vecteur** (ou embedding), par un **graphe** interne, par un **ensemble de règles symboliques** (ou tout autre formalisme logique), ou encore adopter une **approche hybride** combinant ces perspectives. Dans chacun de ces cas, la **synergie** $\mathrm{S}(i,j)$ — définie entre deux entités $\mathcal{E}_i, \mathcal{E}_j$ — se construit à partir d'un opérateur de **similarité** (ou de **compatibilité**) approprié. Le choix de la **forme** de modélisation influe ainsi directement sur la **nature** de $\mathrm{S}(i,j)$, sur la **complexité** du calcul et sur la **qualité** de l'auto-organisation qui en découle.

A. Entités Vectorielles

Dans de nombreux cas, les entités se décrivent à l'aide d'**embeddings** ou de **vecteurs** $\mathbf{x}_i \in \mathbb{R}^d$. Cette approche est couramment employée pour encoder des images, des segments textuels, des extraits audio ou d'autres données sub-symboliques via un réseau neuronal ou une procédure d'extraction de caractéristiques. La fonction S peut alors prendre la forme d'une **distance** (euclidienne, Minkowski) ou d'une **similarité** (produit scalaire, cosinus) :

$$\mathrm{S}_{\text{vectoriel}}(i,j) = \phi(\mathbf{x}_i, \mathbf{x}_j),$$

où ϕ est un opérateur de similarité comme

$$\phi(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\| \mathbf{x}_i \| \| \mathbf{x}_j \|}.$$

L'avantage principal réside dans la **simplicité** du calcul où la mise à jour des pondérations $\omega_{i,j}$ peut alors s'effectuer rapidement, favorisant une bonne **scalabilité** lorsque le **Synergistic Connection Network (SCN)** s'étend à un grand nombre d'entités. De plus, les représentations vectorielles, si elles sont correctement entraînées ou extraites, capturent des **propriétés** sémantiques ou perceptives pertinentes (ex. embeddings textuels pour la sémantique linguistique, features CNN pour l'analyse d'images).

En contrepartie, ces modèles vectoriels peuvent souffrir d'une **opacité** où ils sont moins interprétables et leur dimension peut croître, ce qui alourdit le traitement si le SCN doit parcourir un espace vectoriel de grande taille. La **mise en œuvre** de l'auto-organisation demeure néanmoins directe, puisqu'on effectue souvent un simple calcul de **similarité** pour chaque paire (i, j).

B. Entités Graphiques

Une autre forme de description se fonde sur des **graphes** internes. Chaque entité $\mathcal{E}_i$ est un graphe G_i, par exemple un graphe de relations (conceptuels, biologiques, topologiques) ou un graphe de dépendances. La **synergie** $\mathrm{S}(i, j)$ s'exprime alors par un **score** de correspondance structurale, comme

$$\mathrm{S}_{\text{graphique}}(i, j) = \Psi(G_i, G_j).$$

On rencontre notamment la **graph edit distance** (GED), mesurant le coût minimal de transformations (insertion/suppression de nœuds ou d'arêtes) pour passer de G_i à G_j. Une fonction de similarité exponentielle, de la forme

$$\exp\left(-\alpha\, \mathrm{GED}(G_i, G_j)\right),$$

peut alors servir de **synergie**.

Cette approche est plus **riche** que les simples vecteurs où elle autorise une modélisation explicite de la structure interne, mais elle est plus **coûteuse** où la comparaison de graphes s'avère un problème algorithmique complexe, souvent NP-difficile pour l'isomorphisme de graphes généraux, impliquant ainsi un surcroît de calcul. Sur le plan de la **dynamique** du SCN, l'évaluation de $\mathrm{S}_{\text{graphique}}(i, j)$ peut s'avérer un **goulot d'étranglement** si le réseau compte de nombreuses entités structurées.

C. Entités Logiques ou Symboliques

Dans un univers **symbolique**, chaque entité $\mathcal{E}_i$ peut être un ensemble de **règles**, un **théorème**, un fragment d'**ontologie** ou un **module logique** exprimant des propriétés formelles. Le calcul de

$S(i,j)$ repose alors sur une **compatibilité** conceptuelle, par exemple un degré de "contradiction nulle" ou de "similarité sémantique" entre deux groupes d'axiomes :

$$S_{\text{symbolique}}(i,j) = \text{compat}\Big(\{\text{règles}_i\}, \{\text{règles}_j\}\Big).$$

On peut envisager une fonction de synergie mesurant la **proximité** de deux ontologies (recouvrement d'axiomes) ou la **cohérence** de l'union $\mathcal{E}_i \cup \mathcal{E}_j$. L'attrait principal réside dans l'**interprétabilité** où l'on sait **expliquer** pourquoi deux entités s'avèrent compatibles ou non et l'on peut procéder à des **déductions** explicites. Toutefois, le formalisme symbolique exige une mise à jour **plus rigide**, et demeure peu tolérant au **bruit** ou à l'incertitude.

D. Modélisation Hybride et Multimodale

Dans de nombreux contextes, les entités $\mathcal{E}_i$ se décrivent à la fois par des **caractéristiques sub-symboliques** (e.g. embeddings de textes, d'images) et par des **attributs symboliques** (e.g. champs sémantiques, règles associées, métadonnées). Il s'agit alors d'une **modélisation hybride**, où la synergie combine plusieurs fonctions de similarité :

$$S_{\text{hybride}}(i,j) = \lambda\, \phi\big(\mathbf{x}_i, \mathbf{x}_j\big) + (1-\lambda)\, f_{\text{logic}}\big(\{\text{axiomes}_i\}, \{\text{axiomes}_j\}\big),$$

avec $\lambda \in [0,1]$ un hyperparamètre de balance. Cette approche **hybride** rend possible l'exploitation simultanée d'**informations** quantitatives et qualitatives, donnant une vision plus complète de la relation entre entités. Dans un cadre multimodal (images, textes, sons, etc.), on peut adapter la fonction $\phi(\cdot,\cdot)$ à chaque modalité et agréger les scores partiels par pondération ou par un opérateur plus sophistiqué.

La **richesse** d'un tel modèle s'accompagne d'une **complexité** accrue où l'évaluation de la synergie nécessite de jongler entre plusieurs types de descripteurs et le réglage des poids λ peut s'avérer délicat. D'un point de vue opérationnel, le SCN doit gérer des structures de données plus variées, et la formation de **clusters** engage des entités aux caractéristiques hétérogènes.

Conséquences sur la Dynamique d'Auto-Organisation

Le **choix** d'une forme de modélisation détermine **directement** le **coût** et la **précision** de la synergie $S(i,j)$. Un calcul vectoriel est **rapide**, mais peut manquer de **transparence** ; un calcul symbolique ou graphique peut être sémantiquement **puissant** mais se révèle onéreux et plus fragile au bruit. Dans un système d'**auto-organisation**, où chaque lien $\omega_{i,j}$ est actualisé de manière itérative, la **complexité** de $S(i,j)$ devient cruciale si le réseau comporte un grand nombre d'entités. Par ailleurs, la **stabilité** de la représentation, l'adéquation entre la mesure de similarité et la réalité sémantique, ainsi que la résistance au bruit déterminent la **qualité** de l'émergence de clusters ou de schémas cognitifs.

Dans la pratique, un **DSL** performant peut s'appuyer sur des vecteurs ou embeddings lorsque les données sont massives et qu'on souhaite une **scalabilité** haute, quitte à sacrifier l'interprétabilité. Il peut exploiter des structures logiques pour bénéficier d'une **lisibilité** conceptuelle, en acceptant un calcul de S plus lourd. Il peut aussi combiner des descripteurs, par exemple sub-symboliques et

attributs symboliques, pour dégager des synergies multidimensionnelles, au prix d'un pipeline de calcul plus complexe.

Au sein du **Synergistic Connection Network**, cette souplesse dans la définition de $S(i,j)$ autorise l'**intégration** de données de sources multiples et la **découverte** de correspondances inattendues — moyennant une **ingénierie** soignée pour élaborer la modélisation la plus appropriée aux besoins applicatifs. Les chapitres ultérieurs (notamment ceux traitant de la **fusion multimodale** ou des **architectures SCN** avancées) approfondiront comment cette diversité se reflète dans les mécanismes concrets d'auto-organisation synergique.

3.1.2.3. Aborder des Concepts Avancés : Embeddings à Haut Niveau (Transformers), Représentations Symboliques Complexes, etc.

La **représentation** des entités dans un **DSL** (Deep Synergy Learning) n'est pas limitée à de simples vecteurs statiques ou à des graphes élémentaires où, pour décrire des objets ou des notions de plus en plus complexes, on peut recourir à des **embeddings contextuels** issus de **Transformers**, à des **ontologies** riches ou à des **structures** multi-niveaux voire fractales. Ces choix de modélisation, s'ils rendent la **synergie** plus riche et plus précise, posent également des défis en termes de **coût** algorithmique et de **gestion** de la dynamique. Les paragraphes qui suivent illustrent comment cette sophistication accroît le **pouvoir** du DSL au prix d'une **ingénierie** plus exigeante.

A. Embeddings à Haut Niveau (Transformers et Architectures Contextuelles)

Dans le cadre sub-symbolique, les entités $\mathcal{E}_i$ sont souvent associées à un **embedding** $\mathbf{x}_i \in \mathbb{R}^d$. Auparavant, des modèles statiques comme Word2Vec ou GloVe généraient un vecteur unique pour chaque mot. Aujourd'hui, l'essor des **Transformers** (BERT, GPT, T5, Vision Transformers, etc.) permet d'obtenir des **embeddings contextuels**, où la représentation varie selon l'environnement local (contexte lexical, patchs voisins dans une image, etc.). Mathématiquement, pour une entité $\mathcal{E}_i$, on définit une fonction

$$\mathbf{v}_i\colon \mathbf{C}_i \;\mapsto\; \mathbb{R}^d,$$

où $\mathbf{C}_i$ est le **contexte** (par exemple, les tokens entourant un mot, la position dans une séquence d'image, etc.). La **synergie** $S(i,j)$ peut alors prendre la forme d'une similarité vectorielle appliquée aux **embeddings** contextuels

$$S_{\text{transf}}(i,j) \;=\; \text{sim}\Big(\mathbf{v}_i(\mathbf{C}_i),\, \mathbf{v}_j\big(\mathbf{C}_j\big)\Big).$$

Cette contextualisation renforce la **précision** de la mesure de similarité où un même mot ou un même patch peut être représenté différemment suivant les indices locaux, évitant ainsi les ambiguïtés que l'on retrouve dans les embeddings purement statiques. En contrepartie, un tel modèle implique un **coût** de calcul plus important (multiple tête d'attention, layers de grande taille) et une **dynamique** plus complexe si le **contexte** $\mathbf{C}_i$ se modifie continuellement durant l'apprentissage. Le **DSL** doit donc gérer des synergies potentiellement instables si les embeddings contextuels ne sont pas figés.

B. Représentations Symboliques Complexes (Ontologies Riches, Logiques Hiérarchiques)

Le raffinement ne se limite pas au sub-symbolique. Dans un contexte **symbolique**, les entités peuvent relever de **logiques descriptives**, d'**ontologies** complexes (OWL, RDF) ou de **théories** plus abouties. L'idée est alors de définir

$$\mathrm{S}_{\text{symbolique}}(i,j) \;=\; \mathrm{compat}\big(\mathcal{O}_i,\; \mathcal{O}_j\big),$$

où $\mathcal{O}_i$ et $\mathcal{O}_j$ désignent des sous-ensembles de l'ontologie ou des blocs logiques associées aux entités $\mathcal{E}_i$ et $\mathcal{E}_j$. Le calcul de compat peut se baser sur le **recouvrement** de concepts, la **cohérence** (absence de contradiction), ou encore l'**intersection** sémantique. Cette approche a le **mérite** de l'**explicabilité** où l'on sait pourquoi deux entités sont jugées compatibles puisqu'elles partagent des axiomes, des types ou des relations. D'un point de vue opératoire, cependant, l'exécution peut nécessiter des **moteurs d'inférence** plus lourds, surtout si on emploie des logiques complexes ou des algorithmes de correspondance ontologique (ontology matching). Le DSL profite ici de la grande **lisibilité** des clusters émergents, mais la mise à jour $\omega_{i,j}$ s'expose à la rigidité du formalisme symbolique, moins tolérant au **bruit** et aux incertitudes.

C. Structures Multi-Niveau ou Fractales

Dans des applications **multi-échelle** (cf. chap. 6), chaque entité $\mathcal{E}_i$ peut se décliner en **sous-niveaux** ou exhiber des propriétés fractales. Par exemple, on peut envisager un document organisé en chapitres, sections, paragraphes, phrases, ou un graphe subdivisé hiérarchiquement. Le calcul de la synergie $\mathrm{S}(i,j)$ s'étend alors sur plusieurs **granularités**. Une formalisation schématique pourrait recourir à un opérateur d'intégration multi-résolution :

$$\mathrm{S}_{\text{multi}}(i,j) \;=\; \int_0^L \mathrm{Sim}\left(\mathcal{E}_i^{(\ell)},\, \mathcal{E}_j^{(\ell)}\right) \mathrm{d}\ell,$$

où ℓ indexe le niveau d'observation, entre 0 (global) et L (local très détaillé). Cette construction donne une **vision holistique** de la synergie, dans laquelle deux entités se révèlent proches si leurs structures s'apparentent à **tous** les niveaux ou si l'on détecte un fort **matching** dans plusieurs couches. Le **DSL** exploite alors une **auto-organisation** multiforme où si la synergie est forte à certains niveaux, les liens se renforcent, même si la vue globale n'est pas identique. Le prix à payer réside dans le **coût** de comparaison de multiples niveaux et dans la gestion de la cohérence à travers ces échelles.

D. Combiner Diverses Approches

Souvent, un système DSL se trouve confronté à des données hétérogènes mêlant représentations avancées sub-symboliques, structures symboliques complexes et granularités multiples. Une **approche hybride** consiste alors à définir

$$\mathrm{S}_{\text{avancée}}(i,j) \;=\; \lambda_1\, \mathrm{sim}_{\text{contextuel}}\big(\mathbf{v}_i, \mathbf{v}_j\big) \;+\; \lambda_2\, \mathrm{compat}_{\text{symbol}}\big(\mathcal{O}_i, \mathcal{O}_j\big) \;+\; \lambda_3\, \mathrm{SimMulti}\big(\mathcal{E}_i, \mathcal{E}_j\big),$$

avec $\lambda_1 + \lambda_2 + \lambda_3 = 1$. Cette formule intègre la **similarité contextuelle** (Transformers), la **compatibilité symbolique** (ontologies) et la **cohérence multi-niveau** (hiérarchie interne). Un tel design favorise une **syncrétisation** de l'information, permettant au **SCN** de découvrir des **clusters** sophistiqués englobant plusieurs perspectives. Cependant, la charge algorithmique et la complexité de déploiement augmentent d'autant. De plus, l'**auto-organisation** doit gérer la potentielle **instabilité** due à la variation simultanée de différentes composantes, particulièrement si l'on adapte en continu les embeddings et l'ontologie.

3.1.3. Structure du Chapitre

Dans ce chapitre, nous allons progressivement bâtir une **vision complète** de la représentation et de la modélisation des entités d'information dans le DSL. Nous avons déjà rappelé (sections 3.1.1 et 3.1.2) les **enjeux** et les **objectifs** ; désormais, nous allons suivre une **progression** logique afin de couvrir toutes les approches possibles (sub-symboliques, symboliques, hybrides, etc.) et les approfondir par des illustrations concrètes.

3.1.3.1. Vue d'Ensemble : Principes, Représentations Sub-Symboliques, Symboliques, Hybrides, Aspects Avancés et Études de Cas

Ce chapitre se propose d'explorer, de manière structurée, la diversité des **représentations** qui peuvent être adoptées pour décrire les entités $\mathcal{E}_i$ au sein d'un **Deep Synergy Learning (DSL)**, et d'en illustrer l'influence directe sur le **calcul de synergie** et la **dynamique adaptative**. Les principes généraux de la représentation, tels qu'ils s'appliquent au DSL, constituent un socle méthodologique qui sera mis à profit dans les chapitres ultérieurs portant sur l'**auto-organisation** (Chap. 4) et l'**architecture** du Synergistic Connection Network (Chap. 5).

L'idée centrale, rappelée dans ce chapitre, est que la **qualité** et la **nature** de la représentation conditionnent la **pertinence** de la fonction de synergie $S(i,j)$. Que les entités soient modélisées sous forme de **vecteurs** (apprentissage profond, embeddings neuronaux), de **structures symboliques** (règles logiques, graphes ontologiques) ou d'**architectures hybrides**, la façon dont on code l'information se répercute sur la manière dont le DSL identifie les relations entre entités, renforce ou affaiblit les pondérations $\omega_{i,j}$ et, en définitive, fait émerger des **clusters**. Les sections 3.2 à 3.7 proposent une trajectoire progressive où l'on part des **principes** généraux (3.2), avant d'examiner les cas **sub-symboliques** (3.3) et **symboliques** (3.4), puis d'étudier les **représentations hybrides** (3.5). Enfin, on introduit certains **aspects avancés** (3.6) et l'on conclut (3.7) sur la synthèse de ces différentes approches.

Section 3.2 : Principes généraux de la représentation dans le DSL

On y clarifie le rôle clé que joue la représentation des entités dans l'élaboration de la **synergie** $S(i,j)$. Après un rappel sur les exigences de robustesse, d'expressivité et de modularité, on souligne en quoi la **cohérence** entre la forme de représentation et la dynamique d'**auto-organisation** est décisive pour éviter des clusterisations parasitaires ou des oscillations improductives. Les arguments développés forment un **cadre théorique** dans lequel s'inscriront toutes les variantes ou extensions présentées par la suite.

Section 3.3 : Représentations sub-symboliques

Cette partie met l'accent sur les vecteurs et embeddings neuronaux, abordant notamment les mécanismes usuels de **similarité vectorielle** (distance euclidienne, cosinus, noyaux gaussiens). On met en évidence la facilité avec laquelle s'opère le **calcul** de synergie dans un espace vectoriel, contribuant ainsi à la **scalabilité** du DSL quand le nombre d'entités est important. Les vecteurs issus de modèles de deep learning (CNN, Transformers, autoencodeurs) illustrent l'**efficacité** et la **souplesse** des représentations sub-symboliques, tout en rappelant les limites liées à une **opacité** conceptuelle ou à une sensibilité aux biais de l'entraînement supervisé.

Section 3.4 : Représentations symboliques

Dans un second temps, on passe en revue les **entités** décrites au moyen de **structures logiques** ou d'**ontologies**. Cette approche présente l'intérêt d'une **interprétabilité** et d'une **inférence** plus directe où l'on peut déterminer si deux entités partagent des axiomes et si leurs règles s'avèrent compatibles ou contradictoires. Le **calcul** de $S(i,j)$ peut alors refléter la proportion d'axiomes communs, la distance entre classes ontologiques, ou la cohérence sémantique de leur union. Ce formalisme logique, souvent plus **rigide**, nécessite cependant des algorithmes de vérification plus lourds, et se montre peu tolérant au **bruit**. Les implications sur la dynamique du SCN sont discutées, soulignant la nécessité de contrôler la croissance de la complexité dans un réseau de grande taille.

Section 3.5 : Représentations hybrides

Une synthèse s'impose lorsque l'on veut combiner les atouts des vecteurs sub-symboliques (robustesse, généralisation) avec la lisibilité des structures symboliques. Les **modèles hybrides** associent ainsi, pour chaque entité $\mathcal{E}_i$, un embedding $\mathbf{v}_i$ et des règles $\mathcal{R}_i$. La synergie se conçoit alors comme une combinaison linéaire ou non linéaire de deux ou plusieurs mesures où l'une extrait la **similarité** vectorielle et l'autre évalue la **compatibilité** symbolique. Le paramètre λ dans

$$S_{\text{hybride}}(i,j) \;=\; \lambda\,\rho\big(\mathbf{v}_i,\mathbf{v}_j\big) \;+\; (1-\lambda)\,\text{Compat}\big(\mathcal{R}_i,\mathcal{R}_j\big)$$

illustre la souplesse de ce modèle, capable de s'adapter à des applications où la **sémantique** coexiste avec des propriétés statistiques ou perceptives. Les avantages d'une telle approche sont contrebalancés par la gestion plus complexe du calcul de **synergie**, qui doit agréger plusieurs points de vue.

Section 3.6 : Aspects avancés et études de cas

Cette section présente des **concepts** ou **développements** supplémentaires visant à accroître encore la diversité ou la sophistication des représentations. On évoque notamment les **synergies n-aires**, permettant de modéliser les interactions collectives entre plus de deux entités, ainsi que l'extension vers des **structures fractales** et **hypergraphes**. De plus, on y propose des **études de cas** concrets où des systèmes de vision sémantique enrichis par des ontologies, des applications multimodales combinant audio, vidéo et textes, ou encore la robotique cognitive où chaque entité correspond à un composant sensoriel ou à un module de planification, illustrent la diversité des approches. Ces exemples soulignent la flexibilité du DSL, mais mettent en relief la nécessité d'une **ingénierie** minutieuse pour garantir la **cohérence** et la **performance** du calcul de synergie.

Section 3.7 : Conclusion

Un dernier moment est consacré à **résumer** les points saillants de ce chapitre, tout en traçant des **passerelles** avec les chapitres suivants. On rappelle qu'une **représentation** de haute qualité (qu'elle soit vectorielle, symbolique ou hybride) conditionne la **dynamique** d'auto-organisation du DSL, à la fois dans la **descente d'énergie** qui stabilise les pondérations $\omega_{i,j}$ et dans la **formation** de clusters pertinents. Les choix retenus ici déterminent en grande partie l'efficacité, la robustesse et l'interprétabilité du **Synergistic Connection Network** dans les scénarios réels, un enjeu qu'illustreront les déploiements concrets abordés ultérieurement.

3.1.3.2. Renvoi aux Chapitres Ultérieurs

Le présent chapitre, consacré aux principes et aux méthodes de **représentation** des entités dans le **Deep Synergy Learning (DSL)**, s'inscrit dans un continuum dont les chapitres suivants approfondiront la dimension dynamique, architecturale et opérationnelle. La manière dont on encode une entité $\mathcal{E}_i$ et dont on calcule la synergie $\mathrm{S}(i,j)$ aura un impact direct sur les mécanismes décrits dans ces parties ultérieures. Le fait de bien saisir les choix de représentation vecteurs, règles symboliques ou structures hybrides, et de comprendre leurs avantages et inconvénients, constitue un prérequis essentiel à l'étude des dynamiques d'**auto-organisation**, de l'**architecture** du Synergistic Connection Network et de son déploiement multi-échelle. Les liens les plus directs avec les chapitres suivants s'établissent ainsi.

Dans le **Chapitre 4**, qui détaille la **dynamique d'auto-organisation**, les pondérations $\omega_{i,j}$ évoluent selon une équation itérative qui repose fortement sur la synergie $\mathrm{S}(i,j)$. Si cette synergie est déterminée par des embeddings sub-symboliques, le calcul de similarité sera rapide, favorisant l'implémentation de mécanismes tels que la descente d'énergie ou la stabilisation de clusters. Inversement, si la représentation est symbolique, la règle de mise à jour devra tenir compte de la compatibilité logique, ce qui influence la façon dont les clusters se forment ou se fragmentent. Les notions de robustesse de la représentation, de granularité ou de sensibilité au bruit abordées dans ce chapitre alimenteront la réflexion sur les oscillations, les convergences locales, ou l'émergence de configurations stables que le Chapitre 4 explicitera.

Le **Chapitre 5**, quant à lui, traite de l'**architecture générale** du Synergistic Connection Network. Là encore, les choix de représentation sont déterminants puisque la conception même du réseau – qu'il s'agisse de la gestion de larges embeddings vectoriels, de graphes conceptuels ou de règles logiques – impose des contraintes sur la structure interne du SCN, les protocoles de communication entre nœuds et la répartition des calculs. Ainsi, le fonctionnement d'un SCN manipulant des structures symboliques complexes doit intégrer un module de compatibilité ou d'inférence, tandis qu'un SCN vectoriel peut s'appuyer sur des algorithmes de similarité plus classiques. Ces aspects d'implémentation et de mise à l'échelle sont traités au Chapitre 5, mais prennent appui sur les fondements de la représentation établis ici.

Dans le **Chapitre 6**, consacré à l'**apprentissage synergique multi-échelle**, les représentations décrites dans le présent chapitre se verront transposées à des scénarios où la notion d'échelle ou de hiérarchie devient fondamentale. La façon dont une entité est encodée doit permettre de naviguer entre un niveau micro (par exemple, des attributs locaux d'un signal, des sous-parties d'un graphe) et un niveau macro (des clusters de haut niveau, des super-concepts). Les principes de modularité et de robustesse abordés ici guideront la construction d'architectures capables de traiter

simultanément ces différentes granularités, et influenceront le mode de calcul de la synergie lorsqu'il s'agit de regrouper des "super-entités" ou d'organiser plusieurs couches de représentation.

Le **Chapitre 7**, traitant des **algorithmes d'optimisation** et des **méthodes d'adaptation dynamique**, s'appuiera également sur les fondements énoncés dans le présent chapitre. Les stratégies de recuit simulé, de heuristiques génétiques ou d'optimisation distribuée dans un DSL se conçoivent plus facilement lorsque la fonction de synergie est aisément manipulable et la représentation stable. Les embeddings sub-symboliques se prêtent par exemple à des manœuvres de projection ou de sparsification utiles pour accélérer les routines d'optimisation, alors que des formalismes symboliques imposent des approches plus subtiles pour maintenir la cohérence logique ou résoudre les contradictions.

Le **Chapitre 8**, qui se focalise sur un **DSL multimodal** intégrant par exemple la vision, le langage et les signaux auditifs, prolonge la question de la représentation dans un cadre où chaque modalité est codée différemment où les CNN sont utilisés pour les images, les embeddings BERT pour les textes et les paramètres acoustiques pour l'audio. Les notions de ce chapitre portant sur la fusion sub-symbolique et symbolique, ou sur la construction de synergies hybrides, trouvent leur application concrète dans la conception d'un SCN capable d'identifier des associations ou des regroupements cohérents à travers plusieurs modalités. La qualité de la représentation qu'on aura choisie pour chaque type de contenu détermine alors la richesse ou la précision de la synergie inter-modale.

Enfin, dans d'autres chapitres ultérieurs (Chapitres 9, 10, 11, etc.) traitant de l'**évolution en temps réel**, des mécanismes de **feedback coopératif**, ou encore de la **robustesse** et de la **sécurité**, il apparaîtra qu'une bonne représentation n'est pas un simple détail où elle constitue un véritable moteur de résilience et de flexibilité. Les entités décrites de manière adéquate peuvent se réorganiser ou se mettre à jour plus aisément, et la logique de l'auto-organisation est plus lisible lorsque la synergie repose sur des signatures explicites ou bien définies. De surcroît, l'ajout d'un feedback ascendant ou descendant exige parfois la capacité de retourner à des représentations interprétables, qu'on ne peut dissocier des choix opérés dans le présent chapitre. Les mécanismes de détection d'anomalies ou de perturbations exploitent eux aussi la nature de l'entité où des vecteurs robustes au bruit ou des graphes logiques dotés de mécanismes de validation interne peuvent amortir les secousses et limiter les propagations d'erreurs.

Le **Chapitre 3** jette les bases d'un ensemble de choix et de contraintes qui façonnent la dynamique, l'architecture et la pérennité du Deep Synergy Learning. Les représentations sub-symboliques, symboliques ou hybrides et leurs déclinaisons avancées constitueront un fil rouge qui s'étire jusqu'aux démonstrations et applications pratiques des derniers chapitres, liant la théorie de la synergie et l'implémentation à grande échelle d'un SCN apte à manipuler des données hétérogènes, évolutives et à multiples échelles de granularité.

3.2. Principes Généraux de la Représentation dans le DSL

L'étape de **représentation** d'une entité $\mathcal{E}_i$ est un maillon essentiel de la chaîne d'apprentissage dans le **DSL** (Deep Synergy Learning). De cette modélisation initiale dépend la qualité de la fonction de synergie $S(i,j)$ — et, par conséquent, la pertinence des clusters et de la dynamique d'auto-organisation $\omega_{i,j}$. Avant d'explorer les différents types de représentation (sections 3.3, 3.4, 3.5) et leurs implications (section 3.6), nous allons poser quelques **principes généraux** essentiels à comprendre.

3.2.1. Rôle Fondamental de la Représentation

À ce stade, nous avons déjà souligné (chap. 3.1) que la **description** de chaque entité influe directement sur la manière dont on évalue la synergie. Entrons maintenant dans le détail de ce rôle fondamental, en explicitant comment la représentation oriente tout le processus d'apprentissage.

3.2.1.1. Chaque entité $\mathcal{E}_i$ doit être "capturable" par une structure (vecteur, ensemble de règles…)

Afin de garantir la cohérence globale du **Deep Synergy Learning (DSL)** et de son mécanisme d'**auto-organisation**, il est indispensable que chaque entité $\mathcal{E}_i$ puisse être décrite au moyen d'une **structure** qui reflète ses attributs pertinents et permette de calculer la **synergie** $\mathrm{S}(i,j)$. L'enjeu est de choisir ou de concevoir une représentation à la fois robuste et suffisamment expressive pour mettre en évidence les similitudes ou les compatibilités recherchées au sein du **Synergistic Connection Network (SCN)**. Plusieurs facteurs interviennent alors dans la définition de ce "format" :

Dans un premier temps, il s'agit de **saisir** les attributs importants de l'entité $\mathcal{E}_i$. Lorsque l'objet à décrire est une image, se limiter à une statistique globale comme la moyenne de luminosité ne suffit pas à capturer des singularités plus complexes (formes, couleurs, textures). Au contraire, un **embedding** généré par un réseau de neurones profonds (CNN, Vision Transformer, etc.) peut dégager des caractéristiques discriminantes indiquant la présence de motifs ou d'objets précis. Un raisonnement analogue s'applique aux textes où un simple bag-of-words se révèle souvent insuffisant pour rendre compte des relations contextuelles et des sémantiques sous-jacentes, tandis qu'un **embedding contextualisé** (type BERT ou GPT) prend en compte l'environnement lexical et la structure syntaxique. Dans le cas de données symboliques (règles, axiomes), il faut veiller à recourir à un formalisme (logique, ontologie, graphe conceptuel) permettant de décrire toutes les relations nécessaires, sans perte d'expressivité.

Il est également primordial de **faire correspondre** la forme de la représentation au **mode de calcul** de la fonction $\mathrm{S}(i,j)$. Si la synergie exploite une mesure de similarité vectorielle, par exemple la distance cosinus ou un noyau RBF, alors chaque entité se voit associée à un vecteur $\mathbf{x}_i$ dans un espace de dimension raisonnable. À l'inverse, si l'on s'appuie sur un critère de **compatibilité logique** (comme la proportion d'axiomes communs ou la cohérence de deux ensembles de règles), il devient impératif d'exprimer $\mathcal{E}_i$ sous une forme symbolique adéquate (règles logiques, graphes

de connaissances, etc.), afin que les algorithmes de correspondance ou d'inférence puissent opérer. Ainsi, il doit y avoir une **cohérence** entre le type de structure choisi et l'**opérateur** de similarité ou de compatibilité mis en œuvre dans S.

Cette logique doit être prolongée dans les contextes **multimodaux** ou hétérogènes, où les entités se répartissent entre images, textes, flux audio, signaux bruts, formules logiques ou tout autre format. Il est alors vain de tenter d'imposer un unique schéma identique pour toutes les entités. Une image pourra s'encoder sous forme d'un embedding convolutif $\mathbf{x}_i$, tandis qu'une base de règles symboliques restera décrite par un ensemble $\mathcal{R}_i$. Toutefois, le **DSL** exige qu'il existe un moyen, le cas échéant, de calculer $\mathrm{S}(i,j)$ entre entités de types différents. Cette nécessité peut conduire à prévoir un module de "passerelle" ou de traduction conceptuelle — par exemple, via des métadonnées partagées ou un espace latent commun — garantissant la **compatibilité** des calculs de synergie.

Enfin, il est important de **préserver l'évolutivité** de la représentation. Dans un apprentissage continu (cf. chap. 9), l'embedding d'une entité peut être affiné ou ses règles symboliques peuvent être mises à jour à mesure que de nouvelles informations parviennent au réseau. La structure choisie ne doit pas **verrouiller** l'entité dans un format immuable, sous peine de contraindre la plasticité du système. Au contraire, la capacité à réviser ou à enrichir la description de $\mathcal{E}_i$ en cours de route constitue un point essentiel pour l'**adaptation** et la **réorganisation** dynamique.

Ainsi, l'exigence centrale peut se résumer en ces termes où toute entité $\mathcal{E}_i$ doit être munie d'une **description** ou d'un **formalisme** qui capture les attributs ou les traits indispensables à la distinction et à la mise en relation, qu'il s'agisse d'images, de texte, de règles ou de graphes. Elle doit également s'aligner sur la manière de **définir** ou de **calculer** la synergie $\mathrm{S}(i,j)$, gérer la potentielle hétérogénéité des types de données dans un même **SCN**, et permettre une **évolution** ou un **raffinement** si la situation ou les connaissances changent.

Lorsque ce cadre est respecté, la dynamique **auto-organisée** s'appuyant sur $\omega_{i,j}$ peut alors se déployer sans **biais** majeur. À l'inverse, si la représentation d'une partie des entités échoue à décrire les informations clés — ou se trouve en complet décalage par rapport au schéma de similarité attendu —, la **synergie** en sera faussée et la **structuration** en clusters ou en liens cohérents ne pourra émerger convenablement.

Comme on le verra dans la section suivante (3.2.1.2), le calcul de $\mathrm{S}(i,j)$ lui-même s'appuie directement sur ces considérations, établissant un continuum entre la forme de l'entité, la fonction de similarité et la mise à jour adaptative du réseau ω. De plus, la section 3.2.2 présentera divers critères (pertinence, complexité, robustesse) qui aident à **opérer** les bons choix de format selon le domaine d'application et les contraintes logicielles.

3.2.1.2. La Fonction de Synergie $S(i,j)$ Dépend Directement de ces Représentations

Le **Deep Synergy Learning (DSL)** se caractérise par l'utilisation d'une **fonction de synergie** $\mathrm{S}(i,j)$ permettant d'évaluer à quel point deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ présentent une **affinité** ou une **compatibilité** mutuelle. Avant même de décider si l'on doit renforcer ou affaiblir la pondération $\omega_{i,j}$, le réseau s'appuie sur les **descripteurs** qui modélisent chacune de ces entités. Ainsi, toute

variation dans la manière de représenter $\mathcal{E}_i$ (ou $\mathcal{E}_j$) engendre un **changement** potentiel dans la valeur $\mathrm{S}(i,j)$.

Formellement, on peut écrire :

$$S(i,j) = f\big(\mathbf{r}(i), \mathbf{r}(j)\big),$$

où $\mathbf{r}(i)$ désigne la représentation de l'entité $\mathcal{E}_i$ (par exemple, un vecteur, un ensemble de règles symboliques, ou un graphe), et la fonction f dépend du **format** de ces représentations (similarité vectorielle, mesure de compatibilité logique, distance structurelle, etc.). Ce **schéma** induit que :

1. **Pour des entités vectorielles**

   Si $\mathbf{r}(i) = \mathbf{x}_i$ et $\mathbf{r}(j) = \mathbf{x}_j$ sont des vecteurs en $\mathbb{R}^d$, le calcul de $\mathrm{S}(i,j)$ repose souvent sur une **distance** (euclidienne, manhattan) ou une **similarité** (cosinus, gaussienne).

   $$\mathrm{S}_{\mathrm{vect}}(i,j) = \exp\big(-\alpha \parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2\big) \quad \text{ou} \quad \mathrm{S}_{\mathrm{vect}}(i,j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\parallel \mathbf{x}_i \parallel \parallel \mathbf{x}_j \parallel}.$$

   Le **choix** et la **qualité** de l'embedding $\mathbf{x}_i$ influent directement sur le résultat où modifier la dimension du vecteur ou la façon dont on l'a appris, qu'il s'agisse d'un modèle de deep learning, d'une PCA ou d'un autoencodeur, se traduit par une **variation** dans les valeurs de S.

2. **Pour des entités symboliques**

   Si $\mathbf{r}(i) = R_i$ et $\mathbf{r}(j) = R_j$ représentent des **blocs** (ou des ensembles) de règles, d'axiomes ou de concepts, $\mathrm{S}(i,j)$ peut alors refléter le **taux de recouvrement** (intersection d'axiomes), la **cohérence** s'il s'agit de les unir, ou toute autre **mesure** de compatibilité symbolique.

   $$\mathrm{S}_{\mathrm{sym}}(i,j) = \mathrm{Compat}\big(R_i,\ R_j\big).$$

   Dans ce cas, l'ajout ou la suppression d'une règle dans R_i affecte le **score** final où cela peut drastiquement modifier la structure de l'auto-organisation, rendant deux entités auparavant jugées proches soudain incompatibles, ou inversement.

3. **Représentations hybrides**

   Souvent, le DSL manipule des **descripteurs composites** alliant une partie vectorielle (embedding) et une partie symbolique (règles, concepts). On peut alors définir

   $$\mathrm{S}_{\mathrm{hybride}}(i,j) = \alpha\, S_{\mathrm{vect}}\big(\mathbf{x}_i, \mathbf{x}_j\big) + (1-\alpha)\, S_{\mathrm{sym}}\big(R_i, R_j\big),$$

   où $\alpha \in [0,1]$ détermine l'importance relative du versant sub-symbolique ou symbolique. Toute **évolution** dans l'une ou l'autre des composantes ($\mathbf{x}_i$ ou R_i) impacte la **valeur** de $\mathrm{S}(i,j)$.

4. **Cas n-aire ou hypergraphique**

Un **DSL** avancé peut, par ailleurs, recourir à une synergie $S(i_1, \ldots, i_k)$ dépendant simultanément de la représentation de k entités. Là encore, chaque entité $\mathbf{r}(i_m)$ contribue à la **coopération** ou à la **conflictualité** du groupe.

On peut décrire la représentation par une application $g: \{\mathcal{E}_i\} \to \mathcal{X}$, où $\mathcal{X}$ est l'espace ou la famille d'espaces dans lequel vivent les **descripteurs** (vecteurs, ensembles d'axiomes, graphes, etc.). La fonction S devient alors

$$S: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R},$$

ce qui formalise la dépendance : $S\left(g(\mathcal{E}_i),\ g(\mathcal{E}_j)\right)$. Toute **modification** dans le contenu ou la forme de $g(\mathcal{E}_i)$ (c'est-à-dire la représentation de $\mathcal{E}_i$) répercute un **changement** dans $S(i,j)$. Ce phénomène se propage à la mise à jour des pondérations $\omega_{i,j}$ (voir Chap. 4), car la règle de type

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\left[S(i,j) - \tau\,\omega_{i,j}(t)\right]$$

se fonde sur $S(i,j)$ à chaque itération.

Dans l'esprit de l'**auto-organisation**, la structure du **SCN** (existence de clusters, de liens forts ou faibles) résulte alors de la manière dont S rend compte de la **proximité** ou de la **compatibilité** entre entités. Une **représentation** trop sommaire réduit la capacité de S à distinguer des différences subtiles, tandis qu'une représentation plus riche ou évolutive affine la détection des schémas latents et favorise la formation de clusters pertinents.

3.2.2. Critères de Choix

Après avoir mis en évidence le rôle fondamental de la représentation (section 3.2.1), il convient d'examiner **quels critères** président à la sélection d'une forme de description pour les entités $\mathcal{E}_i$. Ces critères incluent la **pertinence** de la représentation (dimension, nature sub-symbolique vs. symbolique, etc.), la **complexité** de calcul que cela implique, et la **capacité** d'évoluer ou de s'adapter au fil du temps. Nous allons passer en revue chacun de ces aspects.

3.2.2.1. Pertinence (Faut-il un Embedding Large ? un Formalisme Logique ?)

Le **premier** critère d'évaluation d'une représentation, dans le cadre du **Deep Synergy Learning (DSL)**, consiste à s'interroger sur sa **pertinence** par rapport aux entités visées et aux objectifs de la **synergie**. Un format de description n'est en effet pas intrinsèquement bon ou mauvais ; il faut l'analyser à l'aune du **contenu** réel de chaque entité, du **type** de synergie que l'on souhaite dévoiler, du **niveau** de granularité requis et de la **finalité** (raisonnement symbolique, clustering statistique, etc.). En pratique, plusieurs éléments entrent en jeu :

Alignement sur le contenu réel de l'entité

La **nature** même de l'entité $\mathcal{E}_i$ dicte souvent la **forme** de représentation la plus adéquate. Lorsqu'il s'agit d'une **image**, un vecteur d'embedding dérivé d'un **réseau de neurones** (CNN, ViT) capture

des attributs visuels déterminants (formes, textures, objets), alors qu'un simple histogramme d'intensité serait trop restreint pour distinguer des classes complexes. De même, pour un **texte**, un bag-of-words statique efface les relations contextuelles et ne parvient pas à rendre la sémantique profonde, tandis qu'un **embedding contextualisé** (BERT, GPT) intègre précisément ces nuances. Dans tous les cas, la représentation choisie doit couvrir les aspects saillants ou discriminants de l'entité, faute de quoi la **synergie** $\mathrm{S}(i,j)$ perd en expressivité et l'auto-organisation en souffre.

Adéquation au type de synergie souhaité

Si la fonction $\mathrm{S}(i,j)$ est conçue pour évaluer une **compatibilité logique** (absence de contradiction, mise en cohérence d'axiomes, etc.), un **formalisme symbolique** (règles, ontologie) s'impose. On y gagne en **interprétabilité** et en capacité de raisonnement explicite, par exemple pour vérifier si deux ensembles de règles peuvent coexister. À l'inverse, si l'on cherche à détecter des **similitudes** majoritairement basées sur des correspondances statistiques (images, textes bruités, signaux), un **embedding sub-symbolique** est plus approprié. Le calcul de $\mathrm{S}(i,j)$ se fera alors via une mesure de distance ou de similarité vectorielle. Ainsi, la **logique de la synergie** (symbolique vs sub-symbolique) oriente la structure de la représentation, et réciproquement.

Niveau de granularité

Au sein d'une **représentation vectorielle**, la question de la **dimension** (d) se pose où un embedding de grande taille, par exemple 768 dimensions, peut mieux cerner des subtilités mais risque de générer de la redondance ou d'exacerber la sensibilité à la "malédiction de la dimension". Une dimension plus réduite (type autoencodeur) rend la représentation plus stable et moins coûteuse, au risque de perdre des informations fines. Cette problématique se double d'une éventuelle **hiérarchisation** où l'on peut choisir un embedding global, avec une seule dimension du type "chat" ou "voiture", ou opter pour des attributs plus nuancés. Un bon compromis s'efforce de maintenir la diversité descriptive tout en évitant la surcharge dimensionnelle qui nuirait au calcul de $\| \mathbf{x}_i - \mathbf{x}_j \|$.

Possibilité d'exploitation ultérieure

Selon la **finalité** visée, on peut privilégier un certain **formalisme**. Les représentations logiques offrent par exemple la **facilité** de mise en œuvre d'un raisonnement déductif, d'une détection de contradictions ou d'un enrichissement par de nouveaux axiomes. Un vecteur, en revanche, s'intègre mieux dans des **pipelines** de clustering ou de classification statistique, dispose d'un large écosystème d'algorithmes et se manipule aisément pour des calculs de distances ou de produits scalaires. La **composante** "exploitation future" est donc cruciale où il s'agit de déterminer si l'objectif est de raisonner, annoter, expliquer ou si l'on privilégie avant tout une similarité fluide et scalable.

Nature des données

Au-delà du choix conceptuel, la **nature** des données elles-mêmes oriente fortement la **représentation**. Un flux **audiovisuel** ou **sensoriel** continu se prête naturellement à une description sub-symbolique (embedding, vecteur de caractéristiques), tandis qu'un ensemble de **règles expertes** s'incarne naturellement dans un formalisme d'ontologie ou de logique descriptive. Même dans des scénarios multimodaux, on veillera à ne pas forcer un encodage unique pour toutes les sources où il est préférable de conserver la spécificité des formats, au besoin en utilisant des modules passerelles (cf. Chap. 3.5 sur les représentations hybrides).

Mesure indirecte : écart entre S et $S^\star$

La **pertinence** se définit comme la capacité de la représentation (et de la fonction S) à **approximer** une notion "vraie" de similarité $S^\star$. Il s'agit de minimiser la différence $|S^\star(i,j) - S(i,j)|$ pour les couples (i,j) qu'on juge objectivement proches. Bien sûr, on ne dispose pas de $S^\star$ de manière explicite, mais cette considération sert de **guide**. Plus la représentation reflète les attributs réellement discriminants, plus $S(i,j)$ concorde avec le jugement "naturel" ou "souhaité". À défaut, si la représentation omet ou brouille les caractéristiques clés, la fonction de synergie s'en voit faussée.

3.2.2.2. Complexité Computationnelle (Dimension du Vecteur, Structure Symbolique)

Au-delà du simple **critère de pertinence**, il est primordial de considérer le **coût** (en temps et en ressources) associé au **calcul** de la synergie $S(i,j)$. Chaque forme de représentation (embeddings vectoriels, logiques symboliques, etc.) induit une **charge** algorithmique différente, qui peut s'avérer déterminante pour l'évolutivité et la stabilité du **Deep Synergy Learning (DSL)**. Lorsque le nombre d'entités n et la taille (ou la complexité) de la représentation augmentent, une mauvaise anticipation de cette complexité peut rendre la mise à jour des pondérations $\omega_{i,j}$ irréalisable dans un temps raisonnable, voire conduire à des instabilités de la dynamique auto-organisée.

1. Dimension du vecteur et coût en $O(n^2\, d)$

Lorsqu'on adopte des **représentations sub-symboliques** de type vectoriel, chaque entité $\mathcal{E}_i$ est décrite par un vecteur $\mathbf{x}_i \in \mathbb{R}^d$. La plupart des calculs de synergie (ex. cosinus, gaussienne) entre deux vecteurs se font en $O(d)$, ce qui reste modeste si d est de taille modérée (e.g., 100–1000). Néanmoins, si le réseau manipule n entités, la comparaison **exhaustive** de toutes les paires (i,j) se chiffre en $O(n^2\, d)$. Pour un large n (plusieurs millions) et un vecteur d élevé (512, 1024, etc.), ce volume de calcul peut devenir **prohibitif** :

$$S_{\text{vect}}(i,j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\| \mathbf{x}_i \| \, \| \mathbf{x}_j \|}, \quad \text{coût en } O(d) \text{ par comparaison}, \quad O(n^2\, d) \text{ au total.}$$

Stratégies :

- **Approximate Nearest Neighbor** (ANN), qui évite de comparer toutes les paires en se concentrant sur les plus proches.
- **Projection de dimension** (PCA, autoencodeurs) pour réduire d.
- **Filtrage** ou **sparsification** où on n'évalue la synergie que pour des couples (i, j) pré-selectionnés (distants exclus).

2. Structure symbolique et complexité du matching

Pour les entités **symboliques** (règles logiques, axiomes, ontologies), la fonction $\mathrm{S}(i, j)$ peut exiger des algorithmes de correspondance ou de vérification de cohérence, voire d'**unification** (en logique). De tels problèmes tombent parfois dans des classes de complexité élevées (NP-Complets, voire plus) :

$$\mathrm{Compat}(R_i,\ R_j) \quad \rightarrow \quad \text{matching ou unification } \sim \mathrm{O}\left(\exp\left(|R_i| + |R_j|\right)\right) \text{ (pire cas).}$$

Dans le cas de graphes symboliques ou d'ontologies volumineuses, la vérification d'isomorphisme, de sous-isomorphisme ou l'alignement d'ontologies peut aussi s'avérer $\mathrm{O}(\exp(\cdot))$ en théorie. Cela se répercute sur le **calcul total** des synergies, qui peut alors freiner l'évolutivité du DSL :

Stratégies :

- **Restreindre** la logique (Horn clauses, sous-langage OWL décidable) pour limiter la complexité.
- **Indexation** partielle ou agrégation de règles dans des structures plus compactes.
- **Heuristiques** pour unifier ou comparer les entités sans tout explorer (matching local, découpage).

3. Synergie n-aire et explosion combinatoire

Une extension de la synergie binaire $\mathrm{S}(i, j)$ vers une synergie n-aire $\mathrm{S}(i_1, \ldots, i_k)$ (voir chap. 12) fait passer le nombre de sous-ensembles à comparer de $\mathrm{O}(n^2)$ vers $\mathrm{O}(n^k)$. Quand $k > 2$, l'on risque une **explosion** combinatoire des évaluations, aggravée si chaque entité $\mathcal{E}_i$ est décrite par un graphe symbolique volumineux. Cet **effet multiplicatif** rend vital l'introduction de mécanismes filtrant quels groupes d'entités vérifier, ou restreignant k à 2 ou 3.

4. Trade-off entre précision et coût

Plus la représentation est **détaillée** (forte dimension vectorielle, ensemble symbolique volumineux, structure fractale), plus la fonction S risque d'être précise et riche, mais au **prix** d'un coût de calcul parfois considérable. En notation abstraite, si $\mathrm{C}(\mathbf{r})$ désigne la **complexité** liée à la représentation $\mathbf{r}$, alors la comparaison de $\mathbf{r}(i)$ et $\mathbf{r}(j)$ s'évalue en $\mathrm{O}(\mathrm{C}(\mathbf{r}))$. Multipliée par $\mathrm{O}(n^2)$ entités, on obtient une charge globale $\mathrm{O}(n^2 \times \mathrm{C}(\mathbf{r}))$. L'**ingénierie** du DSL doit donc trouver un **compromis** :

$$\min(\mathrm{C}_{\mathrm{calcul}}(\mathrm{S}) \ + \ \alpha\, \mathrm{Err}),$$

où C_{calcul} mesure le coût de l'évaluation de S et Err l'**erreur** ou la **perte de détails** associée à une représentation plus légère (dimension réduite, règles simplifiées, etc.).

5. Distribution et stockage

Enfin, dans un **SCN** distribué (Chap. 5.7), on doit aussi considérer la **taille** du stockage nécessaire et le **trafic** induit par la récupération des descripteurs $\mathbf{r}(i)$. Conserver un embedding de dimension 512 pour 10^6 entités représente déjà $O(512 \times 10^6)$ flottants, sans parler d'un matching symbolique éventuel sur des milliers de règles. On peut répartir ces données sur plusieurs nœuds, mais il demeure crucial de **minimiser** la quantité d'informations échangées lors du calcul de synergie.

3.2.2.3. Évolutivité (Peut-on Ajouter des Attributs ?)

Un troisième critère majeur dans le choix de la **représentation** pour un système de **Deep Synergy Learning (DSL)** réside dans la **capacité** à faire **évoluer** cette description au fil du temps ou à y **intégrer** de nouveaux attributs. Dans des contextes d'**apprentissage continu**, de **données changeantes** ou de **mise à jour** de règles symboliques, il est fréquent que la représentation d'une entité $\mathcal{E}_i$ doive être modifiée, soit pour ajouter un attribut (une nouvelle coordonnée vectorielle, un nouveau champ sémantique), soit pour réviser les informations déjà présentes. Du point de vue formel, cette **évolutivité** implique la possibilité de mettre à jour la fonction $\mathbf{r}(i)$ en cours de route, sans pour autant briser la dynamique d'auto-organisation basée sur la synergie $S(i,j)$. Différents aspects viennent éclairer ce besoin :

A. Évolutivité dans l'Espace Vectoriel

Lorsqu'on recourt à une **représentation sub-symbolique** de type vectoriel, chaque entité $\mathcal{E}_i$ est associée à un vecteur $\mathbf{x}_i \in \mathbb{R}^d$. Il peut survenir qu'une nouvelle **dimension** doive être incorporée, reflétant un attribut nouvellement découvert. Dans ce cas, on passe d'un embedding $\mathbf{x}_i(t) \in \mathbb{R}^d$ à un embedding $\mathbf{x}_i(t+1) \in \mathbb{R}^{d+1}$. Toutefois, l'ajout d'une coordonnée aux vecteurs de certaines entités, et pas à d'autres, risque de **déstabiliser** la fonction de similarité :

$$\| \mathbf{x}_i(t+1) - \mathbf{x}_j(t+1) \| \quad \text{devient ambigu si la dimension change.}$$

La solution la plus straightforward consiste à **initialiser** la nouvelle dimension (attribuée) à une valeur neutre (0, par exemple) pour toutes les entités concernées. Ainsi, l'ensemble des vecteurs se retrouvent à la même dimension $d+1$. Par la suite, on laisse l'**auto-organisation** adapter ces composantes — par apprentissage supervisé ou non, ou par un mécanisme de recalibration — de sorte que la synergie puisse à nouveau être calculée de façon cohérente. Le point crucial est de **maintenir** un alignement des vecteurs où si certains sont encore en $\mathbb{R}^d$ et d'autres en $\mathbb{R}^{d+1}$, la dynamique de mise à jour $\omega_{i,j}$ risque de se muer en une source de divergences.

B. Évolutivité dans une Structure Symbolique ou une Ontologie

Dans le cas d'une représentation **symbolique**, où l'entité $\mathcal{E}_i$ se décrit par un **ensemble** de règles, d'axiomes ou de concepts R_i, on peut être amené à **ajouter** de nouvelles règles ΔR pour refléter un enrichissement de la connaissance. Il se produit alors une évolution :

$$R_i(t+1) \;=\; R_i(t) \;\cup\; \Delta R,$$

qu'il s'agisse de lois supplémentaires (ex. "si fièvre et éruption cutanée alors maladie Z") ou d'une mise à jour dans une ontologie (nouveau concept, nouvelle relation). Cette **extension** de la description modifie la fonction de compatibilité symbolique. Deux entités antérieurement divergentes peuvent désormais partager un axiome, accroissant $\mathrm{S}(i,j)$, ou au contraire révéler une contradiction.

Le **défi** computationnel tient à la nécessité de **recalculer** (ou de mettre à jour) $\mathrm{S}_{\mathrm{sym}}$ pour toutes les paires (i,j). Si ΔR est modeste, on peut envisager un algorithme incrémental, au lieu d'une recombinaison exhaustive. Le **DSL** doit s'assurer que la complexité — déjà évoquée en section 3.2.2.2 pour les représentations symboliques — reste gérable, sans bloquer la dynamique globale.

C. Fusion ou Scission d'Entités

Un cas plus particulier concerne la **fusion** de deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ en une entité unique $\mathcal{E}_k$, ou la **scission** inverse. Du point de vue de la représentation :

1. La **fusion** consiste à **combiner** $\mathbf{r}(i)$ et $\mathbf{r}(j)$ en une unique description $\mathbf{r}(k)$. Dans un cadre vectoriel, on peut, par exemple, prendre un mélange $\alpha\mathbf{x}_i + (1-\alpha)\mathbf{x}_j$. Pour des règles symboliques, on effectue l'union $R_i \cup R_j$.

2. La **scission** divise $\mathbf{r}(k)$ en deux descriptions $\mathbf{r}(k_a)$ et $\mathbf{r}(k_b)$ où un vecteur peut être partitionné ou un ensemble de règles réparti en deux sous-ensembles.

Cette adaptation modifie la **composition** même du **SCN** (des entités disparaissent, d'autres apparaissent), et entraîne une **réévaluation** de la synergie vis-à-vis de toutes les entités. Le système doit alors gérer ce chamboulement structurel (réassignation de liens $\omega_{i,j}$, recalcul partiel de S), et continuer son auto-organisation sans repartir entièrement de zéro.

D. Mise à Jour Incrémentale dans le Temps

De façon générale, un **DSL** opère souvent en **flux** (voir chap. 9), où de nouvelles données ou observations parviennent en continu. La représentation de l'entité $\mathcal{E}_i$ peut alors se réviser itérativement :

$$\mathbf{r}(i,t+1) \;=\; \mathbf{r}(i,t) \;+\; \Delta\mathbf{r}_i(t),$$

et chaque entité incorpore ainsi de l'information fraîche (données supplémentaires, règles découvertes, nouveaux attributs). La fonction $\mathrm{S}(i,j)$ se met à jour en conséquence, puisqu'un changement dans $\mathbf{r}(i)$ implique forcément un recalcul potentiel de la synergie. La **plasticité** du

SCN (pondérations $\omega_{i,j}$) réagit immédiatement à ces modifications où, par exemple, si une amélioration de l'embedding rapproche $\mathbf{r}(i)$ de $\mathbf{r}(j)$, la pondération $\omega_{i,j}$ pourra se renforcer.

E. Considérations de Cohérence

Il importe de veiller à la **cohérence** du système lorsqu'on introduit ou modifie la représentation. Si l'on ajoute une composante dans un vecteur, il faut que toutes les entités possèdent cette nouvelle dimension, au moins à titre de placeholder (valeur 0). De même, dans un formalisme logique, un concept n'a de sens que si toutes les entités susceptibles de l'utiliser le reconnaissent en tant qu'attribut ou axiome possible. Faute de synchronisation dans l'évolution du format, la fonction $\mathrm{S}(i,j)$ peut devenir impraticable pour certaines paires.

Un protocole **échelonné** ou **progressif** peut être imaginé où l'on introduit la nouvelle dimension ou règle dans un sous-ensemble d'entités tout en garantissant un mécanisme de fallback, par exemple une coordonnée 0 ou un axiome neutre, afin d'éviter la perte de comparabilité. L'idée demeure d'éviter un changement brutal qui perturbe la dynamique auto-organisée ou invalide des synergies calculées précédemment.

3.2.3. Impact sur la Synergie et la Dynamique

Après avoir discuté des **critères de choix** (pertinence, complexité, évolutivité) dans la section 3.2.2, il est crucial de saisir **comment** la représentation affecte non seulement la **valeur** de la synergie $S(i,j)$, mais aussi la **dynamique** globale du DSL (mise à jour des pondérations $\omega_{i,j}$, formation de clusters, etc.). Le **rôle** de la représentation est ici déterminant où, en étant plus ou moins proche entre deux entités, on module directement la **probabilité** qu'elles se renforcent mutuellement dans le réseau.

3.2.3.1. Représentation "proche" $\Longrightarrow$ Synergie Potentiellement Élevée, $\omega_{i,j}$ Renforcée

L'un des traits marquants du **Deep Synergy Learning (DSL)** réside dans la façon dont la **proximité** ou la **similarité** entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ (telle qu'elle est traduite par leur **représentation**) induit mécaniquement une **augmentation** de la pondération $\omega_{i,j}$. Cet effet peut se voir directement dans la règle de mise à jour inspirée d'une descente d'énergie, où la **synergie** $\mathrm{S}(i,j)$ agit comme un signal de **renforcement**. Plus la représentation montre une concordance marquée, plus la **liaison** entre ces entités se consolide, favorisant l'émergence de **clusters** cohérents.

Exemple dans le cadre sub-symbolique

Lorsque la représentation de chaque entité $\mathcal{E}_i$ est un **vecteur** $\mathbf{x}_i \in \mathbb{R}^d$, la synergie $\mathrm{S}(i,j)$ peut se définir via une **mesure** de similarité vectorielle (cosinus, distance euclidienne inversée, etc.). Un choix courant est la **similarité cosinus** :

$$\mathrm{S}(i,j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\| \mathbf{x}_i \| \, \| \mathbf{x}_j \|}, \quad \text{valeur dans } [-1,\ 1].$$

Si $\mathbf{x}_i$ et $\mathbf{x}_j$ sont presque colinéaires, c'est-à-dire $\mathbf{x}_i \cdot \mathbf{x}_j \approx \| \mathbf{x}_i \| \, \| \mathbf{x}_j \|$, alors la valeur de $\mathrm{S}(i,j)$ s'approche de 1, traduisant une **forte similarité**. Dans la mise à jour adaptative :

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) \ + \ \eta \left[\mathrm{S}(i,j) - \tau \, \omega_{i,j}(t) \right],$$

la présence d'un $\mathrm{S}(i,j)$ élevé engendre un **terme positif** $\eta \, \mathrm{S}(i,j)$, dépassant la "pénalisation" $\eta \, \tau \, \omega_{i,j}(t)$ si la synergie est assez grande. La **liaison** $\omega_{i,j}$ se voit donc **renforcée** et tend vers un équilibre local

$$\omega_{i,j}^{*} \approx \frac{\mathrm{S}(i,j)}{\tau}.$$

Deux entités largement **proches** (au sens vectoriel) se verront ainsi attribuer une pondération stable et élevée. D'un point de vue **clustering**, elles se retrouveront **attirées** l'une vers l'autre, formant ou rejoignant un **sous-groupe** à plus large échelle dans le Synergistic Connection Network.

Exemple dans le cadre symbolique

La même idée vaut si les entités $\mathcal{E}_i$ et $\mathcal{E}_j$ sont décrites par des **blocs** de règles logiques ou des **concepts** dans une ontologie (R_i et R_j). La fonction de synergie $\mathrm{S}_{\mathrm{sym}}$ évalue la **compatibilité** ou la **cohérence** entre ces deux ensembles :

$$\mathrm{S}_{\mathrm{sym}}(i,j) = \mathrm{Compat}\big(R_i,\ R_j\big),$$

qui peut mesurer la proportion d'axiomes ou de règles partagées, ou l'absence de contradictions sémantiques. Plus cette **intersection** conceptuelle est forte, plus la synergie atteint des valeurs élevées, incitant la pondération $\omega_{i,j}$ à croître. Là encore, on assiste à la **formation** progressive d'un cluster où des entités possédant des blocs de règles similaires ou présentant une forte compatibilité se verront reliées par des liaisons renforcées, favorisant leur regroupement.

Conséquence : du renforcement local vers le cluster global

De manière générale, la mise à jour

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) \ + \ \eta \left[\mathrm{S}(i,j) - \tau \, \omega_{i,j}(t) \right]$$

assure que toute **paires** (i,j) bénéficiant d'une **synergie** élevée (qu'elle soit sub-symbolique ou symbolique) subisse un **renforcement** récurrent, tant que $\mathrm{S}(i,j)$ demeure au-dessus d'un certain point de compensation $\tau \, \omega_{i,j}$. Dans un environnement multi-entités, de **nombreuses** paires peuvent progressivement s'attirer, formant un **noyau** de liens forts. En se propageant, cet effet aboutit à la **coalescence** d'un ensemble d'entités partageant des attributs similaires ou des règles logiques compatibles, c'est-à-dire un **cluster** ou une **communauté** nettement identifiable dans le **SCN**.

Seuil et Trigger de Liaison

Il n'est pas rare de définir, de manière explicite ou implicite, un **seuil** θ pour la synergie $\mathrm{S}(i,j)$. Lorsqu'elle dépasse θ, la croissance de $\omega_{i,j}$ s'accélère. Cette heuristique peut être formalisée, par exemple, en introduisant un **terme** :

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\left[F(\mathrm{S}(i,j),\theta) - \tau\,\omega_{i,j}(t)\right],$$

où la fonction $F(\mathrm{S}(i,j),\theta)$ vaut 0 en dessous du seuil θ, et augmente rapidement au-delà. Cette approche accentue le **contraste** où des entités proches franchissent le seuil et se relient fortement, tandis que d'autres, trop différentes, ne voient pas leur liaison progresser. La réussite de ce **filtrage** dépend à nouveau de la **qualité** de la représentation initiale où si celle-ci n'est pas assez discriminante, ce système de seuil peut mener à des associations erronées ou, au contraire, trop restreintes.

3.2.3.2. Risque de Confusion si la Représentation est Trop Floue ou Trop Simplifiée

En miroir de l'idée qu'une **proximité** marquée dans l'espace de représentation favorise une **synergie** élevée — laquelle renforce la pondération $\omega_{i,j}$ —, il faut souligner la possibilité d'un **effet inversé** si la représentation n'est pas assez **discriminante**. En d'autres termes, quand la description d'une entité $\mathcal{E}_i$ est trop simplifiée, peu informative ou compressée, deux entités **objectivement** différentes risquent d'apparaître artificiellement "proches", gonflant leur synergie $\mathrm{S}(i,j)$ sans réel fondement. Cette confusion conduit alors le **Synergistic Connection Network (SCN)** à les regrouper dans un même cluster de manière erronée.

Représentation sub-symbolique et dimensionnalité insuffisante

Un cas courant se rencontre dans les approches **sub-symboliques** où l'on associe à chaque entité $\mathcal{E}_i$ un **vecteur** $\mathbf{x}_i \in \mathbb{R}^d$. Si la **dimension** d est trop faible pour couvrir la diversité des attributs, ou si l'on a appliqué une **réduction de dimension** trop agressive, par exemple une PCA drastique ou un autoencodeur trop compact, des entités pourtant distinctes finissent par se projeter **proches** dans $\mathbb{R}^d$. Mathématiquement, la distance $\| \mathbf{x}_i - \mathbf{x}_j \|$ ou la similarité cosinus $\frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\|\mathbf{x}_i\|\|\mathbf{x}_j\|}$ donne alors une valeur trompeusement élevée. La synergie $\mathrm{S}(i,j)$ en résulte **surévaluée** et le réseau en vient à renforcer $\omega_{i,j}$:

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) \;+\; \eta\left[\mathrm{S}(i,j) - \tau\,\omega_{i,j}(t)\right].$$

Ces rapprochements "factices" introduisent une confusion dans la **dynamique d'auto-organisation** où, au lieu de séparer clairement des entités distinctes, le SCN tend à créer des liens forts entre elles, formant ainsi un **cluster** incohérent. On peut aboutir à des macro-groupes qui englobent quantité d'entités hétérogènes, ou à des groupements aléatoires privés de sens réel.

Représentation symbolique simpliste

Dans un **cadre symbolique**, un problème similaire survient si l'on ne conserve qu'un **jeu restreint** de règles ou d'axiomes pour chaque entité. Par exemple, la fonction de compatibilité $\mathrm{Compat}(R_i, R_j)$ peut reposer sur la part d'axiomes communs entre deux ensembles R_i et R_j. Si ces ensembles se réduisent à quelques **règles** très générales (ex. "tout objet peut être rouge ou bleu"), alors de multiples entités paraîtront artificiellement "compatibles", faute de disposer de règles plus spécialisées pour les distinguer. La synergie $\mathrm{S}(i,j)$ demeure élevée pour bon nombre de couples, masquant les divergences réelles que des règles plus riches auraient mis en évidence.

Cette situation entraîne le même effet que dans l'espace vectoriel compressé où des **liaisons** $\omega_{i,j}$ s'en trouvent gonflées de façon injustifiée, menant à un **cluster unique** ou à des regroupements inadéquats. Dans le cas d'une ontologie appauvrie, deux concepts très différents peuvent apparaître près l'un de l'autre en raison d'un socle minimal d'axiomes partagés.

Distorsion de la synergie

D'un point de vue **mathématique**, on peut considérer qu'il existe une "**vraie**" similarité $\mathrm{S}^\star(i,j)$ que l'on souhaiterait approximer par $\mathrm{S}(i,j)$. Une représentation **trop floue** augmente l'écart $|\mathrm{S}^\star(i,j) - \mathrm{S}(i,j)|$. Cela se traduit par deux risques :

- **Surévaluation** : Deux entités très différentes ($\mathrm{S}^\star \approx 0$) apparaissent "proches" en S, conduisant à un renforcement indu de $\omega_{i,j}$ et à la formation de liens illusoires.
- **Sous-évaluation** : Inversement, des entités en réalité similaires ($\mathrm{S}^\star \approx 1$) reçoivent un score S faible, se retrouvant dispersées dans des clusters distincts.

Dans les deux cas, la **dynamique** du DSL dérive et forme des structures contraires à la **réalité** des entités (au sens de $\mathrm{S}^\star$). Le réseau se **désorganise** ou, pire, se retrouve dans un état quasi homogène où presque tout le monde est "lié" de façon grossière.

Clusters peu informatifs ou pseudo-homogénéité

Concrètement, une représentation simpliste ou appauvrie donne souvent lieu à des **clusters** anormalement étendus, mélangeant des entités sans vrai rapport. Sur le plan algorithmique, la mise à jour

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\left[\mathrm{S}(i,j) - \tau\,\omega_{i,j}(t)\right]$$

va favoriser des liaisons multiples, lesquelles finissent par "coller" ensemble un grand sous-ensemble. Lorsqu'on visualise la matrice $\{\omega_{i,j}\}$, elle tend à se **remplir** (saturer), aboutissant à une quasi-absence de différenciation.

Comment éviter ce problème ?

D'un point de vue **ingénierie**, il convient :

- D'adopter une **représentation** plus riche (augmenter la dimension vectorielle, introduire davantage d'attributs symboliques).
- De vérifier que les **attributs** sélectionnés sont véritablement discriminants (ne pas se contenter d'une statistique globale).
- D'envisager des **mécanismes** de filtrage ou d'inhibition (cf. chap. 4 et 7) pour ne pas laisser les synergies de faible consistance perturber la structure.
- Dans certains cas, de recourir à des **tests** externes ou à un oracle pour détecter les couples (i, j) qui seraient anormalement jugés "proches", et rétroagir sur la représentation.

3.2.3.3. Rôle de la Normalisation ou de la Calibration des Entités

Lorsque l'on conçoit un **Deep Synergy Learning (DSL)**, il ne suffit pas de sélectionner une représentation (vecteur, règles symboliques, etc.) pour chaque entité $\mathcal{E}_i$. Il est également crucial de **rendre** ces différentes descriptions **comparables**, de telle sorte que la fonction de synergie $\mathrm{S}(i, j)$ ne soit pas faussée par des échelles numériques trop divergentes ou par des attributs symboliques surpondérés. C'est ici qu'interviennent la **normalisation** et la **calibration**, qui visent à **harmoniser** la magnitude ou la pondération des descripteurs et à éviter une domination injustifiée de certaines composantes.

A. Problématique de l'Échelle (Cas Sub-Symbolique)

Lorsqu'une entité $\mathcal{E}_i$ est décrite par un **vecteur** $\mathbf{x}_i \in \mathbb{R}^d$, on peut rencontrer le problème de **dimensions** inégales où certaines coordonnées varient dans un intervalle restreint, par exemple entre 0 et 1, tandis que d'autres couvrent un éventail bien plus large, comme de 100 à 10 000. Pour la distance euclidienne,

$$\| \mathbf{x}_i - \mathbf{x}_j \| = \sqrt{\sum_{\ell=1}^{d} (x_{i,\ell} - x_{j,\ell})^2},$$

une composante numériquement très ample se met à **dominer** la distance, rendant négligeables les écarts sur les autres composantes. Cela peut conduire à des mesures de **synergie** $\mathrm{S}(i, j)$ (souvent définies comme une fonction décroissante de $\| \mathbf{x}_i - \mathbf{x}_j \|$) faussées, car la "dimension la plus grande" éclipse l'information apportée par les autres.

Pour éviter cette situation, on recourt à des **méthodes de normalisation** :

- **Min–max scaling** pour ramener chaque coordonnée dans $[0,1]$,
- **Z-score** où l'on retranche la moyenne et on divise par l'écart-type,
- ℓ_2**-normalisation** (projection sur la sphère unitaire) : $\mathbf{x'}_i = \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|}$.

Ces approches garantissent que la fonction de synergie ne soit pas **dominée** par un attribut particulier. À titre d'exemple, dans une similarité cosinus, on impose souvent $\| \mathbf{x}_i \| = 1$, alors

$j\mathrm{S}(i,j) = \mathbf{x}_i \cdot \mathbf{x}_j$. Toute différence d'échelle disparaît, et la distance ou la similarité répercute mieux les différences sémantiques réelles.

B. Homogénéité entre Entités

Au sein du **Synergistic Connection Network** (SCN), on peut avoir un grand nombre d'entités analogues (ex. plusieurs capteurs) — certains capteurs peuvent naturellement renvoyer des valeurs plus fortes (en volts ou en dB) tandis que d'autres restent dans des gammes plus faibles. Sans un processus de **calibration**, un capteur "fort" dominera la distance ou la similarité, amenant la pondération $\omega_{i,j}$ à se renforcer majoritairement pour ces entités, même si la différence n'est qu'un **artefact** d'échelle.

De même, en logique symbolique, si l'on attribue des **poids** inégaux aux règles (ex. une "règle principale" vaut 1000, les autres valent 1), alors le calcul de $\mathrm{Compat}\big(R_i, R_j\big)$ sera le plus souvent dicté par la présence ou l'absence de cette règle "majeure". Le réseau pourra, là encore, **exagérer** les proximités lorsque cette règle est partagée, formant ainsi des clusters artificiels.

La **calibration** consiste à imposer une **cohérence** dans la pondération où, par exemple, on peut normaliser la somme des poids symboliques afin que $\sum_{r\in R_i} w\,(r) = 1$. On peut alors définir :

$$\mathrm{S}_{\mathrm{sym}}(i,j) = \frac{\sum_{r\in R_i \cap R_j}\ w(r)}{\sum_{r\in R_i \cup R_j}\ w(r)},$$

de sorte qu'une règle unique n'emporte pas la totalité du calcul.

C. Éviter la "Dérive" au Fil du Temps

Dans un **environnement évolutif** (chap. 9), les entités peuvent voir leur représentation se **modifier** où un embedding peut "s'étirer" si le modèle d'apprentissage continu affine ses paramètres, et des règles supplémentaires peuvent être ajoutées à un ensemble symbolique. Si aucun mécanisme de recalibrage n'existe, on court le risque qu'**une** dimension ou **un** attribut gonfle sans limite, bouleversant progressivement la mesure de synergie. Cela se traduit par des **oscillations** dans la mise à jour de $\omega_{i,j}$ ou par l'absorption d'entités disparates dans un même cluster.

Une parade est de fixer un protocole de **renormalisation** ou de "**ré-étalonnage**" à intervalles réguliers, afin de maintenir l'ensemble des entités dans un même **espace** de comparaison :

$$\mathbf{r}(i, t+1) = h\big(\mathbf{r}(i,t)\big),$$

où hh effectue la recalibration nécessaire (remise à l'échelle, recentrage). On peut choisir un **pas** de recalibration adapté ou mettre en place un pilotage heuristique (ex. si la norme moyenne des vecteurs dépasse un seuil, on opère une projection globale).

D. Cas Multimodal ou Hybride

Dans des scénarios **multimodaux**, on a souvent besoin de **fusionner** un score de similarité sub-symbolique (ex. cosinus entre embeddings) et un score symbolique (ex. compatibilité de règles). Si l'on ne borne pas ou ne normalise pas ces scores, l'un peut dominer l'autre. Par exemple, un cosinus variant dans $[-1,1]$ se retrouve dilué face à un score symbolique qui peut atteindre 10 ou 100.

Une **stratégie** consiste à normaliser chaque sous-score dans $[0,1]$ ou $[-1,1]$, puis à combiner linéairement ou par un noyau plus sophistiqué :

$$S_{\text{hybride}}(i,j) = \alpha\, S_{\text{vect}}\big(\mathbf{x}_i, \mathbf{x}_j\big) + (1-\alpha)\, S_{\text{sym}}\big(R_i, R_j\big),$$

où α\alpha est un paramètre. On évite ainsi que le bloc logique (ou le vecteur) n'occupe toute la scène. L'**harmonisation** est d'autant plus critique si le DSL doit manipuler diverses modalités (vision, audio, texte, règles expertes) dans un même SCN.

3.3. Représentations Sub-Symboliques

Les approches sub-symboliques, basées sur des **vecteurs** ou des **embeddings**, constituent l'une des manières les plus répandues de décrire des entités $\mathcal{E}_i$ dans un contexte d'apprentissage automatique. Dans le cadre du DSL (Deep Synergy Learning), elles présentent l'avantage de se prêter à un **calcul** aisément quantifiable de la similarité (voire de la distance) entre entités, conditionnant ainsi la **synergie** $S(i,j)$. Dans cette section, nous étudierons tout d'abord les **vecteurs** et **embeddings** (3.3.1), puis nous aborderons des **méthodes avancées** (autoencodeurs, transformers, etc.) (3.3.2), avant de voir en détail le **calcul** de la synergie entre vecteurs (3.3.3) et la **gestion** du bruit ou de l'évolution de ces représentations (3.3.4).

3.3.1. Vecteurs et Embeddings

La représentation **vectorielle** est sans doute la plus intuitive dans un grand nombre d'applications où chaque entité est codée par un point $\mathbf{x}_i \in \mathbb{R}^d$. Cette approche, déjà ancienne en classification ou en clustering, prend une **nouvelle dimension** avec l'émergence des **embeddings profonds**, capables de saisir des caractéristiques complexes dans les images, les textes ou l'audio.

3.3.1.1. **Origine : CNN (images), Word Embeddings (textes), Spectrogrammes (audio)**

Dans le cadre d'un **Deep Synergy Learning (DSL)**, une grande partie des représentations sub-symboliques est obtenue par l'extraction d'**embeddings** à l'aide de **modèles neuronaux** spécialisés. L'idée fondamentale consiste à transformer un objet initial – qu'il s'agisse d'une image, d'un texte ou d'un segment audio – en un **vecteur** ou un **tenseur** qui capture les **caractéristiques** les plus pertinentes de l'information. Ce vecteur permet ensuite de calculer la **mesure** de synergie, notée $\mathrm{S}(i,j)$, qui reflète la proximité ou la compatibilité entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$. Diverses approches sont utilisées pour générer ces embeddings, et nous détaillons ici trois cas typiques.

Dans le domaine de l'analyse d'images, les **Convolutional Neural Networks (CNN)** ont largement popularisé l'extraction de **vecteurs** de caractéristiques. Un réseau convolutionnel, tel que VGG, ResNet, Inception ou encore Vision Transformer, procède à travers une série de couches de **convolution** et de **pooling** à l'extraction progressive de « feature maps » qui condensent l'information visuelle en attributs de plus en plus abstraits, tels que les **textures**, les **bords** ou encore les **formes**. Au terme de ce processus, on obtient un **embedding global** de dimension d (typiquement 256, 512 ou 1024), représenté par

$$\text{CNN:} \quad \mathbb{R}^{H\times W\times C} \;\rightarrow\; \mathbb{R}^{d},$$

où $H \times W$ indique la taille de l'image, C le nombre de canaux (par exemple, 3 pour une image RGB), et d la dimension du vecteur. Ce **descripteur** capture ainsi des aspects sémantiques essentiels, et dans un DSL, deux images dont les embeddings $\mathbf{x}_i$ et $\mathbf{x}_j$ se montrent très similaires (par exemple, mesurées via la **similarité cosinus** ou une distance euclidienne inversée) conduisent à une synergie élevée, entraînant le renforcement de la pondération $\omega_{i,j}$ dans le SCN et favorisant l'auto-organisation en clusters d'images visuellement cohérents.

Dans le domaine du **Traitement Automatique du Langage Naturel (TALN)**, la représentation des textes s'appuie sur les **word embeddings** tels que Word2Vec et GloVe, ainsi que sur des modèles contextuels plus récents comme BERT, GPT ou T5. Plutôt que d'utiliser des représentations statiques comme les vecteurs one-hot, on projette chaque mot ou token dans un **espace** de dimension d selon la relation

$$\mathbf{w}: \{\text{mots ou tokens}\} \rightarrow \mathbb{R}^d.$$

Ainsi, deux mots ou expressions d'une **signification** similaire se trouvent naturellement proches dans cet espace, ce qui facilite le calcul de distances ou de **similarités**. De plus, pour obtenir une représentation d'une phrase ou d'un document, il est courant d'agréger les embeddings des mots, par exemple par la moyenne ou en extrayant le vecteur associé au token [CLS] dans les modèles Transformer. Cette approche permet d'assigner à chaque segment textuel un vecteur $\mathbf{x}_i$ qui, lorsqu'il est comparé à un autre vecteur $\mathbf{x}_j$, donne une mesure de synergie reflétant la **parenté sémantique** entre les textes.

Pour les données **audio**, l'extraction d'embeddings se fait souvent à partir d'un **spectrogramme** ou par l'usage de modèles neuronaux spécialisés, tels que SoundNet ou des Audio Transformers. Le signal audio $a(t)$ est segmenté et transformé par un modèle spécifique, conduisant à un embedding

$$\mathbf{x}_s = \text{AudioModel}\big(a_s(t)\big) \quad \in \quad \mathbb{R}^d,$$

qui capture des caractéristiques acoustiques comme le **timbre** et la **prosodie**. Dans un DSL, si deux segments audio présentent des embeddings similaires, la synergie $\mathrm{S}(i,j)$ est alors élevée, et la pondération correspondante $\omega_{i,j}$ se renforce, permettant ainsi la formation de clusters cohérents de signaux acoustiques ou la fusion d'une modalité audio avec d'autres modalités dans un cadre multimodal.

D'un point de vue **formel**, après avoir extrait les vecteurs $\mathbf{x}_i$ pour chaque entité, on définit la **mesure** de similarité par exemple par la formule gaussienne

$$\mathrm{S}(i,j) = \exp\big(-\alpha \parallel \mathbf{x}_i - \mathbf{x}_j \parallel\big)$$

ou par la formule du **produit scalaire normalisé**

$$\mathrm{S}(i,j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\parallel \mathbf{x}_i \parallel \parallel \mathbf{x}_j \parallel}.$$

Lorsque ces scores de similarité sont intégrés dans la mise à jour

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[\mathrm{S}(i,j) - \tau\, \omega_{i,j}(t)\big],$$

le DSL renforce les liens entre les entités dont les embeddings sont proches dans l'espace, ce qui conduit à la formation de **clusters** auto-organisés reflétant la **cohérence** des caractéristiques sous-jacentes.

La **force** de cette approche réside dans la capacité des **embeddings** générés par ces modèles neuronaux à capturer une **richesse sémantique** et des aspects contextuels qui permettent d'identifier avec précision les similitudes entre entités issues de domaines variés (vision, langage,

acoustique). Cependant, le succès de cette méthode dépend fortement de la **qualité** des embeddings, laquelle doit être suffisamment robuste pour résister au bruit et capable de généraliser les caractéristiques essentielles des données. Ces principes, approfondis dans le Chapitre 3, soulignent l'importance de disposer de représentations vectorielles de haute qualité pour garantir une auto-organisation efficace au sein du DSL.

3.3.1.2. **Avantages : Calcul Aisé de Similarité (Cosinus, Distance Euclidienne)**

Dans le cadre du **Deep Synergy Learning (DSL)**, chaque entité $\mathcal{E}_i$ est représentée par un **vecteur** $\mathbf{x}_i \in \mathbb{R}^d$ qui est issu d'un modèle d'apprentissage profond tel qu'un CNN, un transformeur pour le langage ou encore un modèle audio pour les spectrogrammes. L'un des atouts majeurs de cette représentation sub-symbolique réside dans la **simplicité** algorithmique avec laquelle il est possible de calculer la **similarité** ou la **distance** entre ces vecteurs. Cette facilité de calcul joue un rôle central dans la formation et l'auto-organisation du **Synergistic Connection Network (SCN)**, puisque la **synergie** $\mathrm{S}(i,j)$ entre deux entités est directement fonction de la proximité mesurée dans l'espace vectoriel, et plus cette proximité est élevée, plus la **pondération** $\omega_{i,j}$ tend à être renforcée selon la règle de mise à jour du SCN.

D'un point de vue mathématique, la **similarité cosinus** est définie par

$$\mathrm{S}_{\cos}(i,j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\| \mathbf{x}_i \| \, \| \mathbf{x}_j \|},$$

cette mesure se concentrant sur l'**angle** entre $\mathbf{x}_i$ et $\mathbf{x}_j$ plutôt que sur leur norme, ce qui est particulièrement pertinent dans des contextes où la magnitude du vecteur n'est pas un indicateur de la **pertinence** ou de la **similarité**. En outre, une transformation linéaire de l'intervalle $[-1,1]$ permet d'obtenir un score de similarité dans $[0,1]$, par exemple

$$\mathrm{S}(i,j) = \frac{1 + \mathrm{S}_{\cos}(i,j)}{2},$$

ce qui garantit une interprétation directe dans le cadre de la mise à jour des pondérations. Par ailleurs, la **distance euclidienne** est définie par

$$d_{\mathrm{eucl}}(i,j) = \| \mathbf{x}_i - \mathbf{x}_j \| = \sqrt{\sum_{k=1}^{d} \left(x_{i,k} - x_{j,k}\right)^2},$$

et cette distance, en quantifiant la longueur du segment reliant $\mathbf{x}_i$ à $\mathbf{x}_j$, peut être convertie en un score de similarité par l'application d'une fonction de décroissance exponentielle, comme le montre la formule

$$\mathrm{S}_{\mathrm{gauss}}(i,j) = \exp\left(-\alpha \, \| \mathbf{x}_i - \mathbf{x}_j \|^2\right),$$

où $\alpha > 0$ est un paramètre réglable. Ces deux formules sont linéaires en la dimension d et reposent sur des opérations de multiplication–accumulation, qui sont bien adaptées aux architectures de calcul modernes telles que les GPU ou les bibliothèques optimisées en BLAS, permettant ainsi un

calcul rapide même pour de grands ensembles d'entités. La **simplicité** de ces mesures permet une implémentation efficace de la **mise à jour** des pondérations dans le SCN selon la formule

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[\mathrm{S}(i,j) - \tau\,\omega_{i,j}(t)\big],$$

ce qui conduit à un renforcement des liens entre les entités qui se rapprochent dans l'espace vectoriel. L'avantage de cette approche réside également dans le fait qu'elle permet une **interopérabilité** naturelle avec un grand nombre d'algorithmes existants, notamment ceux de **clustering** (tels que K-means, DBSCAN) et des techniques d'**approximate nearest neighbor** (comme FAISS ou Annoy), ce qui facilite l'analyse et la visualisation des regroupements obtenus via des méthodes de réduction de dimension telles que PCA, t-SNE ou UMAP. De plus, cette méthode se prête aisément à des scénarios **multimodaux** où, en projetant des données provenant de différentes sources dans un espace commun, on applique de manière uniforme la logique de calcul de la similarité, assurant ainsi une intégration homogène et une gestion efficace de la **variabilité** des données. Finalement, en ajustant les paramètres des kernels (comme α dans le cas du noyau gaussien), il est possible d'optimiser la **sensibilité** du DSL aux distances entre vecteurs, garantissant ainsi que la **dynamique** de mise à jour des pondérations reflète de manière fine les structures sous-jacentes des données.

3.5.3.3. Inconvénients : Dimension Élevée, Parfois Peu Interprétables

L'utilisation de **représentations sub-symboliques** sous forme de vecteurs d'embeddings, générés par des modèles neuronaux tels que les CNN ou les Transformers, offre indéniablement des avantages en termes de simplicité et d'efficacité du calcul de la synergie dans $\mathbb{R}^d$. Toutefois, ces méthodes présentent plusieurs inconvénients qui peuvent poser des problèmes de **scalabilité**, de **stabilité** et d'**explicabilité**. La discussion qui suit s'articule autour de quatre axes majeurs.

A. Dimension Élevée : Risques et Limitations

Les modèles neuronaux modernes produisent des embeddings dont la dimension peut atteindre plusieurs centaines voire milliers (par exemple, 512, 768 ou 1024 dimensions). Lorsque la fonction de synergie $\mathbf{S}(i,j)$ est calculée de manière naïve pour toutes les paires (i,j) au sein d'un Synergistic Connection Network (SCN) composé de n entités, le coût de calcul devient de l'ordre de $\mathrm{O}(n^2 \times d)$. Ainsi, si n est très grand – par exemple, plusieurs millions d'entités – et d élevé, le temps de calcul peut devenir prohibitif, comme l'illustre l'estimation

$$\text{Coût} \approx n^2 \times d.$$

De plus, la malédiction de la dimension peut rendre les distances moins discriminantes, puisque dans des espaces de très haute dimension, les distances entre la plupart des points tendent à converger vers une valeur similaire. Cette uniformisation des distances peut aboutir à une dynamique d'auto-organisation moins stable, avec des clusters qui se forment de manière peu distincte ou qui présentent des frontières floues. Face à ces difficultés, des approches telles que l'utilisation d'algorithmes d'**approximate nearest neighbors**, de techniques de **sparsification** ou de mécanismes de **sélection** ne traitant que les paires prometteuses sont souvent nécessaires afin de réduire la charge de calcul.

B. Opacité et Faible Interprétabilité

Les embeddings issus de réseaux neuronaux sont le produit d'opérations complexes et hiérarchiques réparties sur de multiples couches (convolutions, mécanismes d'attention, etc.). Cette complexité rend souvent l'interprétation de chaque coordonnée de l'embedding difficile. Il n'est pas aisé de déterminer si une dimension particulière de $\mathbf{x}_i$ correspond à une caractéristique précise, telle qu'une texture, une forme ou un concept lexical. De surcroît, lorsqu'on constate qu'un score de similarité élevé existe entre deux vecteurs $\mathbf{x}_i$ et $\mathbf{x}_j$, il est ardu d'attribuer de manière explicite la cause de cette proximité – qu'il s'agisse de la couleur, de la catégorie ou d'une co-occurrence textuelle. Cette opacité pose un réel problème dans des domaines exigeant une forte **explicabilité** ; ainsi, pour des applications critiques (médicales, industrielles, etc.), il est impératif de pouvoir justifier la formation d'un cluster ou la liaison entre deux entités. Même si des techniques de réduction de dimension, telles que PCA, UMAP ou t-SNE, permettent d'obtenir une vue globale des distributions, elles n'améliorent guère l'interprétation des attributs à l'échelle individuelle.

C. Instabilité des Embeddings et Variations Contextuelles

Les embeddings neuronaux peuvent varier sensiblement en fonction du **contexte** ou des modifications apportées lors de la phase d'entraînement. Dans le domaine du traitement du langage naturel, par exemple, une légère modification du corpus d'entraînement ou un nouveau fine-tuning peut entraîner des décalages dans les positions des mots ou des tokens dans l'espace vectoriel. De même, en vision par ordinateur, l'adaptation d'un réseau pour intégrer de nouvelles classes d'images peut modifier la distribution finale des embeddings $\mathbf{x}_i$. De telles variations contextuelles impactent directement la mesure de la synergie, que ce soit par la modification de la distance $\| \mathbf{x}_i - \mathbf{x}_j \|$ ou par le produit scalaire $\mathbf{x}_i \cdot \mathbf{x}_j$. Cette instabilité peut ainsi perturber la dynamique du SCN, induisant des réorganisations brutales ou imprévues dans l'auto-organisation, et posant un défi pour la stabilité à long terme du DSL.

D. Nécessité d'un Prétraitement Potentiellement Coûteux

L'obtention de ces embeddings repose sur l'utilisation de modèles préentraînés, tels que des réseaux CNN pour les images ou des Transformers pour les textes, qui nécessitent une infrastructure de calcul robuste (GPU/TPU) et des ressources importantes pour l'entraînement ou l'inférence. Si l'objectif est de concevoir un DSL autonome fonctionnant sur des ressources matérielles limitées, le coût en termes de calcul et de temps associé au prétraitement des données peut s'avérer prohibitif. De plus, le déploiement d'un tel système requiert la mise en place d'un pipeline complet, capable de fournir en continu des embeddings de haute qualité, ce qui représente une contrainte supplémentaire en termes d'ingénierie et d'infrastructure.

3.3.2. Autoencodeurs, Transformers et Approches Avancées

Les vecteurs et embeddings présentés en section 3.3.1 constituent déjà une base solide pour représenter des entités de nature variée (images, textes, sons). Toutefois, les **techniques** d'apprentissage ne cessant d'évoluer, il est possible d'aller **plus loin** en recourant à des méthodes avancées qui extraient des embeddings encore plus **expressifs** ou plus **compacts**. Dans cette section, nous verrons d'abord (3.3.2.1) comment les **autoencodeurs** peuvent servir à la **réduction dimensionnelle** ou à l'**extraction de features**, puis (3.3.2.2) comment les **Transformers** (BERT,

GPT, ViT) produisent des embeddings **contextuels**, et enfin (3.3.2.3) nous soulignerons **l'intérêt** de ces méthodes avancées pour le DSL, au prix toutefois d'une complexité de calcul plus élevée.

3.3.2.1. Autoencodeurs pour Réduction Dimensionnelle ou Extraction de Features

Dans le cadre d'un **Deep Synergy Learning (DSL)**, l'extraction d'informations pertinentes à partir de représentations sub-symboliques constitue un enjeu majeur pour l'efficacité du processus d'auto-organisation. Les **autoencodeurs (AE)** se présentent comme un outil fondamental permettant d'apprendre, de manière non supervisée, un **code latent z** qui capture l'essence d'un vecteur d'entrée **x**. Cette démarche offre deux avantages essentiels pour le DSL où elle permet la **réduction de dimension** et l'**extraction de features discriminantes**.

A. Principe et Architecture d'un Autoencodeur

Un autoencodeur se compose typiquement de deux modules complémentaires. Le premier, l'**encodeur** E, est une fonction $E: \mathbb{R}^d \to \mathbb{R}^k$ qui reçoit un vecteur d'entrée $\mathbf{x} \in \mathbb{R}^d$ et produit un **code latent** $\mathbf{z} \in \mathbb{R}^k$, où $k < d$ dans le but de compresser l'information. Le second module, le **décodeur** D, opère la transformation inverse, $D: \mathbb{R}^k \to \mathbb{R}^d$, et tente de reconstruire le vecteur d'entrée à partir de ce code latent, produisant ainsi $\hat{\mathbf{x}} = D\big(E(\mathbf{x})\big)$. L'apprentissage se réalise en minimisant une **fonction de perte de reconstruction** $\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}})$, souvent formulée comme la moyenne des erreurs quadratiques, ce qui s'exprime par

$$\min_{E,D} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}\left(\mathbf{x}_i,\ D\big(E(\mathbf{x}_i)\big)\right).$$

Cette optimisation force l'encodeur à extraire les facteurs de variation les plus pertinents de **x**, tout en éliminant le bruit et les détails superflus, aboutissant ainsi à une représentation compacte et significative.

B. Avantages : Réduction de Dimension et Extraction de Traits

L'imposition d'un goulot d'étranglement, où la dimension du code latent k est inférieure à celle de l'entrée d, engendre deux bénéfices notables pour le DSL. D'une part, la **réduction dimensionnelle** permet de diminuer significativement le coût de calcul lors de l'évaluation des distances ou similarités entre entités. En effet, le calcul de la distance entre deux codes latents, par exemple $\| \mathbf{z}_i - \mathbf{z}_j \|$, se réalise en $O(k)$ plutôt qu'en $O(d)$, ce qui est particulièrement avantageux lorsque le réseau traite un grand nombre d'entités. D'autre part, en apprenant à reconstruire **x** à partir de **z**, l'autoencodeur extrait des **features discriminantes** qui résument les aspects sémantiques les plus importants du vecteur d'origine, facilitant ainsi la formation de clusters cohérents et la détection de similarités véritables dans le SCN.

C. Variantes : Denoising, Sparse et Variational Autoencoder

Il existe plusieurs variantes d'autoencodeurs qui visent à améliorer la robustesse et la qualité des représentations latentes.

Premièrement, le **Denoising Autoencoder** introduit du bruit dans $\mathbf{x}$ durant l'entraînement et apprend à reconstruire la version nettoyée de la donnée. Ce procédé confère au code latent $\mathbf{z}$ une meilleure tolérance aux perturbations, ce qui est particulièrement utile dans des environnements où les données sont bruitées.

Ensuite, le **Sparse Autoencoder** incorpore une contrainte de sparsité sur $\mathbf{z}$ – par exemple, via une pénalisation de la norme L_1 – de sorte que seule une fraction limitée des neurones s'active pour chaque donnée, ce qui peut améliorer l'interprétabilité des caractéristiques extraites et favoriser une séparation plus nette des classes.

Enfin, le **Variational Autoencoder (VAE)** adopte une approche probabiliste en imposant que le code latent suive une distribution prédéfinie, généralement gaussienne, c'est-à-dire $\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma^2} I)$. Cette formulation permet non seulement de générer de nouvelles données, mais aussi d'organiser l'espace latent de manière plus lisse, facilitant ainsi la continuité et la robustesse des clusters formés lors de l'auto-organisation.

D. Intégration au DSL : La Synergie entre Codes Latents

Une fois l'autoencodeur entraîné, la partie **encodeur** E est utilisée pour transformer chaque vecteur d'entrée $\mathbf{x}_i$ en un code latent $\mathbf{z}_i = E(\mathbf{x}_i)$ dans un espace de dimension réduite $\mathbb{R}^k$. Ce code latent remplace alors $\mathbf{x}_i$ dans la phase de calcul de la synergie au sein du DSL. La mesure de synergie peut être définie par des formules classiques telles que

$$\mathbf{S}(i,j) = \exp\left(-\alpha \parallel \mathbf{z}_i - \mathbf{z}_j \parallel^2\right)$$

ou par une mesure basée sur la similarité cosinus

$$\mathbf{S}(i,j) = \frac{\mathbf{z}_i \cdot \mathbf{z}_j}{\parallel \mathbf{z}_i \parallel \parallel \mathbf{z}_j \parallel}.$$

Ces scores de synergie, calculés dans un espace de dimension réduite, facilitent le renforcement des pondérations $\omega_{i,j}$ et la formation de clusters dans le Synergistic Connection Network. L'optimisation de la dimension k joue alors un rôle critique où si k est trop petit, l'autoencodeur risque de perdre des informations essentielles, tandis qu'un k trop élevé n'apporte pas le gain de complexité recherché.

E. Limites et Points de Vigilance

Malgré leurs avantages, les autoencodeurs présentent quelques inconvénients notables.

Premièrement, la minimisation de la **perte de reconstruction** n'est pas nécessairement alignée avec l'objectif de séparer distinctement les classes ou de maximiser la discrimination entre les entités. Par conséquent, les codes latents $\mathbf{z}_i$ peuvent parfois ne pas être suffisamment discriminants pour la tâche d'auto-organisation.

Deuxièmement, l'entraînement d'un autoencodeur sur des datasets volumineux peut être coûteux en termes de ressources computationnelles et de temps, notamment lorsqu'il s'agit d'architectures profondes.

Troisièmement, le choix de la dimension latente k est crucial où une valeur inappropriée peut soit sous-estimer les facteurs de variation essentiels, soit ne pas fournir une véritable réduction de complexité. Enfin, dans un contexte d'apprentissage continu, la mise à jour de l'encodeur E doit être soigneusement gérée pour éviter le phénomène de **catastrophic forgetting**, qui pourrait altérer l'organisation de l'espace latent déjà établi.

3.3.2.2. Transformers (BERT, GPT, ViT) : Embeddings Contextuels plus Riches

L'émergence des **Transformers** – tels que BERT, GPT, T5 en NLP et Vision Transformer (ViT) en vision – a profondément transformé la manière dont sont générés les **embeddings** dans divers domaines. Contrairement aux représentations plus statiques (comme Word2Vec) ou aux approches purement convolutionnelles (CNN), les Transformers produisent des **embeddings contextuels**. Ces représentations intègrent non seulement l'information d'un token ou d'un patch de manière isolée, mais elles capturent également le **contexte global** par le biais d'un mécanisme de *self-attention*. Dans le cadre d'un **Deep Synergy Learning (DSL)**, cette capacité à incorporer le contexte se traduit par des mesures de synergie plus fines et expressives, où la proximité entre deux entités reflète non seulement des similarités de caractéristiques isolées, mais aussi des interactions complexes au sein de l'ensemble des données.

A. Principe des Transformers et Self-Attention

L'architecture Transformer repose sur un mécanisme de *self-attention* qui permet à chaque position d'une séquence de « regarder » toutes les autres positions. Soit une séquence d'entrées représentées par la matrice $\mathbf{x} \in \mathbb{R}^{n \times d}$, où chaque ligne correspond à un embedding initial d'un token ou d'un patch. Les Transformers calculent trois matrices essentielles par multiplication avec des poids appris :

$$\mathbf{Q} = \mathbf{x}\, W_Q, \quad \mathbf{K} = \mathbf{x}\, W_K, \quad \mathbf{V} = \mathbf{x}\, W_V,$$

où W_Q, W_K, et W_V sont des matrices de projection de dimensions appropriées (souvent $d \times d_k$). Le mécanisme de self-attention se définit alors par l'opération

$$\mathrm{Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\,\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}.$$

Cette opération permet à chaque token (ou patch) de pondérer les contributions de tous les autres éléments en fonction de la similitude entre ses **requêtes** et les **clés** associées aux autres tokens, produisant ainsi des embeddings qui intègrent l'information contextuelle de l'ensemble de la séquence.

B. BERT, GPT : Embeddings Contextuels en NLP

Dans le domaine du **Traitement Automatique du Langage Naturel (NLP)**, des modèles tels que **BERT** et **GPT** exploitent pleinement le mécanisme de self-attention pour produire des **embeddings contextuels**. Concrètement, après plusieurs couches de self-attention et de transformations non linéaires, chaque token i de la séquence obtient une représentation finale $\mathbf{h}_i^{(L)}$. Deux types d'extractions sont couramment utilisées :

- **Token-level** : Chaque mot ou sous-mot se voit attribuer un embedding $\mathbf{h}_i$ qui capture ses nuances contextuelles.
- **Embedding global** : Un vecteur représentatif de l'ensemble de la séquence est généré, souvent en utilisant le token spécial $[CLS]$ (dans le cas de BERT) ou par agrégation (moyenne ou max) sur tous les tokens.

Dans un DSL, ces embeddings globaux servent de « signature » vectorielle pour une entité textuelle. La synergie entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ peut ainsi être évaluée par des mesures telles que la similarité cosinus :

$$\mathrm{S}(i,j) = \frac{\mathbf{z}_i \cdot \mathbf{z}_j}{\| \mathbf{z}_i \| \, \| \mathbf{z}_j \|},$$

où $\mathbf{z}_i$ et $\mathbf{z}_j$ sont les embeddings finaux issus du Transformer. Grâce à la prise en compte du contexte, ces représentations permettent de discerner des relations sémantiques fines, telles que la synonymie contextuelle ou les paraphrases, améliorant ainsi la qualité de l'auto-organisation du réseau.

C. Vision Transformer (ViT) : Patches et Self-Attention en Vision

Les principes du Transformer se sont également appliqués au domaine de la vision par ordinateur via le **Vision Transformer (ViT)**. Dans ce cas, une image est divisée en petits **patches** (par exemple, de 16×16 pixels), qui sont linéarisés et projetés dans un espace de dimension d. Pour conserver l'information spatiale, des **positional embeddings** sont ajoutés à chaque patch. Le Transformer traite alors l'ensemble des patches de manière similaire aux tokens en NLP, utilisant le mécanisme de self-attention pour capturer des relations globales dans l'image. La sortie finale peut être constituée d'un token spécial $[CLS]$ agissant comme représentation globale ou d'une agrégation des embeddings individuels. Cette approche permet de générer des embeddings d'images qui intègrent non seulement des caractéristiques locales, mais aussi des dépendances à longue portée entre différentes régions de l'image, améliorant ainsi la mesure de la synergie dans le DSL.

D. Enjeux et Coûts : Dimension et Ressources

Bien que les Transformers produisent des embeddings contextuels particulièrement riches, ils impliquent également des coûts importants en termes de **dimension** et de **ressources**. Par exemple, les modèles tels que BERT ou GPT génèrent des vecteurs de dimension 768, 1024 ou plus, ce qui peut amplifier la complexité des calculs de similarité dans un SCN, souvent de l'ordre de $\mathrm{O}(n^2 \times d)$ pour n entités. De plus, le poids des modèles (parfois plusieurs centaines de millions de paramètres) requiert une infrastructure matérielle (GPU, TPU) conséquente pour l'inférence et le fine-tuning. Enfin, la sensibilité des Transformers aux ajustements fins – un léger changement de paramètres peut modifier la distribution des embeddings – nécessite des techniques d'alignement ou de versionnage pour préserver la stabilité des représentations dans un système d'apprentissage continu.

E. Intégration dans un DSL

L'intégration des Transformers dans un DSL se réalise en extrayant, pour chaque entité $\mathcal{E}_i$ (qu'il s'agisse d'un document, d'une image ou d'un segment audio), un embedding contextuel $\mathbf{z}_i$ via une opération telle que

$$\mathbf{z}_i = \text{TransformerEncode}(\mathbf{x}_i),$$

où $\mathbf{x}_i$ représente l'objet brut initial et TransformerEncode désigne le processus de passage par plusieurs couches de self-attention et de feed-forward. Une fois obtenu, l'embedding peut être normalisé afin de faciliter le calcul de similarités, par exemple en utilisant le cosinus. La synergie entre deux entités est alors définie par une fonction f (comme le cosinus ou un noyau RBF) :

$$\text{S}(i,j) = f\big(\mathbf{z}_i, \mathbf{z}_j\big).$$

Dans le processus de mise à jour des pondérations du SCN,

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[\text{S}(i,j) - \tau\,\omega_{i,j}(t)\big],$$

une synergie élevée entre des embeddings contextuels conduira au renforcement des liens, favorisant ainsi la formation de clusters d'entités sémantiquement cohérents, tout en capturant des relations complexes que seules des approches traditionnelles auraient du mal à discerner.

3.3.2.3. **Intérêt pour le DSL : un Embedding plus Robuste, mais plus Complexe à Calculer ou à Mettre à Jour**

Dans le cadre d'un **Deep Synergy Learning (DSL)**, l'emploi de techniques avancées telles que les **Transformers** ou les **autoencodeurs profonds** permet d'obtenir des **embeddings** d'une qualité remarquable, caractérisés par une richesse contextuelle et une capacité de discrimination supérieure. Ces représentations, issues de modèles sophistiqués, se distinguent par leur aptitude à capter des nuances sémantiques fines et à structurer l'espace latent de manière à favoriser la formation de clusters d'entités plus robustes. On définit notamment, par une fonction g, l'embedding contextuel d'une entité initiale $\mathbf{x}_i$ par

$$\mathbf{z}_i = g(\mathbf{x}_i),$$

ce qui permet de mesurer la **synergie** entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ à l'aide d'une distance ou d'un produit scalaire, par exemple

$$\text{S}(i,j) = \exp\big(-\alpha \parallel \mathbf{z}_i - \mathbf{z}_j \parallel^2\big) \quad \text{ou} \quad \text{S}(i,j) = \frac{\mathbf{z}_i \cdot \mathbf{z}_j}{\parallel \mathbf{z}_i \parallel \parallel \mathbf{z}_j \parallel},$$

ce qui permet de regrouper les entités ayant des caractéristiques latentes similaires au sein du **Synergistic Connection Network (SCN)**. Ces méthodes confèrent au DSL une **robustesse** accrue dans la capture des variations contextuelles et une capacité à distinguer subtilement des différences sémantiques que des approches plus simples auraient tendance à négliger.

Cependant, la qualité supérieure des embeddings contextuels se traduit par une **complexité** notable tant dans le calcul de l'inférence que dans la gestion des mises à jour de ces représentations. Le

modèle g, en intégrant plusieurs couches de self-attention et de transformations non linéaires, nécessite en général une infrastructure matérielle puissante, telle que des GPU ou TPU, en raison du nombre élevé de paramètres et de la dimension des vecteurs, souvent de l'ordre de 768, 1024 ou plus. Par conséquent, le coût de calcul d'une opération d'inférence, qui peut être évalué en $O(n^2 \times d_{\text{latent}})$ pour un ensemble de n entités, augmente considérablement, en particulier lorsque le DSL doit traiter de grands volumes de données en temps réel. Cette complexité algorithmique se trouve également exacerbée par le coût associé à la mise à jour des embeddings dans un contexte d'apprentissage continu. En effet, si le modèle g subit un fine-tuning ou une révision partielle, il devient nécessaire de recalculer les embeddings pour toutes les entités, c'est-à-dire

$$\mathbf{z}_i(t+1) = g_{t+1}(\mathbf{x}_i) \quad \text{pour } i = 1, \ldots, n,$$

ce qui, pour des ensembles de données de grande taille, peut engendrer une latence non négligeable et provoquer des fluctuations dans les valeurs de $S(i,j)$ entre les cycles de mise à jour. Ces variations, même modestes, peuvent induire des réorganisations brutales au sein du SCN, compromettant ainsi la stabilité de l'auto-organisation.

L'ingénierie d'un DSL doit donc établir un compromis entre la **richesse** des embeddings, qui améliore la précision des regroupements en capturant des informations contextuelles complexes, et l'**efficacité computationnelle** requise pour leur calcul et leur mise à jour régulière. Dans certains contextes critiques, où la qualité de la représentation est primordiale – par exemple en médecine ou en analyse de données sensibles – il peut être acceptable de supporter des coûts de calcul élevés pour garantir une **précision** et une **robustesse** accrues. À l'inverse, dans des applications massives où la rapidité d'inférence est essentielle, des modèles plus légers ou des embeddings de dimension réduite pourraient être privilégiés afin d'optimiser la scalabilité du système.

L'utilisation d'embeddings plus riches, issus de techniques avancées, permet d'améliorer significativement la **discrimination** et la **robustesse** des synergies dans un DSL, mais se fait au prix d'une complexité computationnelle accrue et d'un défi supplémentaire en termes de mise à jour des représentations dans un cadre d'apprentissage continu. Ce compromis entre la richesse des représentations et l'efficacité opérationnelle constitue un enjeu central dans la conception et l'implémentation de systèmes de Deep Synergy Learning.

3.3.3. Calcul de la Synergie entre Vecteurs

Dans les sections précédentes (3.3.1 et 3.3.2), nous avons mis en évidence l'usage de **vecteurs** (embeddings) pour représenter des entités dans un contexte sub-symbolique, ainsi que l'apport de méthodes avancées (autoencodeurs, Transformers) pour produire des embeddings plus riches. Il reste à voir **comment**, à partir de ces vecteurs, on définit la **synergie** $S(i,j)$ entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$. En pratique, le DSL (Deep Synergy Learning) recourt souvent à des **fonctions** de distance ou de similarité géométriques. Cette section (3.3.3) se divise en trois points :

- (3.3.3.1) présentation des **distances usuelles** (euclidienne, manhattan) et de la **similarité cosinus**,
- (3.3.3.2) usage de **kernels** pour une similarité non linéaire,

- (3.3.3.3) **ajustements** (normalisation, calibration) pour mieux gérer le bruit ou l'échelle des vecteurs.

3.3.3.1. **Distances Usuelles : Euclidienne, Manhattan, Cosinus**

Dans le cadre du **Deep Synergy Learning (DSL)**, chaque entité $\mathcal{E}_i$ est représentée par un **vecteur** $\mathbf{x}_i \in \mathbb{R}^d$. L'évaluation de la **synergie** entre deux entités repose alors sur la comparaison de ces représentations via une fonction de **distance** ou de **similarité**. Ces mesures jouent un rôle fondamental dans le processus d'**auto-organisation** du **Synergistic Connection Network (SCN)**, puisque plus deux vecteurs sont « proches » dans l'espace, plus la **pondération** $\omega_{i,j}$ tend à être renforcée. Parmi les approches classiques pour mesurer cette proximité, on trouve la **distance euclidienne**, la **distance manhattan** et la **similarité cosinus**, chacune offrant une interprétation géométrique différente et répondant à des exigences spécifiques selon le type de données traitées.

*A. **Distance Euclidienne***

La **distance euclidienne** constitue la mesure la plus intuitive, puisqu'elle correspond à la longueur du segment reliant les points représentés par $\mathbf{x}_i$ et $\mathbf{x}_j$ dans $\mathbb{R}^d$. Mathématiquement, elle se définit par

$$d_{\text{eucl}}(i,j) = \| \mathbf{x}_i - \mathbf{x}_j \| = \sqrt{\sum_{k=1}^{d} \left(x_{i,k} - x_{j,k}\right)^2}.$$

La valeur ainsi obtenue représente une notion de **distance** physique dans l'espace, et dans le DSL, cette distance est souvent convertie en un **score de similarité** afin de faciliter son intégration dans la dynamique du SCN. Une fonction d'agrégation classique consiste à appliquer une décroissance exponentielle, ce qui donne

$$\mathbf{S}(i,j) = \exp\left(-\alpha \ \| \mathbf{x}_i - \mathbf{x}_j \|^2\right),$$

où $\alpha > 0$ est un **paramètre** qui ajuste la pente de décroissance. Dans ce contexte, une distance faible entre $\mathbf{x}_i$ et $\mathbf{x}_j$ conduit à un score proche de 1, indiquant une **affinité élevée** entre les deux entités et justifiant ainsi le renforcement de la pondération $\omega_{i,j}$.

*B. **Distance Manhattan (ou ℓ_1)***

La **distance manhattan** ou distance **taxicab** s'exprime en prenant la somme des différences absolues entre les composantes correspondantes des vecteurs. La formule est donnée par

$$d_{\text{manh}}(i,j) = \sum_{k=1}^{d} \left|x_{i,k} - x_{j,k}\right|.$$

Cette mesure, qui évalue la différence cumulée par dimension, est particulièrement utile lorsque les données se combinent de façon additive, et elle présente l'avantage de se montrer souvent plus **robuste** face aux valeurs extrêmes de certaines coordonnées. Pour intégrer cette distance dans un

DSL, on peut transformer la mesure en un score de similarité en appliquant par exemple la fonction exponentielle

$$\mathbf{S}(i,j) = \exp\big(-\beta\, d_{\text{manh}}(i,j)\big),$$

ou encore en utilisant la forme rationnelle

$$\mathbf{S}(i,j) = \frac{1}{1 + d_{\text{manh}}(i,j)},$$

où $\beta > 0$ est un **paramètre** qui ajuste le taux de décroissance. De cette manière, une distance manhattan réduite se traduit par un score de similarité élevé, indiquant une forte **affinité** entre les entités.

C. Similarité Cosinus

La **similarité cosinus** se distingue des mesures de distance classiques en se focalisant sur l'**angle** entre deux vecteurs plutôt que sur leur distance absolue. Elle est définie par

$$\mathbf{S}_{\cos}(i,j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\| \mathbf{x}_i \| \, \| \mathbf{x}_j \|},$$

et prend des valeurs dans l'intervalle $[-1,1]$. Cette mesure atteint la valeur 1 lorsque les vecteurs sont colinéaires dans la même direction, -1 lorsqu'ils sont opposés et 0 lorsque les vecteurs sont orthogonaux. Dans le cadre du DSL, il est souvent souhaitable d'obtenir un score strictement positif, et pour ce faire, on peut transformer l'intervalle en appliquant par exemple la formule

$$\mathbf{S}(i,j) = \frac{1 + \mathbf{S}_{\cos}(i,j)}{2},$$

qui convertit les valeurs de $[-1,1]$ en $[0,1]$. Cette méthode permet de neutraliser l'effet de la norme des vecteurs, en se concentrant uniquement sur leur **direction**, ce qui est particulièrement utile pour des données textuelles ou pour des représentations où la magnitude peut varier de manière significative.

D. Liaison avec la Synergie dans le DSL

L'ensemble des mesures présentées permet de définir la **synergie** entre deux entités, laquelle est utilisée pour ajuster la **pondération** $\omega_{i,j}$ dans le SCN. La règle de mise à jour se formalise par

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[\mathbf{S}(i,j) - \tau\, \omega_{i,j}(t)\big],$$

où $\eta > 0$ est le **taux d'apprentissage** et $\tau > 0$ représente le **coefficient de décroissance**. Ce mécanisme d'auto-organisation implique que plus deux entités sont évaluées comme étant proches (ou similaires) selon la mesure choisie, plus la pondération $\omega_{i,j}$ tend à augmenter, favorisant ainsi la formation de **clusters** cohérents. Le choix entre la **distance euclidienne**, la **distance manhattan** ou la **similarité cosinus** dépend du type de données et des propriétés spécifiques recherchées, car chacune offre des avantages distincts. La distance euclidienne fournit une mesure classique de la proximité, la distance manhattan peut mieux gérer certaines disparités dimensionnelles et la similarité cosinus permet de se focaliser sur l'orientation des vecteurs indépendamment de leur norme.

3.3.3.2. Kernels (RBF, Polynomial) pour une Similarité non Linéaire

Les **distances** usuelles (euclidienne, manhattan) ou la **similarité** cosinus (section 3.3.3.1) considèrent des relations souvent **linéaires** ou du moins directes dans l'espace vectoriel. Cependant, il arrive que la **relation** réelle entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ soit plus subtile, nécessitant des méthodes **non linéaires** pour en rendre compte. Dans un **Deep Synergy Learning (DSL)**, recourir à des **kernels** (noyaux) permet d'introduire des notions de **similarité** bien plus riches que les simples distances ℓ_p. Les **RBF kernels** (noyaux gaussiens) et **kernels polynomiaux** figurent parmi les plus répandus et se traduisent en un **score** $\mathrm{S}(i,j)$ apte à révéler des liens complexes cachés dans l'espace initial $\mathbb{R}^d$.

A. Principe Général des Kernels

Un **kernel** k entre deux vecteurs $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$ se définit comme une **fonction** :

$$k\colon\ \mathbb{R}^d \times \mathbb{R}^d\ \rightarrow\ \mathbb{R},$$

renvoyant un score (positif ou non) quantifiant leur **affinité**. La **particularité** du kernel trick, issu de la théorie des **noyaux de Mercer**, réside dans le fait que $k(\mathbf{x}_i, \mathbf{x}_j)$ peut être interprété comme un **produit scalaire** dans un espace de dimension éventuellement infinie :

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathcal{H}},$$

où $\phi\colon \mathbb{R}^d \rightarrow \mathcal{H}$ est une **transformation** non linéaire, et $\mathcal{H}$ un **espace** (souvent de Hilbert) de dimension potentiellement énorme. La **magie** du kernel trick est qu'on n'a pas besoin de **calculer** explicitement ϕ. Il suffit d'évaluer $k(\mathbf{x}_i, \mathbf{x}_j)$. D'un point de vue **DSL**, cela équivaut à définir la **synergie** $\mathrm{S}(i,j)$:

$$\mathrm{S}(i,j) = k(\mathbf{x}_i, \mathbf{x}_j).$$

Deux entités peuvent se trouver "lointaines" selon une métrique linéaire, tout en étant **fortement similaires** en termes de transformations non linéaires. Dans un **Synergistic Connection Network**, on peut alors capter des formes de **ressemblance** plus complexes, sans explicitement projeter les entités dans un espace étendu.

B. RBF Kernel (Radial Basis Function)

Le **kernel RBF** (aussi appelé "noyau gaussien") constitue un choix très répandu. Il s'écrit :

$$k_{\mathrm{RBF}}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma\ \|\ \mathbf{x}_i - \mathbf{x}_j\ \|^2\right) \quad (\gamma > 0).$$

Il s'agit d'une **exponentielle** négative de la distance euclidienne au carré. On peut l'interpréter comme une "**gaussienne**" centrée sur $\mathbf{x}_j$. Plus $\mathbf{x}_i$ est proche de $\mathbf{x}_j$, plus la valeur se rapproche de 1 ; plus ils sont éloignés, plus elle tend vers 0. Le **paramètre** γ détermine la "largeur" du noyau :

- Un γ **grand** induit une décroissance rapide de $k_{\mathrm{RBF}}(\mathbf{x}_i, \mathbf{x}_j)$ dès que $\|\ \mathbf{x}_i - \mathbf{x}_j\ \|$ grandit, ce qui focalise la similarité sur une zone très **locale**.

- Un γ **petit** rend la fonction plus "plate", donnant une vue plus **globale** où des entités modérément éloignées conservent une similarité notable.

Pour un **DSL**, un **RBF kernel** donne une **synergie** :

$$\mathrm{S}(i,j) = \exp\left(-\gamma \parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2\right).$$

Cette forme **non linéaire** permet à des entités ayant un écart euclidien significatif (dans $\mathbb{R}^d$) de tout de même afficher une similarité mesurable si elles partagent certaines **propriétés** sous-jacentes. En outre, si γ est bien paramétré, la synergie reflète de subtiles variations dans les données — plus subtiles, parfois, qu'une simple distance euclidienne.

C. Polynomial Kernel

Un autre kernel usuel est le **kernel polynomial** :

$$k_{\mathrm{poly}}(\mathbf{x}_i, \mathbf{x}_j) = \left(\mathbf{x}_i \cdot \mathbf{x}_j + c\right)^p,$$

où p est le **degré** (typiquement 2, 3, 4…) et $c \geq 0$ un terme de décalage. Cette définition incorpore toutes les **composantes** polynomiales du produit $\mathbf{x}_i \cdot \mathbf{x}_j$, allant jusqu'au degré p. Ainsi, pour $p = 2$, on inclut toutes les interactions bilinéaires $\left(x_{i,k}x_{j,\ell}\right)$. Cela peut capturer des **corrélations** plus complexes qu'un simple "angle" ou qu'une distance linéaire.

Dans un **SCN**, si on adopte $\mathrm{S}(i,j) = k_{\mathrm{poly}}(\mathbf{x}_i, \mathbf{x}_j)$, on se retrouve avec une **synergie** qui peut s'avérer très sensible à des variations dans certaines dimensions, surtout pour un p élevé. Cela peut amplifier des **propriétés** souhaitées (s'il y a de fortes corrélations polynomiales), mais risque aussi de **surexposer** le bruit ou les valeurs extrêmes. Le choix de p et c réclame donc un réglage attentif.

D. Non-Linéarité et Espace Implicite : Kernel Trick

Le **kernel trick** signifie que l'on n'a **pas** besoin de définir explicitement une transformation $\phi(\cdot)$ vers un espace de dimension (parfois énorme) $\mathcal{H}$. Il suffit de calculer $k(\mathbf{x}_i, \mathbf{x}_j)$. Dans un **DSL**, cela se traduit par une **synergie** :

$$\mathrm{S}(i,j) = k(\mathbf{x}_i, \mathbf{x}_j) \quad \text{où } k \text{ est RBF, polynomial, ou autre.}$$

Mathématiquement, deux vecteurs lointains en $\mathbb{R}^d$ peuvent devenir "proches" dans l'espace $\mathcal{H}$, si la transformation ϕ induite par le kernel met en évidence des attributs partagés. Ainsi, la **mise à jour** $\omega_{i,j}(t+1) = \omega_{i,j}(t) + \cdots$ tient compte de propriétés non linéaires. Dans la **formation** de clusters, des entités qui ne semblaient pas se ressembler au sens euclidien peuvent se rapprocher si leur **synergie** kernel s'avère élevée.

E. Exemples d'Implémentation dans un SCN

Pour un **SCN** gérant n entités, on définit $\mathrm{S}(i,j) = k(\mathbf{x}_i, \mathbf{x}_j)$. À chaque itération, calculer $\omega_{i,j}(t+1)$ exige d'évaluer ces kernels. Sur le plan **coût** :

- **RBF** : $\mathrm{O}(d)$ multiplications pour $\parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2$, puis une exponentiation.
- **Polynomial** : $\mathrm{O}(d)$ pour le produit scalaire $\mathbf{x}_i \cdot \mathbf{x}_j$, puis l'élévation au degré p.

Pour comparer toutes les paires (i, j), on fait face à $O(n^2)$ évaluations, chaque en $O(d)$. Si n est grand, cela devient lourd (voir chap. 3.2.2.2). En pratique, on peut élaguer en ne calculant la synergie que sur un **voisinage** restreint (approximate nearest neighbor, etc.) ou recourir à d'autres heuristiques d'échantillonnage pour gérer la **scalabilité**.

F. Limites et Paramétrage Sensible

Le **RBF kernel** nécessite le choix d'un paramètre γ. Un γ trop grand produit une décroissance exponentielle ultra-rapide, rendant la synergie presque nulle pour la majorité des paires, ce qui peut isoler excessivement les entités. Un γ trop petit, en revanche, aboutit à un noyau très plat, de sorte que presque tout le monde se retrouve "similaire" : on aboutit à un **mégacluster**. Le **polynomial kernel** demande aussi de fixer p et c. Un p trop élevé peut amplifier les bruits, alors qu'un p trop faible (ex. 1) redescend à un comportement linéaire classique.

L'ingénieur d'un **DSL** doit donc **régler** ces hyperparamètres en fonction des données, par validation croisée ou d'autres heuristiques. Les kernels offrent une flexibilité précieuse, mais nécessitent une **phase** d'ajustement pour qu'ils produisent des **synergies** stables et cohérentes dans le SCN.

3.3.3.3. Ajustements : Normalisation, Calibration, Prise en Compte du Bruit

Dans un **Deep Synergy Learning (DSL)** où les entités $\mathcal{E}_i$ sont décrites par des **vecteurs** ou des attributs numériques, l'évaluation de la **synergie** $S(i, j)$ repose sur des **distances** (euclidienne, manhattan) ou des **similarités** (cosinus, kernels). Toutefois, dans la **pratique**, il ne suffit pas de choisir une métrique : il faut aussi **calibrer** et **ajuster** la façon dont on compare les vecteurs. Les questions de normalisation, d'échelle, de saturation, ou encore de **prise en compte** du bruit sont essentielles pour garantir la **cohérence** du calcul de $S(i, j)$ et la stabilité de la mise à jour $\omega_{i,j}$. Les sections qui suivent décrivent en détail comment on gère ces ajustements pour un **Synergistic Connection Network (SCN)** robuste et efficace.

A. Normalisation et Équilibrage des Vecteurs

Lorsqu'on manipule des vecteurs $\mathbf{x}_i \in \mathbb{R}^d$ pour calculer la synergie $S(i, j)$, il arrive souvent que certaines **dimensions** varient sur des plages beaucoup plus larges que d'autres, ou que différentes entités $\mathcal{E}_i$ n'utilisent pas le même **ordonnancement** de valeurs. Sans précautions, la mesure de distance ou de similarité peut s'en trouver **biaisée**. Deux mécanismes de base s'appliquent :

Normalisation en norme

Un classique consiste à normaliser chaque vecteur à la **norme** 1, c'est-à-dire :

$$\mathbf{x}'_i = \frac{\mathbf{x}_i}{\| \mathbf{x}_i \|}, \quad \| \mathbf{x}'_i \| = 1.$$

Ainsi, la **similarité cosinus** entre $\mathbf{x}'_i$ et $\mathbf{x}'_j$ devient un **produit scalaire** direct. Cela annule l'influence de la **magnitude** brute et met l'accent sur l'**angle**. Dans un DSL multimodal, cette démarche évite qu'une modalité sortant des valeurs très élevées ne domine la mesure de proximité.

Normalisation par composante

On peut aussi appliquer un **z-score** (soustraction de la moyenne et division par l'écart-type), ou un min–max scaling dans l'intervalle [0,1], par dimension. Cette approche assure que chaque coordonnée contribue équitablement. Dans un **SCN**, elle prévient la situation où une composante "géante" écrase toutes les autres dans le calcul de $\| \mathbf{x}_i - \mathbf{x}_j \|$ ou $\mathbf{x}_i \cdot \mathbf{x}_j$.

En pratique, la **choix** entre ces normalisations dépend de la distribution des données. L'objectif reste de **préserver** le signal discriminant et de **limiter** l'influence de différences d'échelle, améliorant ainsi la comparabilité et la **qualité** du calcul de $\mathrm{S}(i,j)$.

B. Calibration du Score de Synergie

Une fois les vecteurs normalisés, on définit la synergie par :

$$\mathrm{S}(i,j) = \phi\left(d(\mathbf{x}_i, \mathbf{x}_j)\right) \quad \text{ou} \quad \psi\left(\mathrm{sim}(\mathbf{x}_i, \mathbf{x}_j)\right),$$

où ϕ et ψ sont des transformations (éventuellement exponentielles, sigmoïdes, inverses, etc.). Il se peut qu'on désire **calibrer** la sortie pour moduler l'amplitude. Par exemple :

Noyau gaussien

$$\mathrm{S}(i,j) = \exp\left(-\alpha \ \| \mathbf{x}_i - \mathbf{x}_j \|^2\right),$$

où $\alpha > 0$ gouverne la rapidité de décroissance. Si α est trop grand, la synergie chute quasi immédiatement pour tout écart modéré, regroupant uniquement les entités quasi identiques. Si α est trop petit, elle reste trop élevée pour un large spectre de distances, fusionnant potentiellement tout en un seul méga-cluster.

Distance inversée

$$\mathrm{S}(i,j) = \frac{1}{1 + \beta\, d(\mathbf{x}_i, \mathbf{x}_j)},$$

où $\beta > 0$. Ici, plus la distance $\| \mathbf{x}_i - \mathbf{x}_j \|$ grandit, plus S tend vers 0 mais jamais exactement 0. Le réglage de β dicte le "taux" d'atténuation.

Troncature ou saturation

On peut saturer la sortie pour ne pas dépasser 1 (ou un certain seuil), ou imposer un plancher minimal. Ainsi, la synergie reste dans $[\varepsilon, 1-\varepsilon]$ afin d'éviter un renforcement ou un affaiblissement trop extrême. Ceci peut stabiliser la mise à jour :

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\left[\mathrm{S}(i,j) - \tau\, \omega_{i,j}(t)\right].$$

Le point crucial est de **calibrer** S de façon à refléter *juste assez* la disparité entre entités, sans aboutir à des valeurs saturées pour la plupart des paires (synergie trop extrême) ou trop faibles (peu de différenciation).

C. Prise en Compte du Bruit

Dans bien des environnements (capteurs, images à faible qualité, textes “sales” ou hétérogènes), les **données** sont bruitées. Ce bruit peut fausser le calcul de distance $\| \mathbf{x}_i - \mathbf{x}_j \|$ ou de similarité $\mathbf{x}_i \cdot \mathbf{x}_j$. Plusieurs stratégies se dégagent :

Filtrage ou prétraitement. Avant tout calcul de distance, on peut nettoyer ou lisser $\mathbf{x}_i$, par exemple via du **smoothing** ou un **autoencodeur denoising**. Ce faisant, on réduit la variabilité de bas niveau, améliorant la robustesse de la synergie.

Troncature ou clipping des valeurs extrêmes. Si certaines composantes subissent des pics aberrants, on peut fixer un **cap** sur l’amplitude. Cela évite qu’une seule dimension bruitée ne domine la distance.

Injection de bruit contrôlé. Parfois, on ajoute soi-même un bruit gaussien au vecteur $\mathbf{x}_i$ (ou pendant l’apprentissage) pour **stabiliser** l’auto-organisation, un peu à la manière d’un recuit simulé. Cette démarche évite de se figer dans des minima locaux, et autorise une **exploration** plus large du SCN. Au fil du temps, on diminue ce bruit pour **affiner** la convergence.

Robustesse des fonctions de distance. Au lieu de la distance euclidienne ℓ_2, on peut employer ℓ_1 (manhattan), réputée moins sensible aux outliers sur une coordonnée. On peut également introduire des fonctions qui saturent au-delà d’une certaine différence, réduisant l’influence des valeurs extrêmes.

3.3.4. Gestion du Bruit et de l’Évolution des Embeddings

Même si les vecteurs et embeddings (section 3.3.3) constituent une base pratique pour représenter les entités dans un DSL (Deep Synergy Learning), ils ne sont pas figés. Dans la **réalité** d’un système en évolution où les capteurs changent, les nouvelles données s’intègrent et le modèle s’améliore continuellement, il faut prendre en compte le **bruit** potentiel dans les données ou dans les embeddings, pouvant entraîner des distances $\| \mathbf{x}_i - \mathbf{x}_j \|$ artificiellement élevées ou basses. Il est aussi essentiel de considérer la **mise à jour** ou le re-entraînement des modèles produisant les embeddings comme les réseaux neuronaux ou Transformers, ce qui peut modifier la représentation $\mathbf{x}_i$ au fil du temps. Enfin, les **stratégies** de filtrage, de clipping et d’ajustement local sont cruciales pour éviter la propagation d’erreurs ou la dérive progressive.

Ainsi, dans cette section, nous verrons comment le DSL peut gérer ces aspects (bruit, évolution) et préserver la **cohérence** du calcul de synergie $S(i, j)$.

3.3.4.1. **Embeddings Potentiellement Réentraînés, Fine-Tuned (Chap. 9 sur Apprentissage Continu)**

Dans le cadre du **Deep Synergy Learning (DSL)**, les **embeddings** $\mathbf{x}_i$ qui caractérisent les entités $\mathcal{E}_i$ ne sont pas considérés comme des représentations statiques et définitives. En effet, ces vecteurs, générés par des modèles neuronaux tels que des **CNN**, des **Transformers** ou des **autoencodeurs**, peuvent évoluer au fil du temps lorsque le modèle générateur est soumis à un processus de **fine-**

tuning ou à un schéma d'**apprentissage continu**. Ce phénomène de mise à jour dynamique des embeddings influence directement la **synergie** $\mathrm{S}(i,j)$ entre entités ainsi que la manière dont les **pondérations** $\omega_{i,j}$ sont ajustées dans le **Synergistic Connection Network (SCN)**.

A. Fine-Tuning et Mise à Jour des Embeddings

Considérons qu'à un instant t, chaque entité $\mathcal{E}_i$ est décrite par un **embedding** $\mathbf{x}_i(t)$ obtenu via un modèle g_t, qui s'exprime par la relation

$$\mathbf{x}_i(t) = g_t(\mathbf{d}_i),$$

où $\mathbf{d}_i$ représente les données brutes associées à l'entité, telles qu'une image, un texte ou un signal audio. Lorsqu'un processus de **fine-tuning** est initié – c'est-à-dire que le modèle g_t est réentraîné ou ajusté en réponse à de nouvelles données ou à un objectif révisé – le modèle évolue vers une nouvelle version g_{t+1} et l'embedding correspondant se met à jour selon

$$\mathbf{x}_i(t+1) = g_{t+1}(\mathbf{d}_i).$$

Même une modification modeste des poids du modèle peut entraîner une transformation significative de l'espace vectoriel, modifiant ainsi les distances $\| \mathbf{x}_i(t) - \mathbf{x}_j(t) \|$ entre entités et, par conséquent, la valeur de la **synergie** $\mathrm{S}(i,j)$. Ce réajustement induit une réorganisation du SCN, puisque la règle de mise à jour des pondérations

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[\mathrm{S}(i,j,t) - \tau\,\omega_{i,j}(t)\big],$$

dépend directement des représentations actuelles $\mathbf{x}_i(t)$ et $\mathbf{x}_j(t)$. Une telle dynamique impose une attention particulière dans la gestion des mises à jour pour ne pas perturber brusquement la structure des clusters déjà établis.

B. Apprentissage Continu et Flux Dynamique

Dans un scénario d'**apprentissage continu**, le modèle g_t est régulièrement mis à jour pour intégrer un nouveau flux de données, noté $\Delta\mathcal{D}$. La relation de mise à jour du modèle s'exprime alors par

$$g_{t+1} = \mathrm{Train}(g_t, \Delta\mathcal{D}),$$

ce qui conduit à une évolution progressive des embeddings selon

$$\mathbf{x}_i(t+1) = g_{t+1}(\mathbf{d}_i).$$

Ce mécanisme est particulièrement précieux dans des environnements non stationnaires où de nouvelles classes ou de nouvelles variations de données apparaissent de manière régulière. Toutefois, cette flexibilité s'accompagne d'une instabilité potentielle, puisque la transformation du modèle peut modifier de façon substantielle la géométrie de l'espace latent. Les distances entre les embeddings – et donc les scores de synergie – peuvent varier brutalement, induisant des réorganisations dans le SCN qui se traduisent par des fluctuations des pondérations $\omega_{i,j}$.

C. Impact sur la Mise à Jour des Pondérations

La dynamique d'auto-organisation du SCN est directement affectée par l'évolution des embeddings. La règle de mise à jour des pondérations

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\left[\mathrm{S}(i,j,t) - \tau\,\omega_{i,j}(t)\right]$$

utilise la **synergie** $\mathrm{S}(i,j,t)$, qui dépend des embeddings à l'instant t. Si, suite à un fine-tuning ou à un apprentissage continu, les embeddings se modifient de manière significative, la valeur de $\mathrm{S}(i,j,t+1)$ peut être très différente de celle précédemment obtenue. Ces variations peuvent provoquer des **sauts** ou des oscillations dans les pondérations, affectant la stabilité du regroupement des entités dans le réseau. Pour limiter ce phénomène, il est souvent recommandé d'adopter des stratégies de **lissage** dans la transition entre $\mathbf{x}_i(t)$ et $\mathbf{x}_i(t+1)$, par exemple en définissant une mise à jour pondérée telle que

$$\mathbf{x}_i(t+1) \leftarrow \alpha\,\mathbf{x}_i(t) + (1-\alpha)\,g_{t+1}(\mathbf{d}_i),$$

où le paramètre $\alpha \in [0,1]$ contrôle l'ampleur de la transition, permettant ainsi d'introduire progressivement les changements et de préserver la cohérence de la dynamique du SCN.

D. Équilibre entre Adaptation et Stabilité

L'adaptabilité des embeddings grâce au fine-tuning et à l'apprentissage continu offre une capacité essentielle à un DSL, puisqu'elle permet au système de rester pertinent face à l'évolution des données. Toutefois, cette adaptabilité se heurte à la nécessité de maintenir une stabilité dans l'organisation des entités. Un modèle qui se modifie trop fréquemment risque de déstabiliser les **clusters** existants, car les pondérations $\omega_{i,j}$ basées sur d'anciennes représentations ne seront plus en adéquation avec la nouvelle configuration de l'espace latent. Un compromis se trouve souvent dans la planification de mises à jour épisodiques, suivies de périodes de stabilisation, ou dans l'application de techniques d'adaptation incrémentale qui permettent un ajustement progressif. L'ingénieur se doit de régler la fréquence et l'ampleur des mises à jour de g_t afin de concilier la **richesse contextuelle** des nouvelles représentations et la nécessité d'une **stabilité** du SCN.

E. Gestion Concrète : Références au Chapitre 9 sur l'Apprentissage Continu

Le **chapitre 9** du présent ouvrage se penche en profondeur sur les techniques d'**apprentissage continu** et les stratégies permettant de gérer les mises à jour des embeddings sans compromettre la cohérence globale du DSL. Il y est présenté des protocoles d'adaptation incrémentale, des heuristiques de recalibrage des pondérations, ainsi que des mécanismes visant à atténuer le phénomène de **catastrophic forgetting**. Ces approches incluent notamment l'actualisation progressive des représentations, le verrouillage temporaire de certaines couches du modèle, et l'indexation régulière des embeddings afin d'assurer que la transition entre différentes versions du modèle reste la plus fluide possible.

3.3.4.2. Filtrage Local (k-NN) ou Clipping pour Limiter la Propagation d'Erreurs

Dans le cadre d'un **Deep Synergy Learning (DSL)**, la qualité et la stabilité de la **synergie** entre entités reposent sur la précision des **embeddings** et des **scores de similarité** calculés à partir de ceux-ci. Cependant, en présence de **perturbations** telles que le bruit, des valeurs aberrantes ou des variations imprévues dans les données, il devient indispensable de mettre en place des **mécanismes** visant à limiter la propagation des erreurs qui pourraient déstabiliser l'ensemble du **Synergistic Connection Network (SCN)**. Deux approches complémentaires se distinguent dans cette optique : le **filtrage local** par l'algorithme des **k-plus-proches-voisins (k-NN)** et le **clipping** des vecteurs

ou des scores de similarité. Ces stratégies, en restreignant l'influence d'entités extrêmes ou aberrantes, contribuent à préserver la stabilité globale du système.

A. Filtrage Local via k-Plus-Proches-Voisins (k-NN)

Lorsqu'une entité $\mathcal{E}_i$ est caractérisée par un **embedding** $\mathbf{x}_i \in \mathbb{R}^d$, il est fréquent, dans un SCN, de calculer la **synergie** $\mathrm{S}(i,j)$ entre toutes les paires (i,j). Cette approche, qui aboutit à un graphe complet, peut introduire des liaisons peu informatives ou même trompeuses, notamment lorsque des perturbations locales affectent certains embeddings. Afin de remédier à ce problème, il est judicieux de restreindre le calcul de la synergie à un **voisinage** immédiat, en considérant uniquement les k entités les plus proches de $\mathbf{x}_i$. Formellement, le voisinage local d'une entité se définit par

$$\mathrm{NN}_k(i) = \{j \in \{1, \dots, n\} \mid j \text{ fait partie des } k \text{ plus proches entités de } \mathbf{x}_i\}.$$

En conséquence, la fonction de synergie est tronquée de manière à ce que

$$\mathrm{S}'(i,j) = \begin{cases} \mathrm{S}(i,j), & \text{si } j \in \mathrm{NN}_k(i) \text{ ou } i \in \mathrm{NN}_k(j), \\ 0, & \text{sinon,} \end{cases}$$

ce qui signifie que seules les entités jugées suffisamment proches contribuent à la mise à jour des **pondérations** $\omega_{i,j}$. L'**avantage** de cette approche réside dans la réduction de la complexité, passant d'un nombre de comparaisons de l'ordre de $\mathrm{O}(n^2)$ à $\mathrm{O}(n\,k)$, tout en **limitant** l'impact des **outliers** qui, étant isolés, ne figurent pas dans le voisinage local. En outre, cette stratégie contribue à renforcer la **robustesse** du DSL en ne construisant des liaisons significatives qu'entre des entités réellement compatibles dans l'espace $\mathbb{R}^d$.

B. Clipping des Vecteurs et des Scores de Similarité

En complément du filtrage local, le **clipping** constitue une méthode efficace pour restreindre l'influence d'**valeurs extrêmes**. Cette technique peut s'appliquer à différents niveaux du processus d'inférence. Lorsqu'il s'agit des **embeddings** eux-mêmes, il est souvent pertinent de contraindre chaque coordonnée du vecteur $\mathbf{x}_i$ afin d'éviter que des perturbations locales ne provoquent une domination de certaines dimensions. Pour ce faire, on impose un seuil $T > 0$ et on définit le vecteur « clippé » par

$$x_{i,k}^{\mathrm{clip}} = \min\{\max\{x_{i,k}, -T\}, T\}.$$

Cette opération garantit que les valeurs de chaque dimension restent contenues dans l'intervalle $[-T, T]$, ce qui permet de préserver une **stabilité** géométrique dans $\mathbb{R}^d$. Par ailleurs, il est également possible d'appliquer le clipping directement au **score de synergie**. Si, par exemple, la fonction $\mathrm{S}(i,j)$ calcule une similarité qui pourrait parfois dépasser une borne raisonnable à cause d'un outlier, on peut définir une version clipée de cette synergie par

$$\mathrm{S}'(i,j) = \min\{\mathrm{S}(i,j), S_{\max}\},$$

où $S_{\max}$ est une borne supérieure prédéfinie. Cette approche empêche que la **synergie** n'atteigne des valeurs trop élevées qui pourraient induire un renforcement disproportionné des pondérations $\omega_{i,j}$, préservant ainsi l'équilibre global du SCN.

C. Combinaison des Approches pour une Robustesse Accrue

Dans la pratique, l'application conjointe du **filtrage local** par k-NN et du **clipping** constitue une stratégie robuste pour limiter la propagation d'erreurs. En effet, le filtrage local restreint d'abord le calcul de la synergie aux voisins les plus proches, réduisant la densité du graphe et la probabilité d'étendre l'influence d'entités aberrantes. Par la suite, le clipping, appliqué sur les vecteurs ou directement sur les scores de similarité, permet de modérer l'impact de toute valeur extrême qui pourrait encore se glisser dans le processus. Le SCN met ainsi à jour les pondérations selon la formule

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[\mathrm{S}'(i,j) - \tau\,\omega_{i,j}(t)\big],$$

où $\mathrm{S}'(i,j)$ représente la synergie obtenue après application des mécanismes de filtrage et de clipping. Cette approche intégrée permet de réduire la complexité computationnelle, en limitant le nombre de comparaisons nécessaires, tout en préservant la stabilité de l'auto-organisation en atténuant l'impact des perturbations locales.

3.3.4.3. **Exemples de Scénarios Audio-Visuels**

Dans le cadre du **Deep Synergy Learning (DSL)**, les **applications audio-visuelles** offrent un terrain d'expérimentation particulièrement riche pour illustrer la capacité du système à **auto-organiser** des entités issues de modalités diverses telles que l'image, le son et le texte. En effet, le **Synergistic Connection Network (SCN)** se doit de prendre en compte des **embeddings** extraits de flux multimodaux qui, de par leur nature, peuvent être affectés par le **bruit**, des changements de **contexte** ou des ajustements successifs induits par un fine-tuning continu. Dans cette section, nous exposons en détail deux scénarios qui démontrent comment le DSL exploite des mécanismes de limitation d'erreurs, tels que le filtrage local et le clipping, ainsi que des stratégies d'adaptation dynamique, afin de préserver une organisation robuste et cohérente dans un environnement audio-visuel.

A. Détection d'Événements Audio-Visuels

Dans un premier exemple, nous considérons la détection d'un événement, tel qu'un applaudissement ou un klaxon, qui se manifeste simultanément dans la composante visuelle et acoustique d'un flux multimodal. Pour ce faire, le flux audio-visuel est segmenté en une série de **frames vidéo** $\{\mathcal{E}_i^{(\mathrm{vid})}\}$ et en une série de **segments audio** $\{\mathcal{E}_j^{(\mathrm{aud})}\}$. Chaque frame est alors transformée en un **embedding** $\mathbf{x}_i^{(\mathrm{vid})} \in \mathbb{R}^{d_{\mathrm{vid}}}$ par un modèle de type CNN (tel que ResNet) ou par un Vision Transformer, tandis que chaque segment audio se voit attribuer un vecteur $\mathbf{x}_j^{(\mathrm{aud})} \in \mathbb{R}^{d_{\mathrm{aud}}}$ issu d'un modèle d'autoencodeur appliqué sur un spectrogramme ou d'un transformeur audio tel que wav2vec. La **synergie** entre une frame vidéo et un segment audio peut être quantifiée à l'aide d'un noyau gaussien, défini par

$$S(i,j) = \exp\Big(-\gamma \parallel \mathbf{x}_i^{(\mathrm{vid})} - \mathbf{x}_j^{(\mathrm{aud})} \parallel^2\Big),$$

où $\gamma > 0$ ajuste la sensibilité de la mesure. Toutefois, en raison des perturbations inhérentes à ce type de flux – telles que la saturation audio ou les variations lumineuses dans les images –, des

valeurs aberrantes peuvent fausser le calcul de la synergie. Pour limiter ces effets, le DSL intègre des mécanismes de **filtrage local** en restreignant le calcul de la similarité aux paires d'entités se chevauchant temporellement, ainsi que des techniques de **clipping** qui borner les valeurs extrêmes du score $S(i,j)$. Ce filtrage permet de constituer des **clusters** robustes reliant les frames et segments audio fortement synchronisés, ce qui facilite la détection d'événements multimodaux avec une grande précision.

B. Sous-Titrage Automatique à Partir d'un Flux Audio-Visuel

Dans un second scénario, le DSL est appliqué à la tâche de sous-titrage automatique, qui implique la fusion d'informations issues du flux vidéo et du flux textuel dérivé d'une transcription audio. Dans ce contexte, chaque frame vidéo $\mathcal{E}_i^{(\text{vid})}$ se voit attribuer un embedding $\mathbf{x}_i^{(\text{vid})}$ généré par un Vision Transformer ou un CNN, tandis que la transcription issue d'un module speech-to-text est convertie en un **embedding textuel** $\mathbf{x}_j^{(\text{text})} \in \mathbb{R}^{d_{\text{text}}}$ via un modèle de type BERT ou GPT. La synergie entre la représentation visuelle et celle textuelle peut être évaluée à l'aide de la **similarité cosinus**, exprimée par

$$S_{\cos}(i,j) = \frac{\mathbf{x}_i^{(\text{vid})} \cdot \mathbf{x}_j^{(\text{text})}}{\| \mathbf{x}_i^{(\text{vid})} \| \, \| \mathbf{x}_j^{(\text{text})} \|}.$$

Cette mesure permet de quantifier la proximité sémantique entre le contenu visuel et le texte associé. Cependant, des erreurs peuvent survenir dans la transcription, notamment en cas de bruit, d'accents ou de débit de parole rapide, tandis que la qualité des embeddings vidéo peut être affectée par des mouvements brusques ou des changements de scène. Afin de pallier ces imprécisions, le DSL met en place des mécanismes de **limitation temporelle** (en ne comparant que les frames proches du segment textuel) ainsi que des procédures de **clipping** pour éviter que des valeurs extrêmes ne conduisent à des scores de similarité erronés. De cette manière, le système parvient à associer de manière cohérente les segments vidéo et les extraits textuels, facilitant ainsi la génération de sous-titres ou de résumés contextuels de haute qualité.

C. Ajustement Dynamique et Fine-Tuning

Dans les deux scénarios précédents, le DSL peut être soumis à des processus de **fine-tuning** ou à des mises à jour continues, de sorte que les modèles générateurs d'embeddings (qu'il s'agisse d'un CNN, d'un Vision Transformer ou d'un modèle speech-to-text) évoluent en fonction des nouvelles données. Ainsi, les représentations $\mathbf{x}_i$ et $\mathbf{x}_j$ peuvent être réévaluées périodiquement, modifiant la synergie $S(i,j)$ calculée et, par conséquent, les pondérations $\omega_{i,j}$ dans le SCN. Afin d'éviter que ces changements brusques ne perturbent la dynamique globale du réseau, des techniques de **lissage** des transitions et de **verrouillage** partiel des modèles sont mises en œuvre. Ces stratégies visent à adapter les embeddings de manière progressive, garantissant ainsi une réorganisation continue mais stable des clusters d'entités multimodales.

3.3. Représentations Sub-Symboliques

Les approches sub-symboliques, basées sur des **vecteurs** ou des **embeddings**, constituent l'une des manières les plus répandues de décrire des entités $\mathcal{E}_i$ dans un contexte d'apprentissage automatique. Dans le cadre du DSL (Deep Synergy Learning), elles présentent l'avantage de se prêter à un **calcul** aisément quantifiable de la similarité (voire de la distance) entre entités, conditionnant ainsi la **synergie** $S(i,j)$. Dans cette section, nous étudierons tout d'abord les **vecteurs** et **embeddings** (3.3.1), puis nous aborderons des **méthodes avancées** (autoencodeurs, transformers, etc.) (3.3.2), avant de voir en détail le **calcul** de la synergie entre vecteurs (3.3.3) et la **gestion** du bruit ou de l'évolution de ces représentations (3.3.4).

3.3.1. Vecteurs et Embeddings

La représentation **vectorielle** est sans doute la plus intuitive dans un grand nombre d'applications où chaque entité est codée par un point $\mathbf{x}_i \in \mathbb{R}^d$. Cette approche, déjà ancienne en classification ou en clustering, prend une **nouvelle dimension** avec l'émergence des **embeddings profonds**, capables de saisir des caractéristiques complexes dans les images, les textes ou l'audio.

3.3.1.1. **Origine : CNN (images), Word Embeddings (textes), Spectrogrammes (audio)**

Dans le cadre d'un **Deep Synergy Learning (DSL)**, une grande partie des représentations sub-symboliques est obtenue par l'extraction d'**embeddings** à l'aide de **modèles neuronaux** spécialisés. L'idée fondamentale consiste à transformer un objet initial – qu'il s'agisse d'une image, d'un texte ou d'un segment audio – en un **vecteur** ou un **tenseur** qui capture les **caractéristiques** les plus pertinentes de l'information. Ce vecteur permet ensuite de calculer la **mesure** de synergie, notée $\mathrm{S}(i,j)$, qui reflète la proximité ou la compatibilité entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$. Diverses approches sont utilisées pour générer ces embeddings, et nous détaillons ici trois cas typiques.

Dans le domaine de l'analyse d'images, les **Convolutional Neural Networks (CNN)** ont largement popularisé l'extraction de **vecteurs** de caractéristiques. Un réseau convolutionnel, tel que VGG, ResNet, Inception ou encore Vision Transformer, procède à travers une série de couches de **convolution** et de **pooling** à l'extraction progressive de « feature maps » qui condensent l'information visuelle en attributs de plus en plus abstraits, tels que les **textures**, les **bords** ou encore les **formes**. Au terme de ce processus, on obtient un **embedding global** de dimension d (typiquement 256, 512 ou 1024), représenté par

$$\text{CNN:} \quad \mathbb{R}^{H\times W\times C} \;\rightarrow\; \mathbb{R}^{d},$$

où $H \times W$ indique la taille de l'image, C le nombre de canaux (par exemple, 3 pour une image RGB), et d la dimension du vecteur. Ce **descripteur** capture ainsi des aspects sémantiques essentiels, et dans un DSL, deux images dont les embeddings $\mathbf{x}_i$ et $\mathbf{x}_j$ se montrent très similaires (par exemple, mesurées via la **similarité cosinus** ou une distance euclidienne inversée) conduisent à une synergie élevée, entraînant le renforcement de la pondération $\omega_{i,j}$ dans le SCN et favorisant l'auto-organisation en clusters d'images visuellement cohérents.

Dans le domaine du **Traitement Automatique du Langage Naturel (TALN)**, la représentation des textes s'appuie sur les **word embeddings** tels que Word2Vec et GloVe, ainsi que sur des modèles contextuels plus récents comme BERT, GPT ou T5. Plutôt que d'utiliser des représentations statiques comme les vecteurs one-hot, on projette chaque mot ou token dans un **espace** de dimension d selon la relation

$$\mathbf{w}: \{\text{mots ou tokens}\} \rightarrow \mathbb{R}^d.$$

Ainsi, deux mots ou expressions d'une **signification** similaire se trouvent naturellement proches dans cet espace, ce qui facilite le calcul de distances ou de **similarités**. De plus, pour obtenir une représentation d'une phrase ou d'un document, il est courant d'agréger les embeddings des mots, par exemple par la moyenne ou en extrayant le vecteur associé au token [CLS] dans les modèles Transformer. Cette approche permet d'assigner à chaque segment textuel un vecteur $\mathbf{x}_i$ qui, lorsqu'il est comparé à un autre vecteur $\mathbf{x}_j$, donne une mesure de synergie reflétant la **parenté sémantique** entre les textes.

Pour les données **audio**, l'extraction d'embeddings se fait souvent à partir d'un **spectrogramme** ou par l'usage de modèles neuronaux spécialisés, tels que SoundNet ou des Audio Transformers. Le signal audio $a(t)$ est segmenté et transformé par un modèle spécifique, conduisant à un embedding

$$\mathbf{x}_s = \text{AudioModel}\big(a_s(t)\big) \quad \in \quad \mathbb{R}^d,$$

qui capture des caractéristiques acoustiques comme le **timbre** et la **prosodie**. Dans un DSL, si deux segments audio présentent des embeddings similaires, la synergie $\mathrm{S}(i,j)$ est alors élevée, et la pondération correspondante $\omega_{i,j}$ se renforce, permettant ainsi la formation de clusters cohérents de signaux acoustiques ou la fusion d'une modalité audio avec d'autres modalités dans un cadre multimodal.

D'un point de vue **formel**, après avoir extrait les vecteurs $\mathbf{x}_i$ pour chaque entité, on définit la **mesure** de similarité par exemple par la formule gaussienne

$$\mathrm{S}(i,j) = \exp\big(-\alpha \parallel \mathbf{x}_i - \mathbf{x}_j \parallel\big)$$

ou par la formule du **produit scalaire normalisé**

$$\mathrm{S}(i,j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\parallel \mathbf{x}_i \parallel \parallel \mathbf{x}_j \parallel}.$$

Lorsque ces scores de similarité sont intégrés dans la mise à jour

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[\mathrm{S}(i,j) - \tau\, \omega_{i,j}(t)\big],$$

le DSL renforce les liens entre les entités dont les embeddings sont proches dans l'espace, ce qui conduit à la formation de **clusters** auto-organisés reflétant la **cohérence** des caractéristiques sous-jacentes.

La **force** de cette approche réside dans la capacité des **embeddings** générés par ces modèles neuronaux à capturer une **richesse sémantique** et des aspects contextuels qui permettent d'identifier avec précision les similitudes entre entités issues de domaines variés (vision, langage,

acoustique). Cependant, le succès de cette méthode dépend fortement de la **qualité** des embeddings, laquelle doit être suffisamment robuste pour résister au bruit et capable de généraliser les caractéristiques essentielles des données. Ces principes, approfondis dans le Chapitre 3, soulignent l'importance de disposer de représentations vectorielles de haute qualité pour garantir une auto-organisation efficace au sein du DSL.

3.3.1.2. **Avantages : Calcul Aisé de Similarité (Cosinus, Distance Euclidienne)**

Dans le cadre du **Deep Synergy Learning (DSL)**, chaque entité $\mathcal{E}_i$ est représentée par un **vecteur** $\mathbf{x}_i \in \mathbb{R}^d$ qui est issu d'un modèle d'apprentissage profond tel qu'un CNN, un transformeur pour le langage ou encore un modèle audio pour les spectrogrammes. L'un des atouts majeurs de cette représentation sub-symbolique réside dans la **simplicité** algorithmique avec laquelle il est possible de calculer la **similarité** ou la **distance** entre ces vecteurs. Cette facilité de calcul joue un rôle central dans la formation et l'auto-organisation du **Synergistic Connection Network (SCN)**, puisque la **synergie** $\mathrm{S}(i,j)$ entre deux entités est directement fonction de la proximité mesurée dans l'espace vectoriel, et plus cette proximité est élevée, plus la **pondération** $\omega_{i,j}$ tend à être renforcée selon la règle de mise à jour du SCN.

D'un point de vue mathématique, la **similarité cosinus** est définie par

$$\mathrm{S}_{\cos}(i,j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\| \mathbf{x}_i \| \, \| \mathbf{x}_j \|},$$

cette mesure se concentrant sur l'**angle** entre $\mathbf{x}_i$ et $\mathbf{x}_j$ plutôt que sur leur norme, ce qui est particulièrement pertinent dans des contextes où la magnitude du vecteur n'est pas un indicateur de la **pertinence** ou de la **similarité**. En outre, une transformation linéaire de l'intervalle $[-1,1]$ permet d'obtenir un score de similarité dans $[0,1]$, par exemple

$$\mathrm{S}(i,j) = \frac{1 + \mathrm{S}_{\cos}(i,j)}{2},$$

ce qui garantit une interprétation directe dans le cadre de la mise à jour des pondérations. Par ailleurs, la **distance euclidienne** est définie par

$$d_{\mathrm{eucl}}(i,j) = \| \mathbf{x}_i - \mathbf{x}_j \| = \sqrt{\sum_{k=1}^{d} \left(x_{i,k} - x_{j,k}\right)^2},$$

et cette distance, en quantifiant la longueur du segment reliant $\mathbf{x}_i$ à $\mathbf{x}_j$, peut être convertie en un score de similarité par l'application d'une fonction de décroissance exponentielle, comme le montre la formule

$$\mathrm{S}_{\mathrm{gauss}}(i,j) = \exp\left(-\alpha \, \| \mathbf{x}_i - \mathbf{x}_j \|^2\right),$$

où $\alpha > 0$ est un paramètre réglable. Ces deux formules sont linéaires en la dimension d et reposent sur des opérations de multiplication–accumulation, qui sont bien adaptées aux architectures de calcul modernes telles que les GPU ou les bibliothèques optimisées en BLAS, permettant ainsi un

calcul rapide même pour de grands ensembles d'entités. La **simplicité** de ces mesures permet une implémentation efficace de la **mise à jour** des pondérations dans le SCN selon la formule

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[\mathrm{S}(i,j) - \tau\,\omega_{i,j}(t)\big],$$

ce qui conduit à un renforcement des liens entre les entités qui se rapprochent dans l'espace vectoriel. L'avantage de cette approche réside également dans le fait qu'elle permet une **interopérabilité** naturelle avec un grand nombre d'algorithmes existants, notamment ceux de **clustering** (tels que K-means, DBSCAN) et des techniques d'**approximate nearest neighbor** (comme FAISS ou Annoy), ce qui facilite l'analyse et la visualisation des regroupements obtenus via des méthodes de réduction de dimension telles que PCA, t-SNE ou UMAP. De plus, cette méthode se prête aisément à des scénarios **multimodaux** où, en projetant des données provenant de différentes sources dans un espace commun, on applique de manière uniforme la logique de calcul de la similarité, assurant ainsi une intégration homogène et une gestion efficace de la **variabilité** des données. Finalement, en ajustant les paramètres des kernels (comme α dans le cas du noyau gaussien), il est possible d'optimiser la **sensibilité** du DSL aux distances entre vecteurs, garantissant ainsi que la **dynamique** de mise à jour des pondérations reflète de manière fine les structures sous-jacentes des données.

3.5.3.3. Inconvénients : Dimension Élevée, Parfois Peu Interprétables

L'utilisation de **représentations sub-symboliques** sous forme de vecteurs d'embeddings, générés par des modèles neuronaux tels que les CNN ou les Transformers, offre indéniablement des avantages en termes de simplicité et d'efficacité du calcul de la synergie dans $\mathbb{R}^d$. Toutefois, ces méthodes présentent plusieurs inconvénients qui peuvent poser des problèmes de **scalabilité**, de **stabilité** et d'**explicabilité**. La discussion qui suit s'articule autour de quatre axes majeurs.

A. Dimension Élevée : Risques et Limitations

Les modèles neuronaux modernes produisent des embeddings dont la dimension peut atteindre plusieurs centaines voire milliers (par exemple, 512, 768 ou 1024 dimensions). Lorsque la fonction de synergie $\mathbf{S}(i,j)$ est calculée de manière naïve pour toutes les paires (i,j) au sein d'un Synergistic Connection Network (SCN) composé de n entités, le coût de calcul devient de l'ordre de $\mathrm{O}(n^2 \times d)$. Ainsi, si n est très grand – par exemple, plusieurs millions d'entités – et d élevé, le temps de calcul peut devenir prohibitif, comme l'illustre l'estimation

$$\text{Coût} \approx n^2 \times d.$$

De plus, la malédiction de la dimension peut rendre les distances moins discriminantes, puisque dans des espaces de très haute dimension, les distances entre la plupart des points tendent à converger vers une valeur similaire. Cette uniformisation des distances peut aboutir à une dynamique d'auto-organisation moins stable, avec des clusters qui se forment de manière peu distincte ou qui présentent des frontières floues. Face à ces difficultés, des approches telles que l'utilisation d'algorithmes d'**approximate nearest neighbors**, de techniques de **sparsification** ou de mécanismes de **sélection** ne traitant que les paires prometteuses sont souvent nécessaires afin de réduire la charge de calcul.

B. Opacité et Faible Interprétabilité

Les embeddings issus de réseaux neuronaux sont le produit d'opérations complexes et hiérarchiques réparties sur de multiples couches (convolutions, mécanismes d'attention, etc.). Cette complexité rend souvent l'interprétation de chaque coordonnée de l'embedding difficile. Il n'est pas aisé de déterminer si une dimension particulière de $\mathbf{x}_i$ correspond à une caractéristique précise, telle qu'une texture, une forme ou un concept lexical. De surcroît, lorsqu'on constate qu'un score de similarité élevé existe entre deux vecteurs $\mathbf{x}_i$ et $\mathbf{x}_j$, il est ardu d'attribuer de manière explicite la cause de cette proximité – qu'il s'agisse de la couleur, de la catégorie ou d'une co-occurrence textuelle. Cette opacité pose un réel problème dans des domaines exigeant une forte **explicabilité** ; ainsi, pour des applications critiques (médicales, industrielles, etc.), il est impératif de pouvoir justifier la formation d'un cluster ou la liaison entre deux entités. Même si des techniques de réduction de dimension, telles que PCA, UMAP ou t-SNE, permettent d'obtenir une vue globale des distributions, elles n'améliorent guère l'interprétation des attributs à l'échelle individuelle.

C. Instabilité des Embeddings et Variations Contextuelles

Les embeddings neuronaux peuvent varier sensiblement en fonction du **contexte** ou des modifications apportées lors de la phase d'entraînement. Dans le domaine du traitement du langage naturel, par exemple, une légère modification du corpus d'entraînement ou un nouveau fine-tuning peut entraîner des décalages dans les positions des mots ou des tokens dans l'espace vectoriel. De même, en vision par ordinateur, l'adaptation d'un réseau pour intégrer de nouvelles classes d'images peut modifier la distribution finale des embeddings $\mathbf{x}_i$. De telles variations contextuelles impactent directement la mesure de la synergie, que ce soit par la modification de la distance $\| \mathbf{x}_i - \mathbf{x}_j \|$ ou par le produit scalaire $\mathbf{x}_i \cdot \mathbf{x}_j$. Cette instabilité peut ainsi perturber la dynamique du SCN, induisant des réorganisations brutales ou imprévues dans l'auto-organisation, et posant un défi pour la stabilité à long terme du DSL.

D. Nécessité d'un Prétraitement Potentiellement Coûteux

L'obtention de ces embeddings repose sur l'utilisation de modèles préentraînés, tels que des réseaux CNN pour les images ou des Transformers pour les textes, qui nécessitent une infrastructure de calcul robuste (GPU/TPU) et des ressources importantes pour l'entraînement ou l'inférence. Si l'objectif est de concevoir un DSL autonome fonctionnant sur des ressources matérielles limitées, le coût en termes de calcul et de temps associé au prétraitement des données peut s'avérer prohibitif. De plus, le déploiement d'un tel système requiert la mise en place d'un pipeline complet, capable de fournir en continu des embeddings de haute qualité, ce qui représente une contrainte supplémentaire en termes d'ingénierie et d'infrastructure.

3.3.2. Autoencodeurs, Transformers et Approches Avancées

Les vecteurs et embeddings présentés en section 3.3.1 constituent déjà une base solide pour représenter des entités de nature variée (images, textes, sons). Toutefois, les **techniques** d'apprentissage ne cessant d'évoluer, il est possible d'aller **plus loin** en recourant à des méthodes avancées qui extraient des embeddings encore plus **expressifs** ou plus **compacts**. Dans cette section, nous verrons d'abord (3.3.2.1) comment les **autoencodeurs** peuvent servir à la **réduction dimensionnelle** ou à l'**extraction de features**, puis (3.3.2.2) comment les **Transformers** (BERT,

GPT, ViT) produisent des embeddings **contextuels**, et enfin (3.3.2.3) nous soulignerons **l'intérêt** de ces méthodes avancées pour le DSL, au prix toutefois d'une complexité de calcul plus élevée.

3.3.2.1. Autoencodeurs pour Réduction Dimensionnelle ou Extraction de Features

Dans le cadre d'un **Deep Synergy Learning (DSL)**, l'extraction d'informations pertinentes à partir de représentations sub-symboliques constitue un enjeu majeur pour l'efficacité du processus d'auto-organisation. Les **autoencodeurs (AE)** se présentent comme un outil fondamental permettant d'apprendre, de manière non supervisée, un **code latent z** qui capture l'essence d'un vecteur d'entrée **x**. Cette démarche offre deux avantages essentiels pour le DSL où elle permet la **réduction de dimension** et l'**extraction de features discriminantes**.

A. Principe et Architecture d'un Autoencodeur

Un autoencodeur se compose typiquement de deux modules complémentaires. Le premier, l'**encodeur** E, est une fonction $E: \mathbb{R}^d \to \mathbb{R}^k$ qui reçoit un vecteur d'entrée $\mathbf{x} \in \mathbb{R}^d$ et produit un **code latent** $\mathbf{z} \in \mathbb{R}^k$, où $k < d$ dans le but de compresser l'information. Le second module, le **décodeur** D, opère la transformation inverse, $D: \mathbb{R}^k \to \mathbb{R}^d$, et tente de reconstruire le vecteur d'entrée à partir de ce code latent, produisant ainsi $\hat{\mathbf{x}} = D\big(E(\mathbf{x})\big)$. L'apprentissage se réalise en minimisant une **fonction de perte de reconstruction** $\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}})$, souvent formulée comme la moyenne des erreurs quadratiques, ce qui s'exprime par

$$\min_{E,D} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}\left(\mathbf{x}_i,\ D\big(E(\mathbf{x}_i)\big)\right).$$

Cette optimisation force l'encodeur à extraire les facteurs de variation les plus pertinents de **x**, tout en éliminant le bruit et les détails superflus, aboutissant ainsi à une représentation compacte et significative.

B. Avantages : Réduction de Dimension et Extraction de Traits

L'imposition d'un goulot d'étranglement, où la dimension du code latent k est inférieure à celle de l'entrée d, engendre deux bénéfices notables pour le DSL. D'une part, la **réduction dimensionnelle** permet de diminuer significativement le coût de calcul lors de l'évaluation des distances ou similarités entre entités. En effet, le calcul de la distance entre deux codes latents, par exemple $\| \mathbf{z}_i - \mathbf{z}_j \|$, se réalise en $O(k)$ plutôt qu'en $O(d)$, ce qui est particulièrement avantageux lorsque le réseau traite un grand nombre d'entités. D'autre part, en apprenant à reconstruire **x** à partir de **z**, l'autoencodeur extrait des **features discriminantes** qui résument les aspects sémantiques les plus importants du vecteur d'origine, facilitant ainsi la formation de clusters cohérents et la détection de similarités véritables dans le SCN.

C. Variantes : Denoising, Sparse et Variational Autoencoder

Il existe plusieurs variantes d'autoencodeurs qui visent à améliorer la robustesse et la qualité des représentations latentes.
Premièrement, le **Denoising Autoencoder** introduit du bruit dans **x** durant l'entraînement et apprend à reconstruire la version nettoyée de la donnée. Ce procédé confère au code latent **z** une

meilleure tolérance aux perturbations, ce qui est particulièrement utile dans des environnements où les données sont bruitées.

Ensuite, le **Sparse Autoencoder** incorpore une contrainte de sparsité sur $\mathbf{z}$ – par exemple, via une pénalisation de la norme L_1 – de sorte que seule une fraction limitée des neurones s'active pour chaque donnée, ce qui peut améliorer l'interprétabilité des caractéristiques extraites et favoriser une séparation plus nette des classes.

Enfin, le **Variational Autoencoder (VAE)** adopte une approche probabiliste en imposant que le code latent suive une distribution prédéfinie, généralement gaussienne, c'est-à-dire $\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2 I)$. Cette formulation permet non seulement de générer de nouvelles données, mais aussi d'organiser l'espace latent de manière plus lisse, facilitant ainsi la continuité et la robustesse des clusters formés lors de l'auto-organisation.

D. Intégration au DSL : La Synergie entre Codes Latents

Une fois l'autoencodeur entraîné, la partie **encodeur** E est utilisée pour transformer chaque vecteur d'entrée $\mathbf{x}_i$ en un code latent $\mathbf{z}_i = E(\mathbf{x}_i)$ dans un espace de dimension réduite $\mathbb{R}^k$. Ce code latent remplace alors $\mathbf{x}_i$ dans la phase de calcul de la synergie au sein du DSL. La mesure de synergie peut être définie par des formules classiques telles que

$$\mathbf{S}(i,j) = \exp\left(-\alpha \parallel \mathbf{z}_i - \mathbf{z}_j \parallel^2\right)$$

ou par une mesure basée sur la similarité cosinus

$$\mathbf{S}(i,j) = \frac{\mathbf{z}_i \cdot \mathbf{z}_j}{\parallel \mathbf{z}_i \parallel \parallel \mathbf{z}_j \parallel}.$$

Ces scores de synergie, calculés dans un espace de dimension réduite, facilitent le renforcement des pondérations $\omega_{i,j}$ et la formation de clusters dans le Synergistic Connection Network. L'optimisation de la dimension k joue alors un rôle critique où si k est trop petit, l'autoencodeur risque de perdre des informations essentielles, tandis qu'un k trop élevé n'apporte pas le gain de complexité recherché.

E. Limites et Points de Vigilance

Malgré leurs avantages, les autoencodeurs présentent quelques inconvénients notables.

Premièrement, la minimisation de la **perte de reconstruction** n'est pas nécessairement alignée avec l'objectif de séparer distinctement les classes ou de maximiser la discrimination entre les entités. Par conséquent, les codes latents $\mathbf{z}_i$ peuvent parfois ne pas être suffisamment discriminants pour la tâche d'auto-organisation.

Deuxièmement, l'entraînement d'un autoencodeur sur des datasets volumineux peut être coûteux en termes de ressources computationnelles et de temps, notamment lorsqu'il s'agit d'architectures profondes.

Troisièmement, le choix de la dimension latente k est crucial où une valeur inappropriée peut soit sous-estimer les facteurs de variation essentiels, soit ne pas fournir une véritable réduction de complexité. Enfin, dans un contexte d'apprentissage continu, la mise à jour de l'encodeur E doit

être soigneusement gérée pour éviter le phénomène de **catastrophic forgetting**, qui pourrait altérer l'organisation de l'espace latent déjà établi.

3.3.2.2. Transformers (BERT, GPT, ViT) : Embeddings Contextuels plus Riches

L'émergence des **Transformers** – tels que BERT, GPT, T5 en NLP et Vision Transformer (ViT) en vision – a profondément transformé la manière dont sont générés les **embeddings** dans divers domaines. Contrairement aux représentations plus statiques (comme Word2Vec) ou aux approches purement convolutionnelles (CNN), les Transformers produisent des **embeddings contextuels**. Ces représentations intègrent non seulement l'information d'un token ou d'un patch de manière isolée, mais elles capturent également le **contexte global** par le biais d'un mécanisme de *self-attention*. Dans le cadre d'un **Deep Synergy Learning (DSL)**, cette capacité à incorporer le contexte se traduit par des mesures de synergie plus fines et expressives, où la proximité entre deux entités reflète non seulement des similarités de caractéristiques isolées, mais aussi des interactions complexes au sein de l'ensemble des données.

A. Principe des Transformers et Self-Attention

L'architecture Transformer repose sur un mécanisme de *self-attention* qui permet à chaque position d'une séquence de « regarder » toutes les autres positions. Soit une séquence d'entrées représentées par la matrice $\mathbf{x} \in \mathbb{R}^{n \times d}$, où chaque ligne correspond à un embedding initial d'un token ou d'un patch. Les Transformers calculent trois matrices essentielles par multiplication avec des poids appris :

$$\mathbf{Q} = \mathbf{x}\, W_Q, \quad \mathbf{K} = \mathbf{x}\, W_K, \quad \mathbf{V} = \mathbf{x}\, W_V,$$

où W_Q, W_K, et W_V sont des matrices de projection de dimensions appropriées (souvent $d \times d_k$). Le mécanisme de self-attention se définit alors par l'opération

$$\text{Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\,\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}.$$

Cette opération permet à chaque token (ou patch) de pondérer les contributions de tous les autres éléments en fonction de la similitude entre ses **requêtes** et les **clés** associées aux autres tokens, produisant ainsi des embeddings qui intègrent l'information contextuelle de l'ensemble de la séquence.

B. BERT, GPT : Embeddings Contextuels en NLP

Dans le domaine du **Traitement Automatique du Langage Naturel (NLP)**, des modèles tels que **BERT** et **GPT** exploitent pleinement le mécanisme de self-attention pour produire des **embeddings contextuels**. Concrètement, après plusieurs couches de self-attention et de transformations non linéaires, chaque token i de la séquence obtient une représentation finale $\mathbf{h}_i^{(L)}$. Deux types d'extractions sont couramment utilisées :

- **Token-level** : Chaque mot ou sous-mot se voit attribuer un embedding $\mathbf{h}_i$ qui capture ses nuances contextuelles.

- **Embedding global** : Un vecteur représentatif de l'ensemble de la séquence est généré, souvent en utilisant le token spécial $[CLS]$ (dans le cas de BERT) ou par agrégation (moyenne ou max) sur tous les tokens.

Dans un DSL, ces embeddings globaux servent de « signature » vectorielle pour une entité textuelle. La synergie entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ peut ainsi être évaluée par des mesures telles que la similarité cosinus :

$$\mathrm{S}(i,j) = \frac{\mathbf{z}_i \cdot \mathbf{z}_j}{\| \mathbf{z}_i \| \, \| \mathbf{z}_j \|},$$

où $\mathbf{z}_i$ et $\mathbf{z}_j$ sont les embeddings finaux issus du Transformer. Grâce à la prise en compte du contexte, ces représentations permettent de discerner des relations sémantiques fines, telles que la synonymie contextuelle ou les paraphrases, améliorant ainsi la qualité de l'auto-organisation du réseau.

C. Vision Transformer (ViT) : Patches et Self-Attention en Vision

Les principes du Transformer se sont également appliqués au domaine de la vision par ordinateur via le **Vision Transformer (ViT)**. Dans ce cas, une image est divisée en petits **patches** (par exemple, de 16×16 pixels), qui sont linéarisés et projetés dans un espace de dimension d. Pour conserver l'information spatiale, des **positional embeddings** sont ajoutés à chaque patch. Le Transformer traite alors l'ensemble des patches de manière similaire aux tokens en NLP, utilisant le mécanisme de self-attention pour capturer des relations globales dans l'image. La sortie finale peut être constituée d'un token spécial $[CLS]$ agissant comme représentation globale ou d'une agrégation des embeddings individuels. Cette approche permet de générer des embeddings d'images qui intègrent non seulement des caractéristiques locales, mais aussi des dépendances à longue portée entre différentes régions de l'image, améliorant ainsi la mesure de la synergie dans le DSL.

D. Enjeux et Coûts : Dimension et Ressources

Bien que les Transformers produisent des embeddings contextuels particulièrement riches, ils impliquent également des coûts importants en termes de **dimension** et de **ressources**. Par exemple, les modèles tels que BERT ou GPT génèrent des vecteurs de dimension 768, 1024 ou plus, ce qui peut amplifier la complexité des calculs de similarité dans un SCN, souvent de l'ordre de $\mathrm{O}(n^2 \times d)$ pour n entités. De plus, le poids des modèles (parfois plusieurs centaines de millions de paramètres) requiert une infrastructure matérielle (GPU, TPU) conséquente pour l'inférence et le fine-tuning. Enfin, la sensibilité des Transformers aux ajustements fins – un léger changement de paramètres peut modifier la distribution des embeddings – nécessite des techniques d'alignement ou de versionnage pour préserver la stabilité des représentations dans un système d'apprentissage continu.

E. Intégration dans un DSL

L'intégration des Transformers dans un DSL se réalise en extrayant, pour chaque entité $\mathcal{E}_i$ (qu'il s'agisse d'un document, d'une image ou d'un segment audio), un embedding contextuel $\mathbf{z}_i$ via une opération telle que

$$\mathbf{z}_i = \mathrm{TransformerEncode}(\mathbf{x}_i),$$

où $\mathbf{x}_i$ représente l'objet brut initial et TransformerEncode désigne le processus de passage par plusieurs couches de self-attention et de feed-forward. Une fois obtenu, l'embedding peut être normalisé afin de faciliter le calcul de similarités, par exemple en utilisant le cosinus. La synergie entre deux entités est alors définie par une fonction f (comme le cosinus ou un noyau RBF) :

$$\mathrm{S}(i,j) = f\big(\mathbf{z}_i, \mathbf{z}_j\big).$$

Dans le processus de mise à jour des pondérations du SCN,

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[\mathrm{S}(i,j) - \tau\,\omega_{i,j}(t)\big],$$

une synergie élevée entre des embeddings contextuels conduira au renforcement des liens, favorisant ainsi la formation de clusters d'entités sémantiquement cohérents, tout en capturant des relations complexes que seules des approches traditionnelles auraient du mal à discerner.

3.3.2.3. **Intérêt pour le DSL : un Embedding plus Robuste, mais plus Complexe à Calculer ou à Mettre à Jour**

Dans le cadre d'un **Deep Synergy Learning (DSL)**, l'emploi de techniques avancées telles que les **Transformers** ou les **autoencodeurs profonds** permet d'obtenir des **embeddings** d'une qualité remarquable, caractérisés par une richesse contextuelle et une capacité de discrimination supérieure. Ces représentations, issues de modèles sophistiqués, se distinguent par leur aptitude à capter des nuances sémantiques fines et à structurer l'espace latent de manière à favoriser la formation de clusters d'entités plus robustes. On définit notamment, par une fonction g, l'embedding contextuel d'une entité initiale $\mathbf{x}_i$ par

$$\mathbf{z}_i = g(\mathbf{x}_i),$$

ce qui permet de mesurer la **synergie** entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ à l'aide d'une distance ou d'un produit scalaire, par exemple

$$\mathrm{S}(i,j) = \exp\big(-\alpha \parallel \mathbf{z}_i - \mathbf{z}_j \parallel^2\big) \quad \text{ou} \quad \mathrm{S}(i,j) = \frac{\mathbf{z}_i \cdot \mathbf{z}_j}{\parallel \mathbf{z}_i \parallel \parallel \mathbf{z}_j \parallel},$$

ce qui permet de regrouper les entités ayant des caractéristiques latentes similaires au sein du **Synergistic Connection Network (SCN)**. Ces méthodes confèrent au DSL une **robustesse** accrue dans la capture des variations contextuelles et une capacité à distinguer subtilement des différences sémantiques que des approches plus simples auraient tendance à négliger.

Cependant, la qualité supérieure des embeddings contextuels se traduit par une **complexité** notable tant dans le calcul de l'inférence que dans la gestion des mises à jour de ces représentations. Le modèle g, en intégrant plusieurs couches de self-attention et de transformations non linéaires, nécessite en général une infrastructure matérielle puissante, telle que des GPU ou TPU, en raison du nombre élevé de paramètres et de la dimension des vecteurs, souvent de l'ordre de 768, 1024 ou plus. Par conséquent, le coût de calcul d'une opération d'inférence, qui peut être évalué en $\mathrm{O}(n^2 \times d_{\text{latent}})$ pour un ensemble de n entités, augmente considérablement, en particulier lorsque le DSL doit traiter de grands volumes de données en temps réel. Cette complexité algorithmique se trouve également exacerbée par le coût associé à la mise à jour des embeddings dans un contexte

d'apprentissage continu. En effet, si le modèle g subit un fine-tuning ou une révision partielle, il devient nécessaire de recalculer les embeddings pour toutes les entités, c'est-à-dire

$$\mathbf{z}_i(t+1) = g_{t+1}(\mathbf{x}_i) \quad \text{pour } i = 1, \dots, n,$$

ce qui, pour des ensembles de données de grande taille, peut engendrer une latence non négligeable et provoquer des fluctuations dans les valeurs de $\mathrm{S}(i,j)$ entre les cycles de mise à jour. Ces variations, même modestes, peuvent induire des réorganisations brutales au sein du SCN, compromettant ainsi la stabilité de l'auto-organisation.

L'ingénierie d'un DSL doit donc établir un compromis entre la **richesse** des embeddings, qui améliore la précision des regroupements en capturant des informations contextuelles complexes, et l'**efficacité computationnelle** requise pour leur calcul et leur mise à jour régulière. Dans certains contextes critiques, où la qualité de la représentation est primordiale – par exemple en médecine ou en analyse de données sensibles – il peut être acceptable de supporter des coûts de calcul élevés pour garantir une **précision** et une **robustesse** accrues. À l'inverse, dans des applications massives où la rapidité d'inférence est essentielle, des modèles plus légers ou des embeddings de dimension réduite pourraient être privilégiés afin d'optimiser la scalabilité du système.

L'utilisation d'embeddings plus riches, issus de techniques avancées, permet d'améliorer significativement la **discrimination** et la **robustesse** des synergies dans un DSL, mais se fait au prix d'une complexité computationnelle accrue et d'un défi supplémentaire en termes de mise à jour des représentations dans un cadre d'apprentissage continu. Ce compromis entre la richesse des représentations et l'efficacité opérationnelle constitue un enjeu central dans la conception et l'implémentation de systèmes de Deep Synergy Learning.

3.3.3. Calcul de la Synergie entre Vecteurs

Dans les sections précédentes (3.3.1 et 3.3.2), nous avons mis en évidence l'usage de **vecteurs** (embeddings) pour représenter des entités dans un contexte sub-symbolique, ainsi que l'apport de méthodes avancées (autoencodeurs, Transformers) pour produire des embeddings plus riches. Il reste à voir **comment**, à partir de ces vecteurs, on définit la **synergie** $S(i,j)$ entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$. En pratique, le DSL (Deep Synergy Learning) recourt souvent à des **fonctions** de distance ou de similarité géométriques. Cette section (3.3.3) se divise en trois points :

- (3.3.3.1) présentation des **distances usuelles** (euclidienne, manhattan) et de la **similarité cosinus**,
- (3.3.3.2) usage de **kernels** pour une similarité non linéaire,
- (3.3.3.3) **ajustements** (normalisation, calibration) pour mieux gérer le bruit ou l'échelle des vecteurs.

3.3.3.1. **Distances Usuelles : Euclidienne, Manhattan, Cosinus**

Dans le cadre du **Deep Synergy Learning (DSL)**, chaque entité $\mathcal{E}_i$ est représentée par un **vecteur** $\mathbf{x}_i \in \mathbb{R}^d$. L'évaluation de la **synergie** entre deux entités repose alors sur la comparaison de ces représentations via une fonction de **distance** ou de **similarité**. Ces mesures jouent un rôle fondamental dans le processus d'**auto-organisation** du **Synergistic Connection Network (SCN)**, puisque plus deux vecteurs sont « proches » dans l'espace, plus la **pondération** $\omega_{i,j}$ tend à être renforcée. Parmi les approches classiques pour mesurer cette proximité, on trouve la **distance euclidienne**, la **distance manhattan** et la **similarité cosinus**, chacune offrant une interprétation géométrique différente et répondant à des exigences spécifiques selon le type de données traitées.

*A. **Distance Euclidienne***

La **distance euclidienne** constitue la mesure la plus intuitive, puisqu'elle correspond à la longueur du segment reliant les points représentés par $\mathbf{x}_i$ et $\mathbf{x}_j$ dans $\mathbb{R}^d$. Mathématiquement, elle se définit par

$$d_{\text{eucl}}(i,j) = \| \mathbf{x}_i - \mathbf{x}_j \| = \sqrt{\sum_{k=1}^{d} \left(x_{i,k} - x_{j,k}\right)^2}.$$

La valeur ainsi obtenue représente une notion de **distance** physique dans l'espace, et dans le DSL, cette distance est souvent convertie en un **score de similarité** afin de faciliter son intégration dans la dynamique du SCN. Une fonction d'agrégation classique consiste à appliquer une décroissance exponentielle, ce qui donne

$$\mathbf{S}(i,j) = \exp\left(-\alpha \ \| \mathbf{x}_i - \mathbf{x}_j \|^2\right),$$

où $\alpha > 0$ est un **paramètre** qui ajuste la pente de décroissance. Dans ce contexte, une distance faible entre $\mathbf{x}_i$ et $\mathbf{x}_j$ conduit à un score proche de 1, indiquant une **affinité élevée** entre les deux entités et justifiant ainsi le renforcement de la pondération $\omega_{i,j}$.

*B. **Distance Manhattan (ou ℓ_1)***

La **distance manhattan** ou distance **taxicab** s'exprime en prenant la somme des différences absolues entre les composantes correspondantes des vecteurs. La formule est donnée par

$$d_{\text{manh}}(i,j) = \sum_{k=1}^{d} \left|x_{i,k} - x_{j,k}\right|.$$

Cette mesure, qui évalue la différence cumulée par dimension, est particulièrement utile lorsque les données se combinent de façon additive, et elle présente l'avantage de se montrer souvent plus **robuste** face aux valeurs extrêmes de certaines coordonnées. Pour intégrer cette distance dans un DSL, on peut transformer la mesure en un score de similarité en appliquant par exemple la fonction exponentielle

$$\mathbf{S}(i,j) = \exp\left(-\beta \, d_{\text{manh}}(i,j)\right),$$

ou encore en utilisant la forme rationnelle

$$\mathbf{S}(i,j) = \frac{1}{1 + d_{\text{manh}}(i,j)},$$

où $\beta > 0$ est un **paramètre** qui ajuste le taux de décroissance. De cette manière, une distance manhattan réduite se traduit par un score de similarité élevé, indiquant une forte **affinité** entre les entités.

C. Similarité Cosinus

La **similarité cosinus** se distingue des mesures de distance classiques en se focalisant sur l'**angle** entre deux vecteurs plutôt que sur leur distance absolue. Elle est définie par

$$\mathbf{S}_{\cos}(i,j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\| \mathbf{x}_i \| \, \| \mathbf{x}_j \|},$$

et prend des valeurs dans l'intervalle $[-1,1]$. Cette mesure atteint la valeur 1 lorsque les vecteurs sont colinéaires dans la même direction, -1 lorsqu'ils sont opposés et 0 lorsque les vecteurs sont orthogonaux. Dans le cadre du DSL, il est souvent souhaitable d'obtenir un score strictement positif, et pour ce faire, on peut transformer l'intervalle en appliquant par exemple la formule

$$\mathbf{S}(i,j) = \frac{1 + \mathbf{S}_{\cos}(i,j)}{2},$$

qui convertit les valeurs de $[-1,1]$ en $[0,1]$. Cette méthode permet de neutraliser l'effet de la norme des vecteurs, en se concentrant uniquement sur leur **direction**, ce qui est particulièrement utile pour des données textuelles ou pour des représentations où la magnitude peut varier de manière significative.

D. Liaison avec la Synergie dans le DSL

L'ensemble des mesures présentées permet de définir la **synergie** entre deux entités, laquelle est utilisée pour ajuster la **pondération** $\omega_{i,j}$ dans le SCN. La règle de mise à jour se formalise par

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[\mathbf{S}(i,j) - \tau\, \omega_{i,j}(t)\big],$$

où $\eta > 0$ est le **taux d'apprentissage** et $\tau > 0$ représente le **coefficient de décroissance**. Ce mécanisme d'auto-organisation implique que plus deux entités sont évaluées comme étant proches (ou similaires) selon la mesure choisie, plus la pondération $\omega_{i,j}$ tend à augmenter, favorisant ainsi la formation de **clusters** cohérents. Le choix entre la **distance euclidienne**, la **distance manhattan** ou la **similarité cosinus** dépend du type de données et des propriétés spécifiques recherchées, car chacune offre des avantages distincts. La distance euclidienne fournit une mesure classique de la proximité, la distance manhattan peut mieux gérer certaines disparités dimensionnelles et la similarité cosinus permet de se focaliser sur l'orientation des vecteurs indépendamment de leur norme.

3.3.3.2. Kernels (RBF, Polynomial) pour une Similarité non Linéaire

Les **distances** usuelles (euclidienne, manhattan) ou la **similarité** cosinus (section 3.3.3.1) considèrent des relations souvent **linéaires** ou du moins directes dans l'espace vectoriel. Cependant, il arrive que la **relation** réelle entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ soit plus subtile, nécessitant des méthodes **non linéaires** pour en rendre compte. Dans un **Deep Synergy Learning (DSL)**, recourir à des **kernels** (noyaux) permet d'introduire des notions de **similarité** bien plus riches que les simples distances ℓ_p. Les **RBF kernels** (noyaux gaussiens) et **kernels polynomiaux** figurent parmi les plus répandus et se traduisent en un **score** $\mathrm{S}(i,j)$ apte à révéler des liens complexes cachés dans l'espace initial $\mathbb{R}^d$.

A. Principe Général des Kernels

Un **kernel** k entre deux vecteurs $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^d$ se définit comme une **fonction** :

$$k\colon\ \mathbb{R}^d \times \mathbb{R}^d \ \rightarrow\ \mathbb{R},$$

renvoyant un score (positif ou non) quantifiant leur **affinité**. La **particularité** du kernel trick, issu de la théorie des **noyaux de Mercer**, réside dans le fait que $k(\mathbf{x}_i, \mathbf{x}_j)$ peut être interprété comme un **produit scalaire** dans un espace de dimension éventuellement infinie :

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle_{\mathcal{H}},$$

où $\phi\colon \mathbb{R}^d \rightarrow \mathcal{H}$ est une **transformation** non linéaire, et $\mathcal{H}$ un **espace** (souvent de Hilbert) de dimension potentiellement énorme. La **magie** du kernel trick est qu'on n'a pas besoin de **calculer** explicitement ϕ. Il suffit d'évaluer $k(\mathbf{x}_i, \mathbf{x}_j)$. D'un point de vue **DSL**, cela équivaut à définir la **synergie** $\mathrm{S}(i,j)$:

$$\mathrm{S}(i,j) = k(\mathbf{x}_i, \mathbf{x}_j).$$

Deux entités peuvent se trouver "lointaines" selon une métrique linéaire, tout en étant **fortement similaires** en termes de transformations non linéaires. Dans un **Synergistic Connection Network**, on peut alors capter des formes de **ressemblance** plus complexes, sans explicitement projeter les entités dans un espace étendu.

B. RBF Kernel (Radial Basis Function)

Le **kernel RBF** (aussi appelé "noyau gaussien") constitue un choix très répandu. Il s'écrit :

$$k_{\mathrm{RBF}}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\gamma \ \| \mathbf{x}_i - \mathbf{x}_j \|^2\right) \quad (\gamma > 0).$$

Il s'agit d'une **exponentielle** négative de la distance euclidienne au carré. On peut l'interpréter comme une "**gaussienne**" centrée sur $\mathbf{x}_j$. Plus $\mathbf{x}_i$ est proche de $\mathbf{x}_j$, plus la valeur se rapproche de 1 ; plus ils sont éloignés, plus elle tend vers 0. Le **paramètre** γ détermine la "largeur" du noyau :

- Un γ **grand** induit une décroissance rapide de $k_{\mathrm{RBF}}(\mathbf{x}_i, \mathbf{x}_j)$ dès que $\| \mathbf{x}_i - \mathbf{x}_j \|$ grandit, ce qui focalise la similarité sur une zone très **locale**.
- Un γ **petit** rend la fonction plus "plate", donnant une vue plus **globale** où des entités modérément éloignées conservent une similarité notable.

Pour un **DSL**, un **RBF kernel** donne une **synergie** :

$$\mathrm{S}(i,j) = \exp\left(-\gamma \parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2\right).$$

Cette forme **non linéaire** permet à des entités ayant un écart euclidien significatif (dans $\mathbb{R}^d$) de tout de même afficher une similarité mesurable si elles partagent certaines **propriétés** sous-jacentes. En outre, si γ est bien paramétré, la synergie reflète de subtiles variations dans les données — plus subtiles, parfois, qu'une simple distance euclidienne.

C. Polynomial Kernel

Un autre kernel usuel est le **kernel polynomial** :

$$k_{\mathrm{poly}}(\mathbf{x}_i, \mathbf{x}_j) = \left(\mathbf{x}_i \cdot \mathbf{x}_j + c\right)^p,$$

où p est le **degré** (typiquement 2, 3, 4…) et $c \geq 0$ un terme de décalage. Cette définition incorpore toutes les **composantes** polynomiales du produit $\mathbf{x}_i \cdot \mathbf{x}_j$, allant jusqu'au degré p. Ainsi, pour $p = 2$, on inclut toutes les interactions bilinéaires $\left(x_{i,k} x_{j,\ell}\right)$. Cela peut capturer des **corrélations** plus complexes qu'un simple "angle" ou qu'une distance linéaire.

Dans un **SCN**, si on adopte $\mathrm{S}(i,j) = k_{\mathrm{poly}}(\mathbf{x}_i, \mathbf{x}_j)$, on se retrouve avec une **synergie** qui peut s'avérer très sensible à des variations dans certaines dimensions, surtout pour un p élevé. Cela peut amplifier des **propriétés** souhaitées (s'il y a de fortes corrélations polynomiales), mais risque aussi de **surexposer** le bruit ou les valeurs extrêmes. Le choix de p et c réclame donc un réglage attentif.

D. Non-Linéarité et Espace Implicite : Kernel Trick

Le **kernel trick** signifie que l'on n'a **pas** besoin de définir explicitement une transformation $\phi(\cdot)$ vers un espace de dimension (parfois énorme) $\mathcal{H}$. Il suffit de calculer $k(\mathbf{x}_i, \mathbf{x}_j)$. Dans un **DSL**, cela se traduit par une **synergie** :

$$\mathrm{S}(i,j) = k(\mathbf{x}_i, \mathbf{x}_j) \quad \text{où } k \text{ est RBF, polynomial, ou autre.}$$

Mathématiquement, deux vecteurs lointains en $\mathbb{R}^d$ peuvent devenir "proches" dans l'espace $\mathcal{H}$, si la transformation ϕ induite par le kernel met en évidence des attributs partagés. Ainsi, la **mise à jour** $\omega_{i,j}(t+1) = \omega_{i,j}(t) + \cdots$ tient compte de propriétés non linéaires. Dans la **formation** de clusters, des entités qui ne semblaient pas se ressembler au sens euclidien peuvent se rapprocher si leur **synergie** kernel s'avère élevée.

E. Exemples d'Implémentation dans un SCN

Pour un **SCN** gérant n entités, on définit $\mathrm{S}(i,j) = k(\mathbf{x}_i, \mathbf{x}_j)$. À chaque itération, calculer $\omega_{i,j}(t+1)$ exige d'évaluer ces kernels. Sur le plan **coût** :

- **RBF** : $\mathrm{O}(d)$ multiplications pour $\parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2$, puis une exponentiation.
- **Polynomial** : $\mathrm{O}(d)$ pour le produit scalaire $\mathbf{x}_i \cdot \mathbf{x}_j$, puis l'élévation au degré p.

Pour comparer toutes les paires (i,j), on fait face à $\mathrm{O}(n^2)$ évaluations, chaque en $\mathrm{O}(d)$. Si n est grand, cela devient lourd (voir chap. 3.2.2.2). En pratique, on peut élaguer en ne calculant la

synergie que sur un **voisinage** restreint (approximate nearest neighbor, etc.) ou recourir à d'autres heuristiques d'échantillonnage pour gérer la **scalabilité**.

F. Limites et Paramétrage Sensible

Le **RBF kernel** nécessite le choix d'un paramètre γ. Un γ trop grand produit une décroissance exponentielle ultra-rapide, rendant la synergie presque nulle pour la majorité des paires, ce qui peut isoler excessivement les entités. Un γ trop petit, en revanche, aboutit à un noyau très plat, de sorte que presque tout le monde se retrouve "similaire" : on aboutit à un **mégacluster**. Le **polynomial kernel** demande aussi de fixer p et c. Un p trop élevé peut amplifier les bruits, alors qu'un p trop faible (ex. 1) redescend à un comportement linéaire classique.

L'ingénieur d'un **DSL** doit donc **régler** ces hyperparamètres en fonction des données, par validation croisée ou d'autres heuristiques. Les kernels offrent une flexibilité précieuse, mais nécessitent une **phase** d'ajustement pour qu'ils produisent des **synergies** stables et cohérentes dans le SCN.

3.3.3.3. Ajustements : Normalisation, Calibration, Prise en Compte du Bruit

Dans un **Deep Synergy Learning (DSL)** où les entités $\mathcal{E}_i$ sont décrites par des **vecteurs** ou des attributs numériques, l'évaluation de la **synergie** $\mathrm{S}(i,j)$ repose sur des **distances** (euclidienne, manhattan) ou des **similarités** (cosinus, kernels). Toutefois, dans la **pratique**, il ne suffit pas de choisir une métrique : il faut aussi **calibrer** et **ajuster** la façon dont on compare les vecteurs. Les questions de normalisation, d'échelle, de saturation, ou encore de **prise en compte** du bruit sont essentielles pour garantir la **cohérence** du calcul de $\mathrm{S}(i,j)$ et la stabilité de la mise à jour $\omega_{i,j}$. Les sections qui suivent décrivent en détail comment on gère ces ajustements pour un **Synergistic Connection Network (SCN)** robuste et efficace.

A. Normalisation et Équilibrage des Vecteurs

Lorsqu'on manipule des vecteurs $\mathbf{x}_i \in \mathbb{R}^d$ pour calculer la synergie $\mathrm{S}(i,j)$, il arrive souvent que certaines **dimensions** varient sur des plages beaucoup plus larges que d'autres, ou que différentes entités $\mathcal{E}_i$ n'utilisent pas le même **ordonnancement** de valeurs. Sans précautions, la mesure de distance ou de similarité peut s'en trouver **biaisée**. Deux mécanismes de base s'appliquent :

Normalisation en norme

Un classique consiste à normaliser chaque vecteur à la **norme** 1, c'est-à-dire :

$$\mathbf{x}'_i = \frac{\mathbf{x}_i}{\| \mathbf{x}_i \|}, \quad \| \mathbf{x}'_i \| = 1.$$

Ainsi, la **similarité cosinus** entre $\mathbf{x}'_i$ et $\mathbf{x}'_j$ devient un **produit scalaire** direct. Cela annule l'influence de la **magnitude** brute et met l'accent sur l'**angle**. Dans un DSL multimodal, cette démarche évite qu'une modalité sortant des valeurs très élevées ne domine la mesure de proximité.

Normalisation par composante

On peut aussi appliquer un **z-score** (soustraction de la moyenne et division par l'écart-type), ou un min–max scaling dans l'intervalle [0,1], par dimension. Cette approche assure que chaque coordonnée contribue équitablement. Dans un **SCN**, elle prévient la situation où une composante "géante" écrase toutes les autres dans le calcul de $\| \mathbf{x}_i - \mathbf{x}_j \|$ ou $\mathbf{x}_i \cdot \mathbf{x}_j$.

En pratique, la **choix** entre ces normalisations dépend de la distribution des données. L'objectif reste de **préserver** le signal discriminant et de **limiter** l'influence de différences d'échelle, améliorant ainsi la comparabilité et la **qualité** du calcul de $\mathrm{S}(i,j)$.

B. Calibration du Score de Synergie

Une fois les vecteurs normalisés, on définit la synergie par :

$$\mathrm{S}(i,j) = \phi\left(d(\mathbf{x}_i, \mathbf{x}_j)\right) \quad \text{ou} \quad \psi\left(\mathrm{sim}(\mathbf{x}_i, \mathbf{x}_j)\right),$$

où ϕ et ψ sont des transformations (éventuellement exponentielles, sigmoïdes, inverses, etc.). Il se peut qu'on désire **calibrer** la sortie pour moduler l'amplitude. Par exemple :

Noyau gaussien

$$\mathrm{S}(i,j) = \exp\left(-\alpha \ \| \mathbf{x}_i - \mathbf{x}_j \|^2\right),$$

où $\alpha > 0$ gouverne la rapidité de décroissance. Si α est trop grand, la synergie chute quasi immédiatement pour tout écart modéré, regroupant uniquement les entités quasi identiques. Si α est trop petit, elle reste trop élevée pour un large spectre de distances, fusionnant potentiellement tout en un seul méga-cluster.

Distance inversée

$$\mathrm{S}(i,j) = \frac{1}{1 + \beta\, d(\mathbf{x}_i, \mathbf{x}_j)},$$

où $\beta > 0$. Ici, plus la distance $\| \mathbf{x}_i - \mathbf{x}_j \|$ grandit, plus S tend vers 0 mais jamais exactement 0. Le réglage de β dicte le "taux" d'atténuation.

Troncature ou saturation

On peut saturer la sortie pour ne pas dépasser 1 (ou un certain seuil), ou imposer un plancher minimal. Ainsi, la synergie reste dans $[\varepsilon, 1-\varepsilon]$ afin d'éviter un renforcement ou un affaiblissement trop extrême. Ceci peut stabiliser la mise à jour :

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\left[\mathrm{S}(i,j) - \tau\, \omega_{i,j}(t)\right].$$

Le point crucial est de **calibrer** S de façon à refléter *juste assez* la disparité entre entités, sans aboutir à des valeurs saturées pour la plupart des paires (synergie trop extrême) ou trop faibles (peu de différenciation).

C. Prise en Compte du Bruit

Dans bien des environnements (capteurs, images à faible qualité, textes "sales" ou hétérogènes), les **données** sont bruitées. Ce bruit peut fausser le calcul de distance $\| \mathbf{x}_i - \mathbf{x}_j \|$ ou de similarité $\mathbf{x}_i \cdot \mathbf{x}_j$. Plusieurs stratégies se dégagent :

Filtrage ou prétraitement. Avant tout calcul de distance, on peut nettoyer ou lisser $\mathbf{x}_i$, par exemple via du **smoothing** ou un **autoencodeur denoising**. Ce faisant, on réduit la variabilité de bas niveau, améliorant la robustesse de la synergie.

Troncature ou clipping des valeurs extrêmes. Si certaines composantes subissent des pics aberrants, on peut fixer un **cap** sur l'amplitude. Cela évite qu'une seule dimension bruitée ne domine la distance.

Injection de bruit contrôlé. Parfois, on ajoute soi-même un bruit gaussien au vecteur $\mathbf{x}_i$ (ou pendant l'apprentissage) pour **stabiliser** l'auto-organisation, un peu à la manière d'un recuit simulé. Cette démarche évite de se figer dans des minima locaux, et autorise une **exploration** plus large du SCN. Au fil du temps, on diminue ce bruit pour **affiner** la convergence.

Robustesse des fonctions de distance. Au lieu de la distance euclidienne ℓ_2, on peut employer ℓ_1 (manhattan), réputée moins sensible aux outliers sur une coordonnée. On peut également introduire des fonctions qui saturent au-delà d'une certaine différence, réduisant l'influence des valeurs extrêmes.

3.3.4. Gestion du Bruit et de l'Évolution des Embeddings

Même si les vecteurs et embeddings (section 3.3.3) constituent une base pratique pour représenter les entités dans un DSL (Deep Synergy Learning), ils ne sont pas figés. Dans la **réalité** d'un système en évolution où les capteurs changent, les nouvelles données s'intègrent et le modèle s'améliore continuellement, il faut prendre en compte le **bruit** potentiel dans les données ou dans les embeddings, pouvant entraîner des distances $\| \mathbf{x}_i - \mathbf{x}_j \|$ artificiellement élevées ou basses. Il est aussi essentiel de considérer la **mise à jour** ou le re-entraînement des modèles produisant les embeddings comme les réseaux neuronaux ou Transformers, ce qui peut modifier la représentation $\mathbf{x}_i$ au fil du temps. Enfin, les **stratégies** de filtrage, de clipping et d'ajustement local sont cruciales pour éviter la propagation d'erreurs ou la dérive progressive.

Ainsi, dans cette section, nous verrons comment le DSL peut gérer ces aspects (bruit, évolution) et préserver la **cohérence** du calcul de synergie $S(i, j)$.

3.3.4.1. **Embeddings Potentiellement Réentraînés, Fine-Tuned (Chap. 9 sur Apprentissage Continu)**

Dans le cadre du **Deep Synergy Learning (DSL)**, les **embeddings** $\mathbf{x}_i$ qui caractérisent les entités $\mathcal{E}_i$ ne sont pas considérés comme des représentations statiques et définitives. En effet, ces vecteurs, générés par des modèles neuronaux tels que des **CNN**, des **Transformers** ou des **autoencodeurs**, peuvent évoluer au fil du temps lorsque le modèle générateur est soumis à un processus de **fine-**

tuning ou à un schéma d'**apprentissage continu**. Ce phénomène de mise à jour dynamique des embeddings influence directement la **synergie** $\mathrm{S}(i,j)$ entre entités ainsi que la manière dont les **pondérations** $\omega_{i,j}$ sont ajustées dans le **Synergistic Connection Network (SCN)**.

A. Fine-Tuning et Mise à Jour des Embeddings

Considérons qu'à un instant t, chaque entité $\mathcal{E}_i$ est décrite par un **embedding** $\mathbf{x}_i(t)$ obtenu via un modèle g_t, qui s'exprime par la relation

$$\mathbf{x}_i(t) = g_t(\mathbf{d}_i),$$

où $\mathbf{d}_i$ représente les données brutes associées à l'entité, telles qu'une image, un texte ou un signal audio. Lorsqu'un processus de **fine-tuning** est initié – c'est-à-dire que le modèle g_t est réentraîné ou ajusté en réponse à de nouvelles données ou à un objectif révisé – le modèle évolue vers une nouvelle version g_{t+1} et l'embedding correspondant se met à jour selon

$$\mathbf{x}_i(t+1) = g_{t+1}(\mathbf{d}_i).$$

Même une modification modeste des poids du modèle peut entraîner une transformation significative de l'espace vectoriel, modifiant ainsi les distances $\| \mathbf{x}_i(t) - \mathbf{x}_j(t) \|$ entre entités et, par conséquent, la valeur de la **synergie** $\mathrm{S}(i,j)$. Ce réajustement induit une réorganisation du SCN, puisque la règle de mise à jour des pondérations

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[\mathrm{S}(i,j,t) - \tau\,\omega_{i,j}(t)\big],$$

dépend directement des représentations actuelles $\mathbf{x}_i(t)$ et $\mathbf{x}_j(t)$. Une telle dynamique impose une attention particulière dans la gestion des mises à jour pour ne pas perturber brusquement la structure des clusters déjà établis.

B. Apprentissage Continu et Flux Dynamique

Dans un scénario d'**apprentissage continu**, le modèle g_t est régulièrement mis à jour pour intégrer un nouveau flux de données, noté $\Delta\mathcal{D}$. La relation de mise à jour du modèle s'exprime alors par

$$g_{t+1} = \mathrm{Train}(g_t, \Delta\mathcal{D}),$$

ce qui conduit à une évolution progressive des embeddings selon

$$\mathbf{x}_i(t+1) = g_{t+1}(\mathbf{d}_i).$$

Ce mécanisme est particulièrement précieux dans des environnements non stationnaires où de nouvelles classes ou de nouvelles variations de données apparaissent de manière régulière. Toutefois, cette flexibilité s'accompagne d'une instabilité potentielle, puisque la transformation du modèle peut modifier de façon substantielle la géométrie de l'espace latent. Les distances entre les embeddings – et donc les scores de synergie – peuvent varier brutalement, induisant des réorganisations dans le SCN qui se traduisent par des fluctuations des pondérations $\omega_{i,j}$.

C. Impact sur la Mise à Jour des Pondérations

La dynamique d'auto-organisation du SCN est directement affectée par l'évolution des embeddings. La règle de mise à jour des pondérations

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\left[\mathrm{S}(i,j,t) - \tau\,\omega_{i,j}(t)\right]$$

utilise la **synergie** $\mathrm{S}(i,j,t)$, qui dépend des embeddings à l'instant t. Si, suite à un fine-tuning ou à un apprentissage continu, les embeddings se modifient de manière significative, la valeur de $\mathrm{S}(i,j,t+1)$ peut être très différente de celle précédemment obtenue. Ces variations peuvent provoquer des **sauts** ou des oscillations dans les pondérations, affectant la stabilité du regroupement des entités dans le réseau. Pour limiter ce phénomène, il est souvent recommandé d'adopter des stratégies de **lissage** dans la transition entre $\mathbf{x}_i(t)$ et $\mathbf{x}_i(t+1)$, par exemple en définissant une mise à jour pondérée telle que

$$\mathbf{x}_i(t+1) \leftarrow \alpha\,\mathbf{x}_i(t) + (1-\alpha)\,g_{t+1}(\mathbf{d}_i),$$

où le paramètre $\alpha \in [0,1]$ contrôle l'ampleur de la transition, permettant ainsi d'introduire progressivement les changements et de préserver la cohérence de la dynamique du SCN.

D. Équilibre entre Adaptation et Stabilité

L'adaptabilité des embeddings grâce au fine-tuning et à l'apprentissage continu offre une capacité essentielle à un DSL, puisqu'elle permet au système de rester pertinent face à l'évolution des données. Toutefois, cette adaptabilité se heurte à la nécessité de maintenir une stabilité dans l'organisation des entités. Un modèle qui se modifie trop fréquemment risque de déstabiliser les **clusters** existants, car les pondérations $\omega_{i,j}$ basées sur d'anciennes représentations ne seront plus en adéquation avec la nouvelle configuration de l'espace latent. Un compromis se trouve souvent dans la planification de mises à jour épisodiques, suivies de périodes de stabilisation, ou dans l'application de techniques d'adaptation incrémentale qui permettent un ajustement progressif. L'ingénieur se doit de régler la fréquence et l'ampleur des mises à jour de g_t afin de concilier la **richesse contextuelle** des nouvelles représentations et la nécessité d'une **stabilité** du SCN.

E. Gestion Concrète : Références au Chapitre 9 sur l'Apprentissage Continu

Le **chapitre 9** du présent ouvrage se penche en profondeur sur les techniques d'**apprentissage continu** et les stratégies permettant de gérer les mises à jour des embeddings sans compromettre la cohérence globale du DSL. Il y est présenté des protocoles d'adaptation incrémentale, des heuristiques de recalibrage des pondérations, ainsi que des mécanismes visant à atténuer le phénomène de **catastrophic forgetting**. Ces approches incluent notamment l'actualisation progressive des représentations, le verrouillage temporaire de certaines couches du modèle, et l'indexation régulière des embeddings afin d'assurer que la transition entre différentes versions du modèle reste la plus fluide possible.

3.3.4.2. Filtrage Local (k-NN) ou Clipping pour Limiter la Propagation d'Erreurs

Dans le cadre d'un **Deep Synergy Learning (DSL)**, la qualité et la stabilité de la **synergie** entre entités reposent sur la précision des **embeddings** et des **scores de similarité** calculés à partir de ceux-ci. Cependant, en présence de **perturbations** telles que le bruit, des valeurs aberrantes ou des variations imprévues dans les données, il devient indispensable de mettre en place des **mécanismes** visant à limiter la propagation des erreurs qui pourraient déstabiliser l'ensemble du **Synergistic Connection Network (SCN)**. Deux approches complémentaires se distinguent dans cette optique : le **filtrage local** par l'algorithme des **k-plus-proches-voisins (k-NN)** et le **clipping** des vecteurs

ou des scores de similarité. Ces stratégies, en restreignant l'influence d'entités extrêmes ou aberrantes, contribuent à préserver la stabilité globale du système.

A. Filtrage Local via k-Plus-Proches-Voisins (k-NN)

Lorsqu'une entité $\mathcal{E}_i$ est caractérisée par un **embedding** $\mathbf{x}_i \in \mathbb{R}^d$, il est fréquent, dans un SCN, de calculer la **synergie** $\mathrm{S}(i,j)$ entre toutes les paires (i,j). Cette approche, qui aboutit à un graphe complet, peut introduire des liaisons peu informatives ou même trompeuses, notamment lorsque des perturbations locales affectent certains embeddings. Afin de remédier à ce problème, il est judicieux de restreindre le calcul de la synergie à un **voisinage** immédiat, en considérant uniquement les k entités les plus proches de $\mathbf{x}_i$. Formellement, le voisinage local d'une entité se définit par

$$\mathrm{NN}_k(i) = \{j \in \{1, \dots, n\} \mid j \text{ fait partie des } k \text{ plus proches entités de } \mathbf{x}_i\}.$$

En conséquence, la fonction de synergie est tronquée de manière à ce que

$$\mathrm{S}'(i,j) = \begin{cases} \mathrm{S}(i,j), & \text{si } j \in \mathrm{NN}_k(i) \text{ ou } i \in \mathrm{NN}_k(j), \\ 0, & \text{sinon,} \end{cases}$$

ce qui signifie que seules les entités jugées suffisamment proches contribuent à la mise à jour des **pondérations** $\omega_{i,j}$. L'**avantage** de cette approche réside dans la réduction de la complexité, passant d'un nombre de comparaisons de l'ordre de $\mathrm{O}(n^2)$ à $\mathrm{O}(n\,k)$, tout en **limitant** l'impact des **outliers** qui, étant isolés, ne figurent pas dans le voisinage local. En outre, cette stratégie contribue à renforcer la **robustesse** du DSL en ne construisant des liaisons significatives qu'entre des entités réellement compatibles dans l'espace $\mathbb{R}^d$.

B. Clipping des Vecteurs et des Scores de Similarité

En complément du filtrage local, le **clipping** constitue une méthode efficace pour restreindre l'influence d'**valeurs extrêmes**. Cette technique peut s'appliquer à différents niveaux du processus d'inférence. Lorsqu'il s'agit des **embeddings** eux-mêmes, il est souvent pertinent de contraindre chaque coordonnée du vecteur $\mathbf{x}_i$ afin d'éviter que des perturbations locales ne provoquent une domination de certaines dimensions. Pour ce faire, on impose un seuil $T > 0$ et on définit le vecteur « clippé » par

$$x_{i,k}^{\mathrm{clip}} = \min\{\max\{x_{i,k}, -T\}, T\}.$$

Cette opération garantit que les valeurs de chaque dimension restent contenues dans l'intervalle $[-T, T]$, ce qui permet de préserver une **stabilité** géométrique dans $\mathbb{R}^d$. Par ailleurs, il est également possible d'appliquer le clipping directement au **score de synergie**. Si, par exemple, la fonction $\mathrm{S}(i,j)$ calcule une similarité qui pourrait parfois dépasser une borne raisonnable à cause d'un outlier, on peut définir une version clipée de cette synergie par

$$\mathrm{S}'(i,j) = \min\{\mathrm{S}(i,j), S_{\max}\},$$

où $S_{\max}$ est une borne supérieure prédéfinie. Cette approche empêche que la **synergie** n'atteigne des valeurs trop élevées qui pourraient induire un renforcement disproportionné des pondérations $\omega_{i,j}$, préservant ainsi l'équilibre global du SCN.

C. Combinaison des Approches pour une Robustesse Accrue

Dans la pratique, l'application conjointe du **filtrage local** par k-NN et du **clipping** constitue une stratégie robuste pour limiter la propagation d'erreurs. En effet, le filtrage local restreint d'abord le calcul de la synergie aux voisins les plus proches, réduisant la densité du graphe et la probabilité d'étendre l'influence d'entités aberrantes. Par la suite, le clipping, appliqué sur les vecteurs ou directement sur les scores de similarité, permet de modérer l'impact de toute valeur extrême qui pourrait encore se glisser dans le processus. Le SCN met ainsi à jour les pondérations selon la formule

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[\mathrm{S}'(i,j) - \tau\,\omega_{i,j}(t)\big],$$

où $\mathrm{S}'(i,j)$ représente la synergie obtenue après application des mécanismes de filtrage et de clipping. Cette approche intégrée permet de réduire la complexité computationnelle, en limitant le nombre de comparaisons nécessaires, tout en préservant la stabilité de l'auto-organisation en atténuant l'impact des perturbations locales.

3.3.4.3. **Exemples de Scénarios Audio-Visuels**

Dans le cadre du **Deep Synergy Learning (DSL)**, les **applications audio-visuelles** offrent un terrain d'expérimentation particulièrement riche pour illustrer la capacité du système à **auto-organiser** des entités issues de modalités diverses telles que l'image, le son et le texte. En effet, le **Synergistic Connection Network (SCN)** se doit de prendre en compte des **embeddings** extraits de flux multimodaux qui, de par leur nature, peuvent être affectés par le **bruit**, des changements de **contexte** ou des ajustements successifs induits par un fine-tuning continu. Dans cette section, nous exposons en détail deux scénarios qui démontrent comment le DSL exploite des mécanismes de limitation d'erreurs, tels que le filtrage local et le clipping, ainsi que des stratégies d'adaptation dynamique, afin de préserver une organisation robuste et cohérente dans un environnement audio-visuel.

A. Détection d'Événements Audio-Visuels

Dans un premier exemple, nous considérons la détection d'un événement, tel qu'un applaudissement ou un klaxon, qui se manifeste simultanément dans la composante visuelle et acoustique d'un flux multimodal. Pour ce faire, le flux audio-visuel est segmenté en une série de **frames vidéo** $\{\mathcal{E}_i^{(\mathrm{vid})}\}$ et en une série de **segments audio** $\{\mathcal{E}_j^{(\mathrm{aud})}\}$. Chaque frame est alors transformée en un **embedding** $\mathbf{x}_i^{(\mathrm{vid})} \in \mathbb{R}^{d_{\mathrm{vid}}}$ par un modèle de type CNN (tel que ResNet) ou par un Vision Transformer, tandis que chaque segment audio se voit attribuer un vecteur $\mathbf{x}_j^{(\mathrm{aud})} \in \mathbb{R}^{d_{\mathrm{aud}}}$ issu d'un modèle d'autoencodeur appliqué sur un spectrogramme ou d'un transformeur audio tel que wav2vec. La **synergie** entre une frame vidéo et un segment audio peut être quantifiée à l'aide d'un noyau gaussien, défini par

$$S(i,j) = \exp\Big(-\gamma \parallel \mathbf{x}_i^{(\mathrm{vid})} - \mathbf{x}_j^{(\mathrm{aud})} \parallel^2\Big),$$

où $\gamma > 0$ ajuste la sensibilité de la mesure. Toutefois, en raison des perturbations inhérentes à ce type de flux – telles que la saturation audio ou les variations lumineuses dans les images –, des

valeurs aberrantes peuvent fausser le calcul de la synergie. Pour limiter ces effets, le DSL intègre des mécanismes de **filtrage local** en restreignant le calcul de la similarité aux paires d'entités se chevauchant temporellement, ainsi que des techniques de **clipping** qui borner les valeurs extrêmes du score $S(i,j)$. Ce filtrage permet de constituer des **clusters** robustes reliant les frames et segments audio fortement synchronisés, ce qui facilite la détection d'événements multimodaux avec une grande précision.

B. Sous-Titrage Automatique à Partir d'un Flux Audio-Visuel

Dans un second scénario, le DSL est appliqué à la tâche de sous-titrage automatique, qui implique la fusion d'informations issues du flux vidéo et du flux textuel dérivé d'une transcription audio. Dans ce contexte, chaque frame vidéo $\mathcal{E}_i^{(\text{vid})}$ se voit attribuer un embedding $\mathbf{x}_i^{(\text{vid})}$ généré par un Vision Transformer ou un CNN, tandis que la transcription issue d'un module speech-to-text est convertie en un **embedding textuel** $\mathbf{x}_j^{(\text{text})} \in \mathbb{R}^{d_{\text{text}}}$ via un modèle de type BERT ou GPT. La synergie entre la représentation visuelle et celle textuelle peut être évaluée à l'aide de la **similarité cosinus**, exprimée par

$$S_{\cos}(i,j) = \frac{\mathbf{x}_i^{(\text{vid})} \cdot \mathbf{x}_j^{(\text{text})}}{\| \mathbf{x}_i^{(\text{vid})} \| \, \| \mathbf{x}_j^{(\text{text})} \|}.$$

Cette mesure permet de quantifier la proximité sémantique entre le contenu visuel et le texte associé. Cependant, des erreurs peuvent survenir dans la transcription, notamment en cas de bruit, d'accents ou de débit de parole rapide, tandis que la qualité des embeddings vidéo peut être affectée par des mouvements brusques ou des changements de scène. Afin de pallier ces imprécisions, le DSL met en place des mécanismes de **limitation temporelle** (en ne comparant que les frames proches du segment textuel) ainsi que des procédures de **clipping** pour éviter que des valeurs extrêmes ne conduisent à des scores de similarité erronés. De cette manière, le système parvient à associer de manière cohérente les segments vidéo et les extraits textuels, facilitant ainsi la génération de sous-titres ou de résumés contextuels de haute qualité.

C. Ajustement Dynamique et Fine-Tuning

Dans les deux scénarios précédents, le DSL peut être soumis à des processus de **fine-tuning** ou à des mises à jour continues, de sorte que les modèles générateurs d'embeddings (qu'il s'agisse d'un CNN, d'un Vision Transformer ou d'un modèle speech-to-text) évoluent en fonction des nouvelles données. Ainsi, les représentations $\mathbf{x}_i$ et $\mathbf{x}_j$ peuvent être réévaluées périodiquement, modifiant la synergie $S(i,j)$ calculée et, par conséquent, les pondérations $\omega_{i,j}$ dans le SCN. Afin d'éviter que ces changements brusques ne perturbent la dynamique globale du réseau, des techniques de **lissage** des transitions et de **verrouillage** partiel des modèles sont mises en œuvre. Ces stratégies visent à adapter les embeddings de manière progressive, garantissant ainsi une réorganisation continue mais stable des clusters d'entités multimodales.

3.4. Représentations Symboliques

Alors que les **représentations sub-symboliques** comme les vecteurs et embeddings dominent de nombreuses applications d'apprentissage (sections 3.2 et 3.3), il existe un autre univers de modélisation où les **représentations symboliques** occupent une place centrale. Dans ces approches, chaque entité $\mathcal{E}_i$ est décrite non plus par un vecteur, mais par un **ensemble de règles**, de **concepts** ou d'**axiomes** pouvant relever de la **logique** (règles "if… then…") ou de la **sémantique ontologique** (graphe RDF, OWL, etc.). Cette manière de représenter l'information favorise l'**interprétabilité** et le **raisonnement formel**, deux propriétés parfois difficiles à obtenir avec des embeddings "boîte noire". Toutefois, elle pose également des **défis** où il est nécessaire de maintenir la cohérence logique, de gérer difficilement le bruit et de faire face à une certaine rigidité face aux nouveaux attributs.

Dans cette section (3.4), nous étudierons d'abord (3.4.1) les **logiques, règles et ontologies** (une palette symbolique de base), puis (3.4.2) nous verrons **comment** calculer la synergie symbolique via des mesures de **compatibilité** ou de **cohérence**, et enfin (3.4.3) nous aborderons l'**évolution** d'un ensemble symbolique et la gestion des contradictions dans le temps.

3.4.1. Logiques, Règles et Ontologies

Les **représentations symboliques** couvrent un vaste champ où elles vont de la simple règle conditionnelle ("si A alors B") jusqu'aux **grandes ontologies** hiérarchisées comme OWL ou RDF, qui décrivent un univers de concepts et de relations. L'objectif est de permettre à la machine de **raisonner** ou de **valider** certaines assertions de manière déclarative.

3.4.1.1. Approche Symbolique : Ensembles de Règles (if A then B), Graphes Sémantiques, Ontologies (OWL, RDF)

Dans le cadre de l'analyse des systèmes d'**auto-organisation** et d'**apprentissage** tels que le **Deep Synergy Learning (DSL)**, il est essentiel de distinguer les approches **symboliques** des méthodes **sub-symboliques**. Alors que ces dernières représentent les entités $\mathcal{E}_i$ par des **embeddings** (obtenus par des réseaux convolutionnels, transformers, autoencodeurs, etc.) – c'est-à-dire par des vecteurs dans un espace continu $\mathbb{R}^d$ –, l'approche symbolique se fonde sur des structures explicites et logiques. Dans ce contexte, les entités sont décrites à l'aide d'**ensembles de règles** du type « *if A then B* », de **graphes sémantiques** structurés selon des triplets, ou encore d'**ontologies** formelles telles que celles définies par OWL et RDF. Ces différentes modalités de représentation offrent chacune des avantages spécifiques en termes d'**interprétabilité**, de **raisonnement formel** et de **traçabilité** de l'information.

A. Ensembles de Règles "If A then B"

L'approche la plus intuitive dans le domaine symbolique consiste à modéliser les connaissances sous forme d'implications logiques, telles que

$$\text{if } A \text{ then } B,$$

où A représente une hypothèse ou une condition initiale et B une conclusion ou action associée. Mathématiquement, une telle règle peut être exprimée par la proposition

$$A(\mathcal{E}_i) \Longrightarrow B(\mathcal{E}_i),$$

indiquant que si l'entité $\mathcal{E}_i$ satisfait l'ensemble de conditions A, alors elle doit logiquement satisfaire la conclusion B. Par exemple, dans un contexte médical, une règle du type

if fièvre et toux alors suspect infection respiratoire

permet d'associer un ensemble d'attributs booléens à une conclusion diagnostic. L'**avantage** de cette représentation est sa grande **interprétabilité** où l'humain peut aisément comprendre la justification d'une déduction, ce qui est primordial dans des domaines nécessitant une forte **traçabilité** comme la médecine ou le droit. Cependant, l'**inconvénient** majeur réside dans la **rigidité** des règles. Une légère variation dans les conditions A peut rendre la règle inapplicable, et, à grande échelle, la gestion d'un vaste ensemble de règles peut conduire à des conflits et à une explosion combinatoire difficilement maîtrisable.

B. Graphes Sémantiques

Les **graphes sémantiques** offrent une représentation plus structurée des connaissances en modélisant chaque entité $\mathcal{E}_i$ par un sous-graphe composé de nœuds (représentant des concepts ou attributs) et d'arcs (indiquant les relations entre ces concepts). Dans le standard **RDF** (Resource Description Framework), toute information est codée sous la forme de triplets (s, p, o) où s est le sujet, p le prédicat, et o l'objet. Par exemple, les triplets

(Brutus, type, Chien), (Chien, subclass-of, Mammifère), (Mammifère, subclass-of, Animal),

permettent d'établir une hiérarchie de classes. L'**avantage** des graphes sémantiques réside dans leur capacité à représenter explicitement des **hiérarchies** et à permettre des **requêtes complexes** via des langages tels que **SPARQL**. Ils facilitent ainsi la navigation et l'inférence sur les relations entre entités. Toutefois, ils présentent également des **inconvénients** où la difficulté à maintenir la **consistance** de grandes bases de connaissances et leur sensibilité aux erreurs rendent la gestion complexe, un triplet erroné pouvant perturber l'ensemble de la structure.

C. Ontologies (OWL, RDFSchema)

Les **ontologies** représentent une extension des graphes sémantiques en imposant un formalisme rigoureux pour décrire des classes, des propriétés, et des relations hiérarchiques ou restrictions (telles que des cardinalités ou des disjonctions). Par exemple, dans une ontologie OWL, on pourra formuler des axiomes tels que

$$\text{Chien} \sqsubseteq \text{Mammifère} \quad \text{et} \quad \text{hasOwner:Chien} \rightarrow \text{Humain}.$$

Dans ce cadre, un individu (par exemple, un chien nommé Brutus) sera déclaré comme une instance de la classe **Chien**, et par transitivité, comme instance de **Mammifère** et d'**Animal**. L'**avantage** majeur de cette approche est la **précision** sémantique qu'elle apporte et la capacité d'effectuer des **raisonnements automatiques** grâce à des moteurs d'inférence tels que **HermiT** ou **Fact++**. Cependant, l'**inconvénient** réside dans la complexité de la maintenance d'une ontologie de grande échelle, qui peut devenir difficile à gérer dans des environnements où les connaissances évoluent rapidement ou sont sujettes à des **bruits** et des imprécisions.

3.4.1.2. Avantages : Interprétabilité, Raisonnement Formel

Dans l'analyse des systèmes d'**auto-organisation** et d'**apprentissage** tels que le **Deep Synergy Learning (DSL)**, les approches symboliques – fondées sur des **ensembles de règles**, des **graphes sémantiques** et des **ontologies** – se distinguent nettement des méthodes sub-symboliques qui s'appuient sur des **embeddings** et des **vecteurs** numériques. Ces approches symboliques apportent une **interprétabilité** et une **capacité de raisonnement formel** qui facilitent l'explication, la vérification et l'audit des décisions prises par le système, qualités indispensables dans des domaines où la **traçabilité** et la **cohérence** des conclusions sont primordiales, tels que la médecine, le droit ou l'ingénierie.

A. Interprétabilité Explicite

Dans une représentation symbolique, chaque entité $\mathcal{E}_i$ est décrite par un ensemble explicite de **règles** logiques, d'**axiomes** ou d'**assertions** qui prennent la forme d'implications telles que

$$\text{if } A \text{ then } B,$$

où A représente l'hypothèse et B la conclusion. Par exemple, une règle de diagnostic médical peut être formulée comme suit :

$$\text{if Toux and Fièvre then InfectionRespiratoire.}$$

Ce type de formulation permet à un expert – qu'il soit médecin ou ingénieur – de **lire** facilement la logique sous-jacente et de comprendre de manière transparente **pourquoi** un ensemble de conditions mène à une conclusion particulière. Contrairement aux **embeddings** sub-symboliques, où une entité est représentée par un vecteur $\mathbf{x}_i \in \mathbb{R}^d$ dont les composantes sont souvent difficiles à interpréter, les règles symboliques exposent directement les **relations** et les **attributs** qui justifient la déduction. Dans le contexte d'un DSL, cette **lisibilité** se traduit par une **traçabilité** accrue des liens établis entre les entités. Ainsi, il est possible de lister l'ensemble des règles $\{R_i\}$ associées à chaque entité $\mathcal{E}_i$ et d'observer, de manière explicite, comment ces règles favorisent la formation ou la dissolution d'un **cluster**. Ce mécanisme offre une **confiance** renforcée dans les décisions du système, particulièrement dans des environnements critiques où il est essentiel de pouvoir expliquer les raisons d'un diagnostic ou d'une classification.

B. Raisonnement Formel et Inférences

Un autre avantage majeur des approches symboliques réside dans leur capacité à faciliter le **raisonnement formel** à l'aide de moteurs logiques tels que Prolog ou des reasoners OWL. Ces outils permettent d'effectuer des **inférences** automatiques et de vérifier la **consistance** d'un ensemble de règles. Par exemple, si l'on sait qu'« tout Mammifère a du sang chaud » et que l'on déclare que « Chien est Mammifère », il est possible de déduire automatiquement que « Chien a du sang chaud ». Dans un DSL, deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$, dotées respectivement de blocs de règles $\{R_i\}$ et $\{R_j\}$, peuvent ainsi afficher une **synergie** élevée, notée

$$\mathbf{S}_{\text{sym}}(i,j) = f\big(R_i, R_j\big),$$

si leurs ensembles de règles sont logiquement compatibles ou se complètent. La fonction f peut mesurer, par exemple, la proportion des règles communes, la non-contradiction entre les axiomes, ou encore la conséquence logique partagée par les deux entités. De ce fait, le DSL n'est pas uniquement un système de **clusterisation** basé sur des distances numériques, mais également un outil de **raisonnement**, capable de renforcer les liens entre des entités en fonction de leur **cohérence logique**.

D. Audit et Justification des Décisions

L'un des défis majeurs dans les systèmes d'auto-organisation basés sur des embeddings est le manque de transparence quant aux raisons sous-jacentes à la formation d'un cluster. Dans un DSL symbolique, il est possible d'**auditer** les décisions prises par le système en listant explicitement les règles ou axiomes partagés entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$. Par exemple, pour justifier pourquoi ces deux entités ont été regroupées, on peut présenter l'ensemble des attributs et des conditions qui se recoupent, telles que « elles appartiennent toutes deux à la classe Mammifère » ou « elles possèdent la même propriété hasSymptom('Toux') ». Ce mécanisme de **justification** renforce la crédibilité du système et facilite la prise de décision dans des contextes où il est impératif de pouvoir expliquer les processus de classification ou de diagnostic.

E. Réutilisation et Interopérabilité des Bases de Connaissances

Enfin, les approches symboliques bénéficient d'un important **avantage** en termes de **réutilisation** des ressources existantes. De nombreux domaines disposent déjà d'**ontologies** ou de **répertoires de règles** validés par des experts – tels que **Snomed-CT** et **UMLS** en médecine, **FIBO** dans la finance, ou **ISO 15926** en ingénierie. Dans un DSL symbolique, chaque entité $\mathcal{E}_i$ peut s'appuyer sur ces bases de connaissances pour hériter de classes et de propriétés prédéfinies, ce qui facilite l'intégration et l'interopérabilité avec des systèmes existants. La synergie entre les entités, notée $\mathbf{S}(i,j)$, peut ainsi bénéficier d'une sémantique partagée et validée, renforçant la cohérence globale du réseau et accélérant l'émergence de clusters pertinents.

3.4.1.3. Limites : Rigidité, Besoin de Maintenir la Cohérence Logique

Dans le cadre de la représentation symbolique au sein d'un **Deep Synergy Learning (DSL)**, l'utilisation de règles logiques, de graphes sémantiques et d'ontologies offre une grande **interprétabilité** et permet un raisonnement formel détaillé (voir section **3.4.1.2**). Cependant, ces avantages s'accompagnent de limites intrinsèques, notamment la **rigidité** inhérente aux systèmes symboliques et le besoin impératif de maintenir une **cohérence logique** à l'échelle du système. Dans cette section, nous examinons en détail ces contraintes, en explicitant leurs implications tant du point de vue théorique que pratique.

D'une part, la **rigidité** des représentations symboliques découle de leur caractère souvent **binaire** et catégorique. En effet, dans un système purement symbolique, une règle ou un axiome est formulé sous la forme d'une implication logique, par exemple :

$$\text{if } A \text{ then } B,$$

où A et B représentent respectivement des conditions et des conclusions exprimées à l'aide de prédicats ou de variables booléennes. Cette formulation impose une discontinuité où une petite

variation dans la valeur de A, par exemple une température mesurée à 37,9 °C au lieu de 38 °C, peut conduire à l'échec complet de la condition, ce qui invalide la déduction de B. Si l'on définit une fonction caractéristique $\chi_A(x)$ associée à la condition A de telle sorte que

$$\chi_A(x) = \begin{cases} 1, & \text{si } x \in A, \\ 0, & \text{sinon,} \end{cases}$$

alors une légère déviation qui ferait basculer $\chi_A(x)$ de 1 à 0 engendre une discontinuité de la fonction de déduction. Cette rigidité ne permet pas de modéliser des phénomènes progressifs ou graduels, comme le ferait une mesure sub-symbolique où les distances ou similarités se modélisent par des fonctions continues telles que la distance euclidienne ou le cosinus d'angle. Par conséquent, dans des environnements où les données sont bruitées ou incertaines, une approche symbolique stricte peut conduire à des ruptures soudaines de synergie, rendant la dynamique du DSL moins robuste.

D'autre part, la **nécessité de maintenir la cohérence logique** impose une contrainte supplémentaire dans l'architecture d'un DSL symbolique. Lorsqu'un ensemble d'entités $\{\mathcal{E}_i\}$ est caractérisé par des blocs de règles ou des ontologies, toute modification apportée à l'un de ces blocs doit être vérifiée globalement afin de préserver la consistance du système. Par exemple, considérons les axiomes suivants :

(i) $\forall x,$ if $x \in$ Mammifère then x a du sang chaud,

(ii) $\mathcal{E}_i$ est un Mammifère,

(iii) $\mathcal{E}_i$ a du sang froid.

L'introduction simultanée des axiomes (i) et (ii) conduit inévitablement à la conclusion x a du sang chaud pour $\mathcal{E}_i$, ce qui entre en contradiction avec (iii). Dès lors, la fonction de **synergie** entre les entités, que l'on peut formuler symboliquement par une fonction f telle que

$$\mathbf{S}_{\text{sym}}(i,j) = f\big(R_i, R_j\big),$$

doit être continuellement recalculée et validée afin de s'assurer que les modifications locales (par exemple, l'ajout, la suppression ou la modification d'un axiome dans R_i) ne conduisent pas à des contradictions globales. Ce processus de vérification peut être formulé à l'aide de systèmes de contraintes ou d'algorithmes d'inférence logiques, mais il ajoute une complexité algorithmique non négligeable, en particulier lorsque le nombre d'entités et de règles est très élevé. Du point de vue de la **théorie de la complexité**, la vérification de la cohérence d'une ontologie expressive relève souvent d'un problème NP-complet, voire de classes plus difficiles comme PSPACE ou EXPTIME dans certains cas (cf. OWL DL). Cela limite la **scalabilité** de l'approche symbolique dans un environnement dynamique et massivement distribué.

En outre, la mise à jour incrémentale d'un grand nombre de règles dans un DSL exige une **maintenance** continue de la base de connaissances. Chaque modification locale, par exemple l'ajout d'un nouvel axiome dans le bloc de règles R_i, peut avoir un effet domino sur l'ensemble du système. En effet, une telle modification nécessite non seulement de recalculer la synergie $\mathbf{S}_{\text{sym}}(i,j)$ entre $\mathcal{E}_i$ et toutes les autres entités $\mathcal{E}_j$, mais aussi de vérifier que cette modification ne viole pas la cohérence globale du système. En d'autres termes, la dynamique du DSL symbolique

peut être ralentie par le temps nécessaire aux opérations de validation, ce qui représente un inconvénient majeur lorsqu'on cherche à opérer des mises à jour en temps réel ou en continu.

Enfin, il faut noter que les **systèmes symboliques** purs, en raison de leur caractère binaire, ne gèrent pas naturellement un **degré de confiance** ou un **niveau de flou** dans la représentation des faits. Pour pallier ce manque, il est parfois nécessaire d'introduire des logiques floues ou probabilistes, telles que les réseaux bayésiens ou les Markov Logic Networks, ce qui complexifie encore la modélisation et l'implémentation. La **conversion** d'un problème symbolique strict en un problème flou nécessite souvent une transformation non triviale de la fonction de synergie, et par conséquent une réévaluation de toute la dynamique d'auto-organisation.

3.4.2. Calcul de la Synergie Symbolique

Lorsque deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ sont décrites par des **représentations symboliques** (ensembles de règles, ontologies, graphes sémantiques), la notion de synergie $S(i,j)$ ne repose plus sur une simple distance ou similarité vectorielle. Il s'agit plutôt d'évaluer la **compatibilité**, la **non-contradiction** ou la **cohérence** entre leurs blocs de connaissances. Dans ce paragraphe, nous évoquerons successivement :

- (3.4.2.1) la **compatibilité de règles**, le **degré de contradiction** ou la **co-occurrence** des axiomes,
- (3.4.2.2) la **similarité** entre ontologies (ou sous-ontologies),
- (3.4.2.3) des approches de **distance sémantique** plus générales, voire d'**information mutuelle symbolique**.

3.4.2.1. Compatibilité de Règles, Degré de Contradiction ou Co-Occurrence

Dans le cadre du **Deep Synergy Learning (DSL)**, lorsque les entités $\mathcal{E}_i$ sont décrites de manière symbolique – par exemple, à l'aide d'ensembles de règles, d'axiomes ou de sous-graphes ontologiques – la **synergie S**(i,j) entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ dépend essentiellement de la **compatibilité** de leurs ensembles de règles, notés respectivement R_i et R_j. La compatibilité peut être vue comme le degré de recouvrement entre les deux ensembles, mais aussi comme l'absence de contradictions logiques entre eux. Nous exposerons ci-après les différents aspects de cette démarche, en intégrant les formules mathématiques nécessaires pour formaliser ces notions.

A. Définition de la Compatibilité ou Co-Occurrence

Une première approche pour quantifier la proximité symbolique entre R_i et R_j consiste à considérer leurs ensembles comme des collections d'éléments discrets et à mesurer leur **intersection** par rapport à leur **union**. On définit ainsi la mesure de recouvrement ou de **co-occurrence** par la formule :

$$S_{\text{overlap}}(i,j) = \frac{|R_i \cap R_j|}{|R_i \cup R_j|}.$$

Cette expression renvoie une valeur comprise entre 0 et 1, où un score proche de 1 indique que les deux ensembles partagent une grande partie de leurs règles (ce qui traduit une forte **similarité conceptuelle**), tandis qu'un score proche de 0 indique un faible recouvrement. Dans le contexte d'un **Synergistic Connection Network (SCN)**, il est alors possible de relier la pondération $\omega_{i,j}$ entre deux entités à cette mesure de recouvrement. Autrement dit, lorsque R_i et R_j partagent un nombre important de règles, le système renforcera la connexion entre $\mathcal{E}_i$ et $\mathcal{E}_j$.

Cas pratique :

Prenons l'exemple de deux entités médicales. Supposons que l'entité $\mathcal{E}_i$ possède l'ensemble de règles R_i incluant des assertions telles que « if Toux and Fièvre then InfectionRespiratoire », et que $\mathcal{E}_j$ comporte un ensemble R_j similaire, par exemple incluant aussi « if Toux and Fièvre then InfectionRespiratoire » et d'autres règles liées au diagnostic d'infections respiratoires. Si $|R_i \cap R_j|$ est élevé, cela indique que les deux entités appartiennent au même domaine diagnostique, et le score $S_{\text{overlap}}(i,j)$ sera alors élevé, conduisant à un renforcement de la pondération $\omega_{i,j}$.

B. Degré de Contradiction

Outre le recouvrement, il est crucial de mesurer si les ensembles R_i et R_j présentent des **contradictions** logiques. En logique formelle, on dit qu'il y a contradiction lorsque la conjonction de certaines formules $a \in R_i$ et $b \in R_j$ conduit à une fausseté, symbolisée par $\perp$. Pour formaliser ce concept, on définit la fonction de contradiction de manière binaire par :

$$\text{Contradiction}(R_i, R_j) = \begin{cases} 1, & \text{si } \exists\, (a \in R_i,\ b \in R_j) \text{ tels que } a \wedge b \vDash \perp, \\ 0, & \text{sinon.} \end{cases}$$

Cette définition peut être affinée en comptant le nombre de conflits entre les règles, ce qui se traduit par :

$$\text{ContradictionCount}(R_i, R_j) = \sum_{(a,b) \in R_i \times R_j} \mathbf{1}\,[a \wedge b \vDash \perp],$$

où $\mathbf{1}[\cdot]$ est la fonction indicatrice qui vaut 1 si la condition est satisfaite et 0 sinon. Un nombre élevé de contradictions, c'est-à-dire une grande valeur de $\text{ContradictionCount}(R_i, R_j)$, devrait entraîner une diminution de la synergie $\mathbf{S}(i,j)$ entre les deux entités, car il n'est pas souhaitable de regrouper des entités dont les règles sont logiquement incompatibles.

C. Combinaison de Co-Occurrence et de Contradiction : Synergie Symbolique

Pour tenir compte simultanément du recouvrement et des contradictions, il convient de définir une fonction de **synergie symbolique** qui intègre à la fois un terme positif relatif à l'intersection des ensembles et un terme négatif qui pénalise les contradictions. On peut définir cette fonction par :

$$S_{\text{sym}}(i,j) = \alpha\, \frac{|R_i \cap R_j|}{|R_i \cup R_j|} - \beta\, \text{ContradictionCount}(R_i, R_j),$$

où α et β sont des constantes strictement positives qui pondèrent respectivement l'importance du recouvrement et celle du degré de contradiction. Ce score, que l'on peut ensuite contraindre à être non négatif (par exemple, en le plafonnant à zéro si nécessaire), guide la dynamique du DSL en influençant la mise à jour des pondérations selon la règle :

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\left[S_{\text{sym}}(i,j) - \tau\,\omega_{i,j}(t)\right],$$

où η est le taux d'apprentissage et τ le coefficient de décroissance. Dans cette dynamique, si $\text{ContradictionCount}(R_i, R_j)$ est élevé, le terme $S_{\text{sym}}(i,j)$ diminue, conduisant à un affaiblissement de $\omega_{i,j}$.

D. Complexité d'Inférence Logique

La mise en œuvre pratique de ces concepts pose toutefois des défis liés à la **complexité** du raisonnement logique. La vérification de la contradiction, c'est-à-dire déterminer si une conjonction $a \wedge b$ déduit $\bot$, nécessite l'emploi d'un **raisonneur** ou d'un module de déduction automatique. Dans des ontologies ou des systèmes logiques expressifs (par exemple, OWL avec une expressivité complète), la vérification de la cohérence d'un ensemble de règles peut être un problème de complexité exponentielle. Pour atténuer ce problème, il est souvent nécessaire de :

- Limiter l'expressivité du système logique en passant à des versions simplifiées telles que RDF basique ou OWL Lite.
- Utiliser des structures d'indexation ou des techniques de hashage sémantique pour repérer rapidement les conflits potentiels.
- Se concentrer sur des sous-ensembles critiques des ensembles R_i et R_j lorsqu'un recouvrement minimal est déjà détecté, afin de réduire le nombre de vérifications nécessaires.

Ces heuristiques permettent d'obtenir une approximation de la synergie symbolique qui reste **scalable** pour de grandes bases de connaissances.

E. Application dans la Mise à Jour $\omega_{i,j}$

L'ensemble des considérations ci-dessus se traduit directement dans la règle d'auto-organisation du **SCN**. En effet, la synergie symbolique $S_{\text{sym}}(i,j)$ influence la dynamique des pondérations $\omega_{i,j}$ par la mise à jour suivante :

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\left[S_{\text{sym}}(i,j) - \tau\,\omega_{i,j}(t)\right].$$

Ce schéma permet d'ajuster les liens entre entités en fonction de la compatibilité de leurs ensembles de règles. Ainsi, si deux entités partagent un grand nombre d'axiomes sans contradictions (c'est-à-dire si $S_{\text{overlap}}(i,j)$ est élevé et $\text{ContradictionCount}(R_i, R_j)$ est faible), la pondération $\omega_{i,j}$ sera renforcée. À l'inverse, la présence de contradictions entraînera une réduction de $\omega_{i,j}$, contribuant à la formation de clusters de règles cohérentes.

3.4.2.2. Similarité d'Ontologies, Mesure d'Overlap Conceptuel

Dans le cadre du **Deep Synergy Learning (DSL)**, lorsque les entités $\mathcal{E}_i$ sont décrites par des représentations symboliques, leur caractérisation repose souvent sur des ontologies formalisées selon des standards tels que OWL ou RDF. Ces ontologies définissent de manière explicite des **classes**, des **sous-classes**, des **relations** et des **axiomes** qui structurent la connaissance d'un domaine. Ainsi, chaque entité $\mathcal{E}_i$ se voit associée à un **sous-graphe ontologique** $\mathcal{G}_i$, lequel regroupe l'ensemble des triplets, assertions et axiomes pertinents à cette entité. Pour évaluer la **synergie** $\mathbf{S}(i,j)$ entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$, il convient de mesurer la similarité entre leurs sous-graphes respectifs, c'est-à-dire d'estimer l'**overlap conceptuel** entre $\mathcal{G}_i$ et $\mathcal{G}_j$. Cette section présente une approche formelle et détaillée de la mesure d'overlap conceptuel, en intégrant des formules mathématiques précises pour quantifier cette similarité.

A. Notion d'Ontologie et de Sous-Graphe

Une **ontologie** se définit comme un ensemble structuré de concepts, de relations et d'axiomes qui permettent de représenter la connaissance d'un domaine de manière formelle. Dans ce contexte, chaque entité $\mathcal{E}_i$ est associée à un sous-graphe ontologique $\mathcal{G}_i$ qui contient les triplets RDF et autres assertions telles que

$$\mathcal{E}_i \text{ rdf:type } \mathcal{C}, \quad \mathcal{C}_1 \text{ rdfs:subClassOf } \mathcal{C}_2,$$

ainsi que d'autres relations définies en OWL, comme « equivalentClass » ou « disjointWith ». Cette représentation permet d'organiser la connaissance en hiérarchies et en réseaux de relations sémantiques. Elle offre l'avantage de rendre explicite la structure de la connaissance et de faciliter le raisonnement automatique grâce à des reasoners tels que HermiT ou Fact++.

B. Mesure d'Overlap Conceptuel

1. Overlap de Triplets

La méthode la plus directe pour quantifier la similarité entre deux sous-graphes $\mathcal{G}_i$ et $\mathcal{G}_j$ consiste à mesurer le **recouvrement** de leurs triplets. Si l'on considère chaque triplet comme un élément discret, une mesure simple de co-occurrence peut être définie par l'opérateur suivant :

$$S_{\text{ont}}(i,j) = \frac{|\,\mathcal{G}_i \cap \mathcal{G}_j\,|}{|\,\mathcal{G}_i \cup \mathcal{G}_j\,|}.$$

Cette expression attribue une valeur comprise entre 0 et 1. Un score proche de 1 signifie que les deux sous-graphes partagent presque tous leurs triplets, indiquant une forte **cohérence sémantique**. À l'inverse, un score proche de 0 indique que les deux entités se définissent par des concepts et relations très différents. Ainsi, dans le cadre d'un **SCN**, la pondération $\omega_{i,j}$ peut être renforcée lorsque $S_{\text{ont}}(i,j)$ est élevé, favorisant la formation de clusters d'entités ontologiquement similaires.

2. Isomorphisme Partiel et Distance Hiérarchique

Bien que le recouvrement direct des triplets offre une première approximation de la similarité, il peut arriver que deux sous-graphes ne se recouvrent pas textuellement tout en représentant des concepts équivalents ou proches. Par exemple, deux entités peuvent utiliser des terminologies

différentes pour désigner un même concept (par ex. “Flu” et “Influenza”). Pour traiter ce cas, il est nécessaire d’introduire des mécanismes d’**isomorphisme partiel**. L’idée est d’identifier un **maximum common subgraph (MCS)** entre $\mathcal{G}_i$ et $\mathcal{G}_j$ qui tient compte des synonymes et des relations de hiérarchie telles que « rdfs:subClassOf » ou « owl:equivalentClass ». On peut alors définir un score de similarité par le rapport :

$$S_{\text{MCS}}(i,j) = \frac{|\,\text{MCS}(\mathcal{G}_i, \mathcal{G}_j)\,|}{\max(|\,\mathcal{G}_i\,|, |\,\mathcal{G}_j\,|)}.$$

Une autre approche consiste à mesurer la **distance hiérarchique** entre les concepts correspondants. Par exemple, si deux classes se trouvent à des niveaux différents dans la hiérarchie ontologique, une fonction de distance $d(\mathcal{C}_i, \mathcal{C}_j)$ peut être utilisée pour ajuster la similarité, de sorte qu’un décalage important dans la hiérarchie diminue la valeur du score de similarité.

3. Combinaison en une Fonction de Synergie Ontologique

Pour intégrer de manière globale ces notions, on peut définir la **synergie ontologique** $\mathbf{S}_{\text{ont}}(i,j)$ comme une combinaison d’un terme de recouvrement et d’un terme ajusté par la distance ou par une évaluation plus fine de l’isomorphisme partiel. Par exemple, une formule possible est :

$$\mathbf{S}_{\text{ont}}(i,j) = \alpha\, S_{\text{ont}}(i,j) + \beta\, S_{\text{MCS}}(i,j),$$

où α et β sont des coefficients de pondération qui reflètent l’importance relative du recouvrement direct et de la similarité structurelle. Dans le cas où la distance hiérarchique est également prise en compte, on peut ajuster le score par une fonction décroissante de cette distance, $g(d)$, de sorte que :

$$\mathbf{S}_{\text{ont}}(i,j) = \alpha\, S_{\text{ont}}(i,j) + \beta\, S_{\text{MCS}}(i,j) \cdot g\left(d(\mathcal{G}_i, \mathcal{G}_j)\right).$$

Cette approche permet d’obtenir une mesure de synergie qui valorise à la fois la **co-occurrence** des triplets et la similarité structurelle au sein des sous-graphes ontologiques.

C. Application dans la Dynamique du DSL

Dans un **Synergistic Connection Network**, la synergie ontologique $\mathbf{S}_{\text{ont}}(i,j)$ ainsi définie peut être intégrée directement dans la règle de mise à jour des pondérations. La mise à jour s’exprime alors par :

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\left[\mathbf{S}_{\text{ont}}(i,j) - \tau\,\omega_{i,j}(t)\right],$$

où η représente le taux d’apprentissage et τ le coefficient de décroissance. Cette dynamique permet d’ajuster les liaisons entre entités en fonction de leur **similarité conceptuelle**. Lorsque deux entités partagent un grand nombre de triplets ou présentent une forte similitude structurelle dans leurs sous-graphes ontologiques, la pondération $\omega_{i,j}$ est renforcée, favorisant ainsi l’émergence de clusters d’entités ayant des liens sémantiques forts. À l’inverse, un faible recouvrement ou une grande distance hiérarchique conduira à une diminution de $\omega_{i,j}$.

D. Problèmes de Complexité et Approximations

Il est important de noter que le calcul précis de la similarité ontologique peut devenir très coûteux en termes de complexité algorithmique, notamment lorsqu'il s'agit de trouver le maximum common subgraph (MCS) ou d'évaluer des distances hiérarchiques dans des ontologies très expressives. En pratique, pour des systèmes à grande échelle, il est souvent nécessaire de recourir à des heuristiques ou à des méthodes d'**approximate graph matching** afin de rendre le calcul de $\mathbf{S}_{\text{ont}}(i,j)$ faisable dans un temps raisonnable. Des techniques d'**indexation sémantique** ou des simplifications de l'ontologie (par exemple, en utilisant OWL Lite plutôt qu'OWL DL) peuvent également être employées pour réduire la charge computationnelle tout en conservant une précision acceptable.

Conclusion

La **mesure d'overlap conceptuel** entre deux sous-graphes ontologiques se fonde sur la quantification du recouvrement des triplets et la prise en compte d'une similarité structurelle, ajustée par la distance hiérarchique ou l'isomorphisme partiel. La synergie ontologique est ainsi définie par une fonction combinant ces aspects :

$$\mathbf{S}_{\text{ont}}(i,j) = \alpha \, \frac{|\,\mathcal{G}_i \cap \mathcal{G}_j\,|}{|\,\mathcal{G}_i \cup \mathcal{G}_j\,|} + \beta \, S_{\text{MCS}}(i,j) \cdot g\left(d(\mathcal{G}_i, \mathcal{G}_j)\right),$$

et intervient dans la dynamique d'auto-organisation du SCN via la mise à jour

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\left[\mathbf{S}_{\text{ont}}(i,j) - \tau\,\omega_{i,j}(t)\right].$$

Cette approche permet de renforcer les liens entre entités dont les représentations ontologiques se recouvrent fortement et de diminuer ceux présentant des divergences structurelles, aboutissant ainsi à des clusters sémantiquement cohérents. Les défis principaux résident dans la complexité des calculs d'inférence logique et dans la nécessité d'employer des méthodes approximatives pour assurer la **scalabilité** du système. Les sections **3.4.2** et **3.4.3** approfondiront ces compromis et présenteront des solutions hybrides pour intégrer ces mesures dans un cadre de Deep Synergy Learning.

3.4.2.3. Approches "Distance Sémantique" ou "Information Mutuelle Symbolique"

Dans le cadre du **Deep Synergy Learning (DSL)**, la capacité à mesurer la synergie entre deux entités symboliques repose non seulement sur la simple co-occurrence ou le recouvrement de règles, mais aussi sur des méthodes quantitatives permettant d'estimer de façon plus fine le degré de proximité conceptuelle. Deux approches complémentaires se distinguent ainsi où la **distance sémantique** et l'**information mutuelle symbolique** offrent des perspectives différentes mais complémentaires. Ces deux outils visent à quantifier la richesse informative partagée ou l'éloignement conceptuel entre les représentations symboliques des entités, et sont ensuite intégrés dans la dynamique d'auto-organisation du SCN (Synergistic Connection Network).

A. Approche par Distance Sémantique

Lorsqu'une entité $\mathcal{E}_i$ est représentée par un sous-graphe ontologique $\mathcal{G}_i$, l'**ontologie** fournit une structure hiérarchique reliant des concepts par des relations telles que rdfs:subClassOf ou owl:equivalentClass. Pour évaluer la **proximité sémantique** entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$, on peut définir une fonction de **distance sémantique** $d_{\text{sem}}(\mathcal{E}_i, \mathcal{E}_j)$ qui exploite la structure hiérarchique de l'ontologie. Par exemple, en identifiant pour deux concepts C_i et C_j leur **plus bas ancêtre commun** (LCS, *lowest common subsumer*), on peut mesurer la distance en nombre d'arêtes parcourues pour atteindre ce LCS. Les travaux issus de WordNet ont introduit des mesures classiques telles que la **mesure de Resnik** et la **mesure de Lin**, définies respectivement par :

$$\text{sim}_{\text{Resnik}}(C_i, C_j) = \text{IC}\left(\text{lcs}(C_i, C_j)\right),$$

$$\text{sim}_{\text{Lin}}(C_i, C_j) = \frac{2\,\text{IC}\left(\text{lcs}(C_i, C_j)\right)}{\text{IC}(C_i) + \text{IC}(C_j)},$$

où $\text{IC}(C)$ représente le **contenu informationnel** du concept C, une mesure qui augmente avec la spécificité du concept. Dans le contexte du DSL, on peut associer à chaque entité $\mathcal{E}_i$ un ou plusieurs concepts dominants extraits de son sous-graphe ontologique et définir la synergie sémantique entre $\mathcal{E}_i$ et $\mathcal{E}_j$ par :

$$\mathbf{S}_{\text{sem}}(i,j) = \text{sim}_{\text{Lin}}(C_i, C_j) \quad \text{ou} \quad \text{sim}_{\text{Resnik}}(C_i, C_j),$$

selon le choix de la mesure, de sorte que deux entités seront considérées comme sémantiquement proches lorsque leurs concepts associés présentent un fort degré d'information partagée. Une autre variante consiste à appliquer une transformation exponentielle afin de convertir la distance sémantique en une similitude, par exemple :

$$\mathbf{S}_{\text{sem}}(i,j) = \exp\left(-\alpha\, d_{\text{sem}}(\mathcal{E}_i, \mathcal{E}_j)\right),$$

où $\alpha > 0$ est un paramètre de réglage. Ce type d'approche permet une tolérance aux variations et aux synonymes, en tirant parti de la structure hiérarchique de l'ontologie.

B. Approche par Information Mutuelle Symbolique

Une alternative quant à elle consiste à adapter le concept d'**information mutuelle (MI)**, largement utilisé en théorie de l'information pour mesurer la dépendance entre deux variables aléatoires, à un contexte symbolique. Si l'on considère qu'un ensemble de règles ou d'axiomes associé à une entité $\mathcal{E}_i$ peut être représenté par un vecteur binaire $\mathbf{x}_i \in \{0,1\}^m$ — où chaque composante $x_i(k) = 1$ indique la présence de l'axiome r_k dans l'ensemble R_i de $\mathcal{E}_i$ — alors on peut définir des variables aléatoires X_i et X_j pour les entités $\mathcal{E}_i$ et $\mathcal{E}_j$. L'**information mutuelle** entre X_i et X_j se définit par :

$$\text{I}(X_i; X_j) = \sum_{\mathbf{x}\in\{0,1\}^m} \sum_{\mathbf{y}\in\{0,1\}^m} p\,(\mathbf{x}, \mathbf{y}) \log\left(\frac{p(\mathbf{x}, \mathbf{y})}{p(\mathbf{x})\,p(\mathbf{y})}\right),$$

où $p(\mathbf{x})$ et $p(\mathbf{y})$ sont les distributions marginales et $p(\mathbf{x}, \mathbf{y})$ la distribution conjointe. Une valeur élevée de $\text{I}(X_i; X_j)$ signifie qu'une connaissance du vecteur binaire associé à $\mathcal{E}_i$ réduit

significativement l'incertitude concernant celui de $\mathcal{E}_j$, indiquant ainsi une forte **synergie symbolique**. Afin de normaliser ce score dans l'intervalle $[0,1]$, on peut utiliser la formule :

$$\mathbf{S}_{\mathrm{MI}}(i,j) = \frac{\mathrm{I}(X_i; X_j)}{\max\{\mathrm{H}(X_i), \mathrm{H}(X_j)\}},$$

où $\mathrm{H}(X)$ représente l'**entropie** de la variable X. Ce score permet de capturer la part d'information partagée entre deux ensembles d'axiomes, offrant une mesure plus nuancée que le simple recouvrement.

C. Intégration dans la Dynamique du DSL

Dans un **Deep Synergy Learning (DSL)**, la synergie entre deux entités symboliques est utilisée pour adapter la pondération de leur connexion. Ainsi, la mise à jour de la pondération $\omega_{i,j}$ s'effectue généralement selon la règle :

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta \left[\mathbf{S}(i,j) - \tau\, \omega_{i,j}(t)\right],$$

où $\mathbf{S}(i,j)$ peut être défini soit comme $\mathbf{S}_{\mathrm{sem}}(i,j)$ dans le cas d'une approche par distance sémantique, soit comme $\mathbf{S}_{\mathrm{MI}}(i,j)$ dans le cadre de l'information mutuelle symbolique, ou une combinaison des deux. Par exemple, on peut poser :

$$\mathbf{S}(i,j) = \alpha\, \mathbf{S}_{\mathrm{sem}}(i,j) + \beta\, \mathbf{S}_{\mathrm{MI}}(i,j),$$

où α et β sont des coefficients d'ajustement qui déterminent l'importance relative de chaque méthode dans le calcul global de la synergie. Cette formulation permet au **SCN** de renforcer les liens entre entités dont les sous-graphes ontologiques ou les ensembles d'axiomes présentent une forte similarité sémantique et une information mutuelle élevée, tout en affaiblissant les connexions entre entités peu compatibles.

D. Problèmes de Complexité et Méthodes Approximatives

Il est essentiel de souligner que le calcul exact de la **similarité ontologique** peut s'avérer extrêmement coûteux en termes de complexité algorithmique, surtout lorsqu'il s'agit de déterminer le **maximum common subgraph (MCS)** ou d'évaluer précisément la distance hiérarchique entre concepts dans une ontologie expressive. Pour pallier ce problème, plusieurs stratégies d'approximation peuvent être mises en œuvre :

- **Simplification de l'ontologie** : en réduisant l'expressivité (par exemple en utilisant OWL Lite ou RDF basique), la complexité des inférences logiques peut être significativement diminuée.

- **Utilisation d'index sémantiques** : le recours à des structures de données de type hash ou à des index pré-calculés peut accélérer la détection des correspondances ou des conflits entre axiomes.

- **Heuristiques d'approximation** : des algorithmes d'approximate graph matching permettent d'obtenir des estimations de similarité en temps raisonnable, sans garantir l'exactitude du MCS.

Ces méthodes permettent de rendre l'intégration de la synergie ontologique dans la dynamique du DSL plus **scalable**, même si elles introduisent un compromis entre précision et efficacité.

Conclusion

La **mesure d'overlap conceptuel** entre deux sous-graphes ontologiques s'appuie sur des approches complémentaires telles que la **distance sémantique** et l'**information mutuelle symbolique**. La première exploite la structure hiérarchique des concepts pour évaluer la proximité entre deux entités, tandis que la seconde quantifie la part d'information partagée entre leurs ensembles d'axiomes. Ces deux approches, qui peuvent être combinées de manière linéaire pour former un score global

$$\mathbf{S}_{\text{ont}}(i,j) = \alpha \, \frac{|\, \mathcal{G}_i \cap \mathcal{G}_j \,|}{|\, \mathcal{G}_i \cup \mathcal{G}_j \,|} + \beta \, \mathbf{S}_{\text{MI}}(i,j),$$

s'intègrent dans la règle de mise à jour du DSL :

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta \left[\mathbf{S}_{\text{ont}}(i,j) - \tau \, \omega_{i,j}(t)\right],$$

permettant ainsi d'ajuster dynamiquement la force des connexions en fonction de la similarité sémantique. Malgré les coûts de calcul inhérents à ces méthodes d'inférence, l'usage de techniques d'approximation et d'indexation offre une voie réaliste pour appliquer ces concepts à grande échelle. Ces mesures enrichissent le **SCN** en lui conférant une dimension explicative et robuste, essentielle pour des applications où la cohérence sémantique joue un rôle crucial, notamment dans des domaines tels que la médecine, l'ingénierie ou la recherche scientifique. Les sections **3.4.2** et **3.4.3** aborderont en détail les compromis et les solutions hybrides permettant d'optimiser ces approches dans le cadre du Deep Synergy Learning.

3.4.3. Évolution Symbolique et Gestes des Contradictions

Les systèmes symboliques, bien qu'offrant une **interprétabilité** et un **raisonnement formel** (sections 3.4.1 et 3.4.2), n'échappent pas à la nécessité d'**évoluer** au fil du temps. Au même titre que les embeddings sub-symboliques (chap. 3.3.4), un bloc de connaissances symboliques doit pouvoir accueillir de **nouvelles** règles, en supprimer d'obsolètes ou de contradictoires, et gérer les conflits potentiels qui naissent de ces changements. Cette section (3.4.3) examine comment on peut **insérer** ou **supprimer** des règles (3.4.3.1), comment **détecter** et traiter les **contradictions** (3.4.3.2), et donne un **exemple** d'évolution continue d'un ensemble de postulats (3.4.3.3).

3.4.3.1. Insertion / Suppression de Règles (Chap. 9, Flux Continu)

Dans un contexte de **Deep Synergy Learning (DSL)** à composante symbolique, chaque entité $\mathcal{E}_i$ est caractérisée par un ensemble de règles ou d'axiomes, noté R_i. Ces ensembles R_i incarnent la base de connaissances qui sous-tend la représentation symbolique d'une entité. Lorsque de nouveaux faits sont découverts ou que certaines règles deviennent obsolètes, il est alors nécessaire de modifier ces blocs R_i par l'insertion ou la suppression de postulats. Cette opération de mise à jour dynamique s'inscrit dans la **dynamique** d'un **Synergistic Connection Network (SCN)** et

influence directement la **synergie symbolique** entre entités ainsi que la formation des **clusters** qui en résultent.

A. Insertion de Règles : Actualisation et Cohérence

L'insertion de nouvelles règles intervient lorsque l'on souhaite **étendre** ou **affiner** la base de connaissances associée à une entité $\mathcal{E}_i$. Formellement, si l'on considère qu'à l'instant t l'ensemble de règles associé à $\mathcal{E}_i$ est $R_i(t)$, l'ajout d'un nouveau postulat p se traduit par la mise à jour :

$$R_i(t+1) = R_i(t) \cup \{p\}.$$

Par exemple, dans un contexte médical, l'introduction de la règle « if Toux and OdeurPerdue then CovidSuspect » permet d'actualiser le diagnostic en fonction des nouvelles observations. De même, en ingénierie, l'insertion d'une règle telle que « if PièceX subtend AxisY then isRotational » formalise une relation d'articulation nouvellement identifiée.

Dès l'insertion du postulat p, la **synergie** entre l'entité $\mathcal{E}_i$ et une autre entité $\mathcal{E}_j$, notée $\mathrm{S}(i,j)$, peut être affectée. Si p renforce la compatibilité entre R_i et R_j – par exemple, en augmentant le recouvrement entre les ensembles de règles – alors la pondération $\omega_{i,j}$ se voit augmenter conformément à la dynamique d'auto-organisation donnée par la formule :

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta \left[S(i,j) - \tau\, \omega_{i,j}(t) \right].$$

Dans ce contexte, $S(i,j)$ pourra être recalculé, par exemple en utilisant une fonction symbolique $\mathrm{S}_{\mathrm{sym}}(i,j)$ qui intègre le nouvel axiome. À l'inverse, si l'ajout de p engendre une contradiction avec un axiome existant dans R_j, le score de synergie diminuera, ce qui se traduira par une baisse de $\omega_{i,j}$.

Dans un **flux continu** tel que développé au Chapitre 9, ces opérations d'insertion se produisent fréquemment et doivent être intégrées de façon à ce que la dynamique du SCN puisse s'ajuster en temps réel. Ainsi, chaque modification dans R_i entraîne un réexamen partiel – voire complet – des pondérations $\omega_{i,j}$ associées, afin de refléter l'impact des nouvelles règles sur la **compatibilité** entre entités.

B. Suppression de Règles : Réorganisation Logique

Le retrait de règles constitue l'autre facette de la mise à jour de la base de connaissances dans un DSL symbolique. Lorsque certains axiomes deviennent obsolètes ou sont invalidés par de nouvelles données, on procède à la suppression de ces éléments dans R_i. Cette opération s'exprime par :

$$R_i(t+1) = R_i(t) \backslash \{q\},$$

où q représente le postulat à retirer. Par exemple, dans le domaine médical, la règle « if Toux then Infectieux » pourra être supprimée si elle est jugée trop générale ou contredite par de nouveaux diagnostics. De même, dans un contexte industriel, l'ancienne contrainte « chargeMax = 500\,N » pourrait être écartée au profit d'une nouvelle norme.

La suppression de q affecte également la synergie $\mathrm{S}(i,j)$ entre $\mathcal{E}_i$ et d'autres entités $\mathcal{E}_j$. Si q constituait un point de recouvrement important avec R_j, sa disparition diminuera la valeur de

l'intersection $R_i \cap R_j$, et par conséquent, le score de similarité. Inversement, si q était à l'origine source de contradiction avec R_j, son retrait peut améliorer la compatibilité, entraînant une augmentation de $\omega_{i,j}$.

C. Problème de l'Échelle : Révisions en $\mathcal{O}(n)$ et Gestion de la Charge

Dans un SCN comportant un grand nombre d'entités, chaque opération d'insertion ou de suppression sur R_i peut théoriquement nécessiter une réévaluation de la synergie $\mathrm{S}(i,j)$ pour chaque entité $\mathcal{E}_j$ du réseau, ce qui représente un coût de l'ordre de $\mathcal{O}(n)$ par modification. Lorsque de telles mises à jour se produisent simultanément ou de manière fréquente, la charge de recalcul peut devenir prohibitive. Pour atténuer ce problème, deux stratégies d'optimisation peuvent être mises en œuvre :

- **Index sémantique** : En associant à chaque règle ou concept une structure de données indiquant l'ensemble des entités qui l'utilisent, il est possible de restreindre la réévaluation aux seules entités affectées par la modification. Ainsi, l'insertion ou la suppression d'un axiome p dans R_i ne nécessite de recalculer $\mathrm{S}(i,j)$ que pour les entités j qui partagent p.
- **Approche paresseuse (lazy evaluation)** : Au lieu de recalculer immédiatement la synergie pour toutes les paires affectées, la mise à jour est différée et réalisée uniquement lorsque la synergie est requise pour une opération spécifique, réduisant ainsi la fréquence des mises à jour globales.

D. Dynamique et Versionnage

Afin de gérer efficacement l'évolution des ensembles de règles, il est crucial de mettre en place un mécanisme de **versionnage**. Ce mécanisme permet de tracer l'historique des modifications et de gérer les mises à jour de manière structurée. On définit ainsi :

$$R_i^{(t+1)} = \left(R_i^{(t)} \cup R_{\mathrm{add}}(t)\right) \setminus R_{\mathrm{del}}(t),$$

où $R_{\mathrm{add}}(t)$ et $R_{\mathrm{del}}(t)$ représentent respectivement les ensembles de règles ajoutées et supprimées à l'itération t. Ce versionnage offre plusieurs avantages où il permet de conserver un historique des états $\{R_i^{(0)}, R_i^{(1)}, \dots\}$, de définir des points de contrôle pour revenir à une version antérieure en cas de conflit majeur et d'analyser l'évolution de la complexité des blocs R_i au fil du temps. Ainsi, le suivi des versions facilite la gestion des contradictions et offre une base pour des mécanismes de rollback si nécessaire.

E. Risques d'Oscillation et de Chaos Symbolique

Un système DSL symbolique, qui évolue en flux continu, peut être sujet à des phénomènes d'**oscillation** ou même de **chaos logique** si les mises à jour des règles se produisent de manière trop rapide ou contradictoire. Par exemple, si une entité $\mathcal{E}_i$ insère successivement un axiome $A \Rightarrow B$ qui contredit les règles d'une entité $\mathcal{E}_j$ et que, peu de temps après, ce même axiome est supprimé pour rétablir la compatibilité, la pondération $\omega_{i,j}$ peut fluctuer de manière excessive. De même, un changement global simultané dans plusieurs blocs R_i peut provoquer un « big bang » dans le SCN, où un grand nombre de synergies doivent être recalculées, ce qui pourrait engendrer des cycles d'instabilité. Pour pallier ces risques, il est souvent nécessaire de regrouper les modifications dans

des phases de mise à jour (batch updates) afin de permettre à la dynamique $\omega_{i,j}$ de converger partiellement entre deux vagues de changements.

Conclusion

L'insertion et la suppression de règles dans un **DSL symbolique** constituent des opérations essentielles permettant l'actualisation et l'affinement de la base de connaissances associée à chaque entité $\mathcal{E}_i$. La mise à jour de R_i se traduit par l'équation

$$R_i(t+1) = \big(R_i(t) \cup R_{\text{add}}(t)\big) \backslash R_{\text{del}}(t),$$

et entraîne une réévaluation des synergies $\mathrm{S}(i,j)$ et, par conséquent, des pondérations $\omega_{i,j}$ dans le réseau. Bien que ces opérations permettent d'adapter le SCN aux nouvelles connaissances ou à l'obsolescence de certains axiomes, elles impliquent également un coût algorithmique élevé, particulièrement à grande échelle, ainsi qu'un risque d'oscillation ou de chaos symbolique si les mises à jour ne sont pas régulées. Pour surmonter ces obstacles, il est impératif d'implémenter des stratégies d'optimisation telles que l'indexation sémantique, l'évaluation paresseuse, ainsi qu'un mécanisme de versionnage robuste. Ces stratégies permettent de limiter les révisions globales, d'assurer la cohérence logique du système, et de contrôler le risque d'instabilité lors des mises à jour en flux continu. Dans le Chapitre 9, nous approfondirons ces stratégies et heuristiques pour gérer l'apprentissage continu et la **plasticité** des règles, en combinant la **souplesse** auto-organisée du DSL avec les contraintes strictes d'un système symbolique.

3.4.3.2. Détection de Contradiction, Feedback Top-Down pour Forcer un Réagencement (Chap. 10)

Dans un système de **Deep Synergy Learning (DSL)** à composante symbolique, chaque entité $\mathcal{E}_i$ est décrite par un ensemble de règles ou d'axiomes R_i. La dynamique de mise à jour de ces blocs, comme présentée en section **3.4.3.1**, permet d'insérer ou de supprimer des postulats afin d'actualiser la base de connaissances. Toutefois, lorsque ces modifications conduisent à l'introduction de contradictions – c'est-à-dire, lorsque l'union $R_i \cup R_j$ de deux ensembles de règles permet de déduire simultanément un énoncé et son contraire – la **synergie symbolique** $\mathbf{S}(i,j)$ entre les entités concernées doit être réévaluée et, souvent, fortement pénalisée. La présente section se propose d'expliquer de manière rigoureuse comment détecter ces contradictions, de quelle manière elles impactent la dynamique du SCN, et comment un mécanisme de **feedback top-down** peut être introduit pour forcer un réagencement global et éviter des oscillations ou un chaos logique.

A. Détection de Contradiction : Fondements et Méthodes

La contradiction entre deux ensembles de règles R_i et R_j se produit lorsqu'ils, pris ensemble, engendrent une incohérence formelle, c'est-à-dire lorsque :

$$R_i \cup R_j \vdash \bot,$$

où le symbole $\bot$ représente l'absurdité ou la fausseté. Pour quantifier cette contradiction, on définit une fonction binaire de contradiction de la manière suivante :

$$\text{Contradiction}(R_i, R_j) = \begin{cases} 1, & \text{si } \exists\, a \in R_i,\ b \in R_j \text{ tels que } a \wedge b \vdash \bot, \\ 0, & \text{sinon.} \end{cases}$$

Cette définition peut être raffinée en comptant le nombre de conflits effectifs entre les deux ensembles, en posant :

$$\text{ContradictionCount}(R_i, R_j) = \sum_{(a,b) \in R_i \times R_j} \mathbf{1}\,[a \wedge b \vdash \bot],$$

où $\mathbf{1}[\cdot]$ désigne la fonction indicatrice qui vaut 1 si l'argument est vrai et 0 sinon. Ce comptage permet d'obtenir une mesure plus graduelle de l'incohérence entre R_i et R_j, laquelle pourra être intégrée dans la fonction de synergie symbolique.

En effet, si l'on considère une fonction de synergie symbolique qui combine à la fois les aspects de recouvrement et de contradiction, on peut écrire :

$$S_{\text{sym}}(i,j) = \alpha\, \frac{|R_i \cap R_j|}{|R_i \cup R_j|} - \beta\, \text{ContradictionCount}(R_i, R_j),$$

où $\alpha, \beta > 0$ sont des constantes pondératrices qui ajustent l'importance relative du recouvrement par rapport aux contradictions. Ainsi, une forte contradiction (c'est-à-dire une grande valeur de $\text{ContradictionCount}(R_i, R_j)$) fera tendre $S_{\text{sym}}(i,j)$ vers zéro, ce qui, dans la dynamique du DSL, entraînera une diminution de la pondération $\omega_{i,j}$.

B. Impact sur la Dynamique du SCN et Mise à Jour des Pondérations

La règle de mise à jour du SCN, dans sa version symbolique, est donnée par :

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\, \left[S_{\text{sym}}(i,j) - \tau\, \omega_{i,j}(t)\right],$$

où $\eta > 0$ est le taux d'apprentissage et $\tau > 0$ le coefficient de décroissance. Dès qu'une contradiction est détectée, c'est-à-dire dès que $\text{Contradiction}(R_i, R_j) = 1$ ou que $\text{ContradictionCount}(R_i, R_j)$ est élevé, le score $S_{\text{sym}}(i,j)$ est pénalisé. Cela se traduit par une réduction immédiate de la pondération $\omega_{i,j}$, isolant ainsi les entités dont les bases de connaissances sont incohérentes.

Si ces contradictions apparaissent de manière isolée, la diminution locale de $\omega_{i,j}$ peut suffire à empêcher la formation de clusters incohérents. Cependant, dans un système à grande échelle, de nombreux conflits peuvent émerger simultanément, engendrant ainsi un risque de **chaos** local ou d'oscillations persistantes dans la dynamique de mise à jour.

C. Mécanisme de Feedback Top-Down pour Forcer le Réagencement

Pour pallier le risque d'instabilité engendré par un excès de contradictions, il est nécessaire d'introduire un **feedback top-down**. Ce mécanisme agit comme un **régulateur global** qui surveille la cohérence du système et intervient lorsque le nombre de contradictions dépasse un certain seuil critique.

On peut définir une métrique globale de contradiction pour l'ensemble du SCN :

$$\text{ContradictCount} = \sum_{i,j} \text{ContradictionCount}\left(R_i, R_j\right).$$

Si cette valeur excède un seuil prédéfini T, alors le mécanisme de feedback top-down se déclenche. Ce module supérieur peut alors entreprendre plusieurs actions pour restaurer l'ordre :

- **Revue et Révision des Règles** : Le système identifie les entités $\mathcal{E}_i$ présentant le plus grand nombre de conflits (par exemple, celles pour lesquelles $\text{ContradictionCount}(R_i) = \sum_j \text{ContradictionCount}\left(R_i, R_j\right)$ est élevé) et demande une révision de leurs ensembles R_i. Cela peut impliquer la suppression ou la modification des axiomes responsables des conflits.

- **Forçage d'un Réagencement** : En imposant une baisse significative de la pondération pour les liens conflictuels, c'est-à-dire en réglant $\omega_{i,j}$ à une valeur faible ou nulle pour les paires en contradiction, le système peut isoler les entités problématiques, forçant ainsi une réorganisation des clusters.

- **Re-Clustering Global** : Dans des cas extrêmes, le méta-module peut déclencher une phase de re-clusterisation, regroupant de manière forcée les entités compatibles et séparant celles qui présentent des incohérences majeures.

Ce mécanisme de feedback top-down permet de sortir d'un régime où des modifications locales répétées – par insertion et suppression successives de règles – conduiraient à des oscillations ou à un chaos logique, et assure ainsi une **stabilité** globale du SCN.

Conclusion

La détection des contradictions logiques dans un DSL symbolique est cruciale pour maintenir la **cohérence** et la **stabilité** des synergies entre entités. Nous avons défini formellement la notion de contradiction à travers la relation

$$R_i \cup R_j \vdash \bot,$$

et avons introduit des mesures telles que $\text{ContradictionCount}\left(R_i, R_j\right)$ pour quantifier le nombre de conflits entre les ensembles de règles. En intégrant cette mesure dans la fonction de synergie symbolique

$$S_{\text{sym}}(i,j) = \alpha \frac{|R_i \cap R_j|}{|R_i \cup R_j|} - \beta\, \text{ContradictionCount}\left(R_i, R_j\right),$$

le SCN est amené à ajuster la pondération $\omega_{i,j}$ selon la compatibilité logique entre les entités. Néanmoins, lorsque de nombreuses contradictions apparaissent, le système risque de se désorganiser localement. Pour contrer ce phénomène, un mécanisme de **feedback top-down** (tel que détaillé dans le Chapitre 10) intervient pour réorganiser globalement les règles, soit en imposant la révision des axiomes conflictuels, soit en isolant les entités problématiques, soit en déclenchant une phase de re-clusterisation. Ce processus garantit que, malgré les mises à jour locales potentiellement contradictoires, le SCN demeure globalement cohérent et converge vers une structure de liens stable et explicable.

3.4.3.3. Exemple : un DSL stockant un “Ensemble de Postulats” qui se Modifie au Fil du Temps

Dans un **Deep Synergy Learning (DSL)** intégrant une composante symbolique, chaque entité $\mathcal{E}_i$ est caractérisée par un ensemble de postulats ou d’axiomes, noté R_i. Ces postulats représentent la connaissance explicite associée à l’entité et, en fonction des évolutions du domaine ou de l’arrivée de nouvelles données, R_i peut évoluer de manière dynamique par l’insertion de nouveaux postulats ou la suppression d’anciens. Cette capacité à modifier le contenu de R_i en flux continu est essentielle pour que le **Synergistic Connection Network (SCN)** puisse adapter ses liaisons, c’est-à-dire ses pondérations $\omega_{i,j}$, en temps réel.

A. Contexte : Entités “Patients” avec Blocs de Connaissances Médicales

Considérons un système médical où chaque entité $\mathcal{E}_i$ correspond à un patient et est associé à un bloc de connaissances R_i qui regroupe à la fois les observations cliniques et les règles diagnostiques pertinentes. Par exemple, pour un patient donné, R_i peut contenir des assertions telles que :

"Patient $\mathcal{E}_i$ présente Toux"

"Patient $\mathcal{E}_i$ a Fièvre(39°C)"

$R_i \ni$ {if Toux $\wedge$ Fièvre then Infectieux}

$R_i \ni$ {if Vacciné(Grippe) then RisqueRéduit}

Ces ensembles R_i forment la base symbolique qui sera utilisée pour calculer la **synergie symbolique** entre les patients. En effet, deux patients $\mathcal{E}_i$ et $\mathcal{E}_j$ dont les ensembles de postulats présentent un fort recouvrement (c'est-à-dire, un grand nombre d’axiomes communs ou compatibles) auront une synergie $\mathbf{S}(i,j)$ élevée, traduite par une pondération $\omega_{i,j}$ renforcée dans le SCN. À l’inverse, une faible similarité ou la présence de contradictions entraînera une diminution de $\omega_{i,j}$.

B. Ajout de Règles : Extension du Bloc de Connaissances

Au fil du temps, de nouvelles découvertes ou mises à jour dans le domaine médical peuvent conduire à l’ajout de nouveaux postulats dans le bloc de connaissances d’un patient. Cette opération d’insertion se formalise par :

$$R_i(t+1) = R_i(t) \cup \{p\},$$

où p représente le nouvel axiome, par exemple :

$$p\text{:if PerteOdorat } \wedge \text{ Fièvre then CovidSuspect.}$$

L’insertion de p peut modifier la synergie symbolique entre le patient $\mathcal{E}_i$ et un autre patient $\mathcal{E}_j$. Si, par exemple, R_j contient déjà une règle similaire ou compatible – telle que “if PerteOdorat ∧ Fièvre then InfectionVirale” – le recouvrement entre R_i et R_j augmente, ce qui se traduit par une augmentation de la synergie, notée $\mathbf{S}_{\text{sym}}(i,j)$. Ainsi, la mise à jour de la pondération dans le SCN s’effectue selon la formule habituelle :

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[\mathbf{S}_{\text{sym}}(i,j) - \tau\,\omega_{i,j}(t)\big].$$

En revanche, si l'axiome p entre en contradiction avec des règles déjà présentes dans R_j, par exemple si R_j contient « if Fièvre ∧ PerteOdorat then NonInfectieux », alors la fonction de synergie se verra pénalisée, et $\omega_{i,j}$ diminuera en conséquence.

C. Suppression ou Révision de Règles : Retrait de Postulats

La suppression de règles intervient lorsque certains axiomes se révèlent obsolètes ou erronés. On définit alors l'opération de suppression par :

$$R_i(t+1) = R_i(t) \backslash \{q\},$$

où q est l'axiome retiré, par exemple une règle trop générale comme :

q:if Toux then Infectieux.

Le retrait de q peut modifier la synergie de deux entités. Si q contribuait à une co-occurrence importante entre R_i et R_j, sa suppression pourrait entraîner une diminution de $\mathbf{S}_{\text{sym}}(i,j)$ et, par la suite, une baisse de la pondération $\omega_{i,j}$. Inversement, si q était à l'origine d'une contradiction avec R_j, son retrait pourrait améliorer la compatibilité et renforcer $\omega_{i,j}$. La dynamique de mise à jour reste donc régie par :

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\left[\mathbf{S}_{\text{sym}}\left(R_i(t+1), R_j(t+1)\right) - \tau\,\omega_{i,j}(t)\right].$$

Cette équation illustre que toute modification dans le bloc R_i nécessite une réévaluation immédiate de la synergie avec les autres entités, afin de répercuter l'évolution de la base de connaissances dans la structure du SCN.

D. Gestion Continue et Versionnage

Dans un environnement en flux continu, il est indispensable de suivre l'évolution des blocs de règles afin de pouvoir revenir sur des changements si nécessaire et de garantir la cohérence du système. Pour ce faire, on introduit un mécanisme de **versionnage** :

$$R_i^{(t+1)} = \left(R_i^{(t)} \cup R_{\text{add}}^{(i,t)}\right) \backslash R_{\text{del}}^{(i,t)},$$

où $R_{\text{add}}^{(i,t)}$ représente l'ensemble des règles ajoutées à l'itération t et $R_{\text{del}}^{(i,t)}$ celles supprimées. Ce mécanisme permet de conserver un historique des états $\{R_i^{(0)}, R_i^{(1)}, \dots\}$, facilitant ainsi l'analyse rétroactive des décisions prises et, en cas de conflits majeurs, la possibilité d'un rollback. Un tel versionnage s'avère essentiel pour maintenir la cohérence globale et pour minimiser le coût de recalcul de la synergie dans de grands réseaux.

E. Risques d'Oscillation et Feedback Top-Down

Si les modifications des blocs R_i se produisent trop fréquemment ou de manière contradictoire, le SCN risque d'entrer dans une phase d'oscillations ou de chaos symbolique. Par exemple, une entité $\mathcal{E}_i$ pourrait insérer un axiome qui augmente temporairement la synergie avec $\mathcal{E}_j$, puis le retirer peu après, entraînant ainsi une fluctuation répétée de $\omega_{i,j}$. Pour éviter de telles instabilités, un **feedback**

top-down peut intervenir (voir section **3.4.3.2**). Ce mécanisme de rétroaction consiste en un module de contrôle global qui surveille l'état du SCN et, si le nombre de contradictions ou le taux de variation des pondérations dépasse un seuil critique, il impose une révision ou une réorganisation forcée des règles. Par exemple, on peut définir une condition de déclenchement :

$$\text{si } \mathrm{ContradictionCount}(t) > \Theta, \text{ alors initier une révision massive.}$$

Une telle intervention permet d'isoler ou de corriger les entités présentant des incohérences, assurant ainsi la convergence de la dynamique et la formation de clusters stables.

Conclusion

L'exemple d'un **DSL** stockant un ensemble de postulats, qui se modifie au fil du temps, illustre de manière concrète comment la dynamique symbolique peut s'inscrire dans un flux continu d'actualisation des connaissances. Chaque entité $\mathcal{E}_i$ est associée à un bloc de règles R_i qui évolue par des opérations d'insertion et de suppression, modifiant ainsi la synergie symbolique $\mathbf{S}_{\mathrm{sym}}(i,j)$ entre les entités. La mise à jour des pondérations se fait selon la formule :

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[\mathbf{S}_{\mathrm{sym}}(i,j,t+1) - \tau\,\omega_{i,j}(t)\big],$$

ce qui permet au SCN de s'ajuster en temps réel aux évolutions de la base de connaissances. Toutefois, à grande échelle et en présence de modifications fréquentes, des mécanismes d'optimisation (tels que l'indexation sémantique et les mises à jour paresseuses) ainsi qu'un système de versionnage devient indispensable pour assurer la cohérence globale. Par ailleurs, le recours à un feedback top-down est souvent nécessaire pour prévenir les oscillations ou le chaos symbolique, garantissant ainsi la stabilité du système. En définitive, ce scénario démontre comment un DSL peut intégrer et exploiter l'évolution continue des connaissances symboliques pour adapter sa structure d'interactions, tout en maintenant une cohérence logique et une organisation auto-organisée du réseau.

3.5. Représentations Hybrides (Sub-Symbolique + Symbolique)

Les sections précédentes ont montré les **forces** et **faiblesses** respectives des représentations **sub-symboliques** (chap. 3.3) et **symboliques** (chap. 3.4). Dans la pratique, un **DSL** (Deep Synergy Learning) peut grandement tirer profit d'une **fusion** des deux approches — c'est-à-dire que chaque entité $\mathcal{E}_i$ combine à la fois un **embedding** (capable de manipuler le bruit, les données massives, etc.) et un **bloc** de règles ou d'axiomes (garantissant l'interprétabilité, la cohérence logique). C'est ce qu'on appelle une **représentation hybride** ou **neuro-symbolique**, où l'on cherche à marier la **puissance statistique** et la **lisibilité** sémantique.

Dans la section 3.5, nous verrons la **motivation** (3.5.1) qui pousse à mélanger sub-symbolique et symbolique, puis nous discuterons (3.5.2) des **stratégies** concrètes pour concevoir un tel assemblage dans un DSL (comment définir $\mathbf{r}(i)$ comme un couple ou deux nœuds, comment moduler la synergie mixte, etc.). Enfin, nous conclurons (3.5.3) sur les **avantages** (explicabilité, adaptabilité) et les **défis** (complexité, gestion des contradictions), en prenant un exemple d'**agent conversationnel** où la composante logique et l'embedding linguistique cohabitent.

3.5.1. Motivation de la Fusion

L'idée d'**hybrider** le sub-symbolique (vecteurs, embeddings neuronaux) et le symbolique (règles, ontologies) découle du constat que ni l'un ni l'autre ne suffisent, à eux seuls, à couvrir tous les besoins du DSL (Deep Synergy Learning). Les embeddings gèrent bien le **bruit** et la **variabilité**, mais souffrent d'un manque d'**interprétabilité** et ne fournissent pas de raisonnement logique direct ; les représentations symboliques, elles, sont **rigides** et intransigeantes face à l'ambiguïté ou l'imprécision, mais elles offrent un cadre formel pour **expliquer** et **inférer**.

3.5.1.1. BÉNÉFICIER DE LA PUISSANCE STATISTIQUE DES EMBEDDINGS ET DE LA CLARTÉ DES RÈGLES LOGIQUES

Dans le cadre d'un **Deep Synergy Learning (DSL)**, l'intégration d'**embeddings sub-symboliques** et de blocs de **règles logiques** offre un paradigme hybride capable de conjuguer la robustesse statistique des approches d'apprentissage profond et la transparence inhérente aux systèmes symboliques. Ce modèle hybride tire profit de la **puissance** des techniques de projection vectorielle tout en fournissant un **raisonnement explicite** qui facilite l'interprétation des résultats dans des domaines critiques tels que la médecine ou la surveillance industrielle.

A. Dimension Sub-symbolique et Extraction d'Embeddings

Chaque entité $\mathcal{E}_i$ est associée à un **vecteur** $\mathbf{x}_i \in \mathbb{R}^d$ qui encode de manière dense ses attributs discriminants. Ces représentations issues d'architectures telles que les **CNN**, **Transformers** ou **autoencodeurs** permettent d'exploiter la **puissance statistique** des modèles d'apprentissage profond pour capturer les structures complexes dans des données massives et bruitées. Par exemple, la proximité entre deux entités peut être mesurée à l'aide de la **distance euclidienne** :

$$d_{\text{eucl}}(\mathbf{x}_i, \mathbf{x}_j) = \| \mathbf{x}_i - \mathbf{x}_j \|,$$

ou à l'aide d'un **noyau Gaussien** défini par

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2),$$

où $\gamma > 0$ ajuste la sensibilité de la mesure. Ces formules illustrent la capacité des **embeddings** à tolérer un **bruit** modéré et à gérer un flux massif d'instances, en extrayant automatiquement des caractéristiques discriminantes pertinentes.

B. Dimension Symbolique et Règles Logiques

En complément de la représentation sub-symbolique, le DSL dote chaque entité $\mathcal{E}_i$ d'un **bloc** de règles ou d'axiomes, noté $\{R_i\}$. Cette dimension symbolique permet d'initier un **raisonnement logique** explicite sur les entités. Les règles logiques offrent plusieurs avantages essentiels :

- Elles permettent d'identifier des **contradictions** ou des incohérences dans l'ensemble des connaissances.
- Elles facilitent l'**inférence explicite**, par exemple, en déduisant qu'« si A et B sont vraies, alors C l'est également ».
- Elles confèrent une **traçabilité** qui rend les décisions du système plus compréhensibles pour des experts dans des domaines où la **justification** des recommandations est indispensable.

Ainsi, dans des domaines tels que le **médical**, le bloc de règles peut encapsuler des protocoles diagnostiques (par exemple, « si **Toux**, **Fièvre** et saturation en oxygène $O2 < 92\%$, alors **Admission Urgente** »), offrant ainsi un **langage intelligible** aux experts.

C. Fusion Hybride : La Synergie Globale

Pour exploiter au mieux ces deux dimensions complémentaires, le DSL hybride définit la **synergie globale** entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ comme un **mélange** pondéré des similarités sub-symboliques et symboliques. Cette synergie est formalisée par l'équation suivante :

$$S_{\text{hybrid}}(i,j) = \alpha\, S_{\text{sub}}(\mathbf{x}_i, \mathbf{x}_j) + (1-\alpha)\, S_{\text{sym}}(R_i, R_j),$$

où $\alpha \in [0,1]$ est un paramètre de **pondération** qui permet de calibrer l'influence relative de chaque composante. La fonction S_{sub} évalue la **similarité vectorielle** (par exemple, en utilisant la formule $\exp(-\parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2)$ ou le **cosinus normalisé**), tandis que S_{sym} mesure la **compatibilité symbolique** entre les ensembles de règles $\{R_i\}$ et $\{R_j\}$ en tenant compte de critères tels que la **co-occurrence** des axiomes, l'absence de contradiction ou encore la **distance ontologique** entre les concepts.

Cette approche hybride permet d'obtenir une mesure de **synergie** qui bénéficie simultanément de la **robustesse statistique** et de la **clarté logique**, assurant ainsi que les entités se rapprochent dans le réseau non seulement lorsqu'elles sont proches sur le plan des **embeddings** (c.-à-d. lorsque $\mathbf{x}_i \approx \mathbf{x}_j$) mais également lorsqu'elles partagent des **règles logiques compatibles**.

D. Mise à Jour des Pondérations et Auto-Organisation

L'architecture du DSL utilise la synergie hybride dans la mise à jour dynamique des pondérations entre entités. La règle de mise à jour s'exprime par :

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\left[S_{\text{hybrid}}(i,j) - \tau\,\omega_{i,j}(t)\right],$$

où $\eta > 0$ représente le **taux d'apprentissage** et $\tau > 0$ est un **coefficient de décroissance** qui assure la stabilité des poids. Ainsi, le système procède à une **auto-organisation** dans laquelle les liaisons se renforcent si la **synergie hybride** est élevée et s'affaiblissent dans le cas contraire, aboutissant à des **clusters** d'entités cohérents tant du point de vue statistique que logique.

E. Applications dans des Domaines Critiques

L'intérêt d'une telle approche hybride se manifeste particulièrement dans des domaines où la **précision** et la **transparence** des décisions sont primordiales. Par exemple :

- Dans le **médical**, la partie sub-symbolique du DSL peut encoder des données brutes telles que des images médicales, des relevés cliniques ou des textes issus de l'analyse NLP, tandis que la composante symbolique intègre des règles diagnostiques éprouvées. Cette dualité permet de détecter des corrélations complexes tout en offrant des justifications claires pour chaque diagnostic.

- Dans la **surveillance industrielle**, les vecteurs décrivant des signaux (températures, vibrations) sont couplés à des règles de sécurité et de contrôle, garantissant ainsi une détection fine des pannes et la certification que les limites opérationnelles ne sont pas dépassées.

F. Conclusion

L'approche hybride proposée par le DSL permet de surmonter les limites d'un système purement sub-symbolique, souvent critiqué pour son **opacité**, et d'un système exclusivement symbolique, qui peut manquer de **souplesse** face aux signaux bruités et à la variabilité des données massives. En combinant la **puissance statistique** des **embeddings** et la **clarté** des **règles logiques**, le DSL hybride offre une solution robuste et **explicable**. Ce modèle permet non seulement une auto-organisation efficace par renforcement des liens entre entités similaires, mais fournit également une **traçabilité** indispensable dans des contextes où le **contrôle** et la **transparence** sont essentiels pour la prise de décision.

Ainsi, en conjuguant ces deux dimensions complémentaires, le DSL hybride confère au réseau la **capacité** de s'auto-organiser de manière **robuste** et **explicable**, répondant aux exigences des domaines critiques et offrant un avantage concurrentiel tant sur le plan de la performance que de l'interprétabilité.

3.5.1.2. APPROCHE "NEURO-SYMBOLIQUE" : DÉJÀ MENTIONNÉE DANS LA LITTÉRATURE IA

L'approche **neuro-symbolique** se présente comme une stratégie intégrative visant à fusionner la **puissance statistique** des techniques d'**embeddings neuronaux** avec la **rigueur logique** des

systèmes symboliques, c'est-à-dire l'ensemble de règles ou d'axiomes formels. Cette démarche, qui a été explorée dès les années 1980-1990 (voir notamment les travaux de *Smolensky*), a connu un regain d'intérêt avec la montée du **deep learning** dans les années 2010. Dans le contexte d'un **Deep Synergy Learning (DSL)**, cette approche permet d'associer, dans un même système, un apprentissage efficace sur de larges volumes de données bruitées à un raisonnement explicite assurant **traçabilité** et **cohérence** globale.

A. Dimension Sub-symbolique : Puissance des Embeddings Neuronaux

Dans le sous-système sub-symbolique, chaque entité $\mathcal{E}_i$ est représentée par un **vecteur** $\mathbf{x}_i \in \mathbb{R}^d$ issu d'un processus d'extraction automatique de caractéristiques via des réseaux de neurones profonds tels que les **CNN**, **Transformers** ou **autoencodeurs**. Ces **embeddings** permettent d'obtenir des représentations continues qui capturent des propriétés discriminantes des données. La proximité entre deux entités peut être quantifiée par des mesures telles que la **distance euclidienne** :

$$d_{\text{eucl}}(\mathbf{x}_i, \mathbf{x}_j) = \| \mathbf{x}_i - \mathbf{x}_j \|,$$

ou par un **noyau Radial Basis Function (RBF)** défini par

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \ \| \mathbf{x}_i - \mathbf{x}_j \|^2),$$

où $\gamma > 0$ est un paramètre d'échelle. Ces mesures sont particulièrement adaptées pour gérer des données massives et bruyantes, puisque les **embeddings** permettent de projeter des instances dans un espace où les structures sous-jacentes sont révélées par des similitudes quantitatives. La tolérance au bruit et la scalabilité de ces méthodes confèrent au système une **robustesse** statistique essentielle dans le traitement de flux de données complexes.

B. Dimension Symbolique : Rigueur Logique et Raisonnement Formatif

Parallèlement au traitement sub-symbolique, le DSL intègre une dimension symbolique qui se matérialise par un **bloc** de règles ou d'**axiomes** associé à chaque entité, noté R_i. Ce sous-système symbolique permet d'effectuer un **raisonnement explicite** et de garantir la **cohérence** des inférences. Par exemple, un ensemble de règles peut être formulé pour détecter des contradictions ou pour valider la cohérence d'un ensemble de postulats. Dans des domaines exigeant une **explicabilité** accrue, comme le domaine médical ou industriel, la capacité à justifier une décision à l'aide d'un langage formel (par exemple, « si A et B sont vraies, alors C doit l'être ») est un atout majeur.

La **dimension symbolique** permet également d'opérer des inférences en se basant sur des formalismes tels que les logiques classiques, la logique floue ou des systèmes experts, assurant ainsi que le raisonnement ne repose pas uniquement sur des corrélations statistiques, mais sur des critères explicites et traçables.

C. Fusion Hybride : Définition de la Synergie Globale

Pour conjuguer harmonieusement ces deux dimensions complémentaires, un **DSL hybride** définit la **synergie globale** entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ par une combinaison linéaire pondérée. Cette synergie est formalisée par l'équation

$$S_{\text{hybrid}}(i,j) = \alpha\, S_{\text{sub}}(\mathbf{x}_i, \mathbf{x}_j) + (1-\alpha)\, S_{\text{sym}}(R_i, R_j),$$

où $\alpha \in [0,1]$ module l'influence relative de la composante sub-symbolique par rapport à la composante symbolique. La fonction S_{sub} peut être, par exemple, une mesure de **similarité vectorielle** telle que

$$S_{\text{sub}}(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\| \mathbf{x}_i - \mathbf{x}_j \|^2)$$

ou une mesure basée sur le **cosinus normalisé**. Parallèlement, S_{sym} évalue la **compatibilité symbolique** entre les blocs de règles R_i et R_j en tenant compte d'éléments tels que la co-occurrence des règles, l'absence de contradiction ou la proximité ontologique entre les concepts impliqués. Ce modèle hybride permet ainsi aux entités de se regrouper dans le réseau non seulement en fonction de leur **proximité** dans l'espace vectoriel, mais aussi en fonction de leur **compatibilité logique**.

D. Auto-organisation du SCN Basée sur la Synergie Hybride

Le **Synergistic Connection Network (SCN)** intègre la synergie hybride dans la mise à jour dynamique de ses **pondérations** $\omega_{i,j}$. La règle de mise à jour adoptée est généralement de la forme

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\left[S_{\text{hybrid}}(i,j) - \tau\, \omega_{i,j}(t)\right],$$

où $\eta > 0$ représente le **taux d'apprentissage** et $\tau > 0$ un **coefficient de décroissance** permettant de réguler l'évolution des poids. Cette équation assure que les connexions entre entités se renforcent lorsque la synergie hybride est élevée – c'est-à-dire lorsque les entités sont à la fois proches du point de vue des **embeddings** et compatibles du point de vue des règles – et s'affaiblissent dans le cas contraire. Ainsi, le SCN s'auto-organise pour former des **clusters** d'entités cohérents, caractérisés par des liens forts qui résultent d'un double critère d'affinité.

E. Références Historiques et Applications Marquantes

L'**approche neuro-symbolique** s'inscrit dans un courant de recherche de longue date dans le domaine de l'**IA**. Dès les années 1980-1990, des chercheurs tels que *Smolensky* ont proposé des idées visant à relier les **représentations discrètes** aux modèles neuronaux, soulignant l'intérêt de combiner les forces des approches **connexionnistes** et **symboliques**. Avec l'essor du **deep learning**, de nombreux frameworks tels que **DeepProbLog** ou des méthodes de **Differentiable Reasoners** ont émergé, proposant des solutions pour intégrer la logique au sein de réseaux de neurones différentiables. De plus, le couplage d'ontologies (par exemple via OWL) avec des **embeddings** a permis d'enrichir les modèles de raisonnement en introduisant des connaissances formelles.

Les applications de l'approche neuro-symbolique sont nombreuses :

- Dans le domaine **médical**, l'intégration d'un module d'**embedding** pour l'analyse d'images ou de textes cliniques avec des règles diagnostiques permet d'expliquer les décisions d'un système et d'assurer une **explicabilité** indispensable.

- Dans un contexte **industriel**, la gestion de flux sensoriels massifs combinée à des règles de sécurité garantit une **tolérance** au bruit tout en assurant le respect des normes.

- En **linguistique**, le couplage d'un **transformer** avec des règles syntaxiques ou sémantiques permet de concilier l'**apprentissage** des représentations linguistiques et la rigueur de la grammaire explicite.

F. Difficultés et Interfaces Entre Systèmes Neuronaux et Symboliques

L'interface entre le sous-système **différentiable** (neuronale) et le sous-système **non différentiable** (symbolique) représente un défi majeur. Tandis que la partie neuronale se met à jour par rétropropagation et bénéficie de gradients continus, la partie symbolique relève souvent d'un raisonnement binaire. Des approches telles que la logique floue, la sémantique continue des règles ou l'intégration de solveurs SAT dans des architectures neuronales ont été proposées pour atténuer cette discontinuité et permettre une meilleure **intégration** des deux mondes. Par ailleurs, la gestion simultanée de nombreux **embeddings** et de multiples blocs de règles pose des problèmes de **complexité**, nécessitant des stratégies d'optimisation pour garantir l'évolutivité du système.

. Conclusion

L'**approche neuro-symbolique** se présente comme une voie prometteuse pour allier les avantages des méthodes **connexionnistes** – telles que la **tolérance** au bruit, la **scalabilité** et la capacité d'**apprentissage** sur de larges ensembles de données – aux atouts des systèmes **symboliques**, notamment la **traçabilité**, la **cohérence** et la capacité d'**inférence** explicable. Dans le cadre d'un **DSL**, cette intégration se matérialise par la formulation de la synergie hybride

$$S_{\text{hybrid}}(i,j) = \alpha\, S_{\text{sub}}(\mathbf{x}_i, \mathbf{x}_j) + (1-\alpha)\, S_{\text{sym}}(R_i, R_j),$$

et par l'adaptation des pondérations du SCN via

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[S_{\text{hybrid}}(i,j) - \tau\, \omega_{i,j}(t)\big].$$

Cette intégration donne naissance à des systèmes d'**IA robustes** capables de manipuler des données complexes tout en garantissant une **explicabilité** et une **cohérence** conceptuelle, confirmant ainsi l'intérêt soutenu de la recherche neuro-symbolique pour la construction d'algorithmes IA complets et explicables.

3.5.2.1. Entité = Couple $(\mathbf{x}_i, R_i)$? Ou Bien Deux Nœuds Connectés ?

L'intégration simultanée de la **partie sub-symbolique** (les embeddings) et de la **partie symbolique** (les règles logiques) dans un **DSL** (*Deep Synergy Learning*) peut être réalisée suivant deux stratégies architecturales distinctes dans la construction du **Synergistic Connection Network (SCN)**. Ces deux approches se distinguent par la manière dont elles structurent les entités du réseau et par la granularité à laquelle la double information – à la fois sub-symbolique et symbolique – est traitée.

A. Entité = Couple $(\boldsymbol{x}_i, R_i)$

Dans cette première configuration, chaque entité $\mathcal{E}_i$ est modélisée comme un **couple** unique constitué d'un **embedding** $\mathbf{x}_i \in \mathbb{R}^d$ et d'un ensemble de règles ou axiomes symboliques R_i. Le vecteur $\mathbf{x}_i$ peut être obtenu par divers mécanismes d'extraction automatique (par exemple, un

réseau de neurones convolutif pour l'analyse d'images, un Transformer pour le traitement de texte ou encore un autoencodeur pour d'autres types de signaux), tandis que R_i recense des connaissances explicites telles que des règles logiques, des axiomes ou des déclarations formelles qui décrivent l'entité.

Dans ce schéma, le **SCN** comporte exactement n nœuds pour n entités, chaque nœud encapsulant simultanément la dimension sub-symbolique et la dimension symbolique. La **synergie** $S(i,j)$ entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ est alors évaluée à partir d'une combinaison des similarités issues de leurs embeddings et de la compatibilité de leurs blocs de règles. Ainsi, le calcul de $S(i,j)$ intègre par exemple des mesures telles que

$$S_{\text{sub}}(\mathbf{x}_i, \mathbf{x}_j)$$

pour quantifier la proximité dans l'espace vectoriel, et

$$S_{\text{sym}}(R_i, R_j)$$

pour juger de l'absence de contradiction ou de la co-occurrence des axiomes associés.

Un avantage majeur de cette approche réside dans sa **simplicité** architecturale, puisque l'on ne duplique pas le nombre de nœuds dans le réseau. Chaque entité gère en interne la double information, ce qui facilite une mise à jour unifiée des pondérations $\omega_{i,j}$ selon une règle d'auto-organisation unique. Toutefois, cette fusion présente également des inconvénients où la gestion conjointe du vecteur $\mathbf{x}_i$ et du bloc R_i peut complexifier le processus de réapprentissage, notamment si l'on souhaite réentraîner la composante sub-symbolique indépendamment des règles logiques. La modularité interne peut ainsi s'avérer moins flexible, car la mise à jour globale repose sur un seul nœud regroupant des modalités hétérogènes.

B. Deux Nœuds Connectés : L'Entité Sub-Symbolique et l'Entité Symbolique

La seconde stratégie consiste à dissocier les deux composantes d'une entité $\mathcal{E}_i$ en deux **nœuds** distincts, chacun spécialisé dans l'une des deux modalités. Ainsi, on définit :

$$\begin{aligned} \mathcal{E}_i^{(\text{sub})} &: \text{ embedding neuronal } \mathbf{x}_i \in \mathbb{R}^d, \\ \mathcal{E}_i^{(\text{sym})} &: \text{ bloc de règles ou axiomes } R_i. \end{aligned}$$

Ces deux nœuds, qui représentent respectivement la partie sub-symbolique et la partie symbolique de l'entité, sont interconnectés par un **lien** explicite ou un identifiant commun qui permet d'indiquer qu'ils décrivent ensemble la même entité $\mathcal{E}_i$. Dans le SCN ainsi constitué, il est alors possible de distinguer deux sous-réseaux où un sous-réseau sub-symbolique associe les pondérations $\omega_{(i^{(\text{sub})}, j^{(\text{sub})})}$ à la proximité vectorielle, par exemple via la norme $\| \mathbf{x}_i - \mathbf{x}_j \|$, tandis qu'un sous-réseau symbolique quantifie la compatibilité entre les ensembles de règles R_i et R_j à travers les pondérations $\omega_{(i^{(\text{sym})}, j^{(\text{sym})})}$. Par la suite, un mécanisme de **fusion** combine ces deux scores pour restituer la synergie globale $S(i,j)$.

L'approche à deux nœuds offre une meilleure **modularité** puisqu'elle permet de traiter séparément et d'optimiser indépendamment les deux sous-systèmes. Par exemple, il devient aisé de réentraîner la partie d'extraction d'embeddings sans impacter le module symbolique, ou inversement.

Néanmoins, cette dissociation a pour conséquence de doubler le nombre de nœuds dans le réseau (passant de n à $2n$ pour n entités), ce qui peut accroître le coût de calcul lors de la mise à jour des pondérations. De plus, il faut élaborer une stratégie explicite pour fusionner les scores sub-symbolique et symbolique, c'est-à-dire déterminer la fonction de combinaison qui produira le score final de synergie entre deux entités.

C. Choix Pratique et Impact Architectural

Le choix entre la modélisation d'une entité en tant que **couple unique** $(\mathbf{x}_i, R_i)$ et la dissociation en deux **nœuds distincts** dépend de plusieurs considérations pratiques :

- **Échelle du Système** : Dans un DSL comportant un grand nombre d'entités, la multiplication du nombre de nœuds (passage de n à $2n$) peut alourdir la structure du SCN et impacter la complexité computationnelle de la mise à jour des pondérations.

- **Modularité et Flexibilité** : Si l'on souhaite réentraîner ou modifier indépendamment la composante sub-symbolique (les embeddings) et la composante symbolique (les règles), la séparation en deux nœuds offre une meilleure modularité et facilite la gestion de la mise à jour de chacun des sous-systèmes.

- **Simplicité de l'Implémentation** : Un nœud mixte, qui intègre directement les deux composantes dans un unique vecteur complexe, présente l'avantage d'une architecture plus simple et d'une auto-organisation unifiée. Cependant, cette simplicité peut être contrebalancée par une complexification dans la gestion interne des informations hétérogènes.

Dans les deux cas, la mise à jour des pondérations suit une règle de la forme

$$\omega_{a,b}(t+1) = \omega_{a,b}(t) + \eta\big[\text{synergie}(a,b) - \tau\,\omega_{a,b}(t)\big],$$

où le terme synergie(a, b) est défini différemment selon que a et b représentent des nœuds mixtes (cas A) ou des nœuds séparés (cas B). La fonction de fusion, qui combine la similarité sub-symbolique et la compatibilité symbolique, doit être soigneusement conçue pour garantir que le SCN converge vers une configuration stable et interprétable.

Conclusion

Les deux stratégies pour intégrer simultanément la partie sub-symbolique et la partie symbolique dans un DSL se distinguent par leur approche de la représentation des entités. La première option, qui définit chaque entité $\mathcal{E}_i$ comme un **couple** $(\mathbf{x}_i, R_i)$, offre une solution unifiée et simple en termes de nombre de nœuds, mais peut limiter la modularité en imposant une gestion conjointe des deux types d'informations. La seconde option, qui dissocie l'entité en deux nœuds distincts – l'un pour l'**embedding neuronal** et l'autre pour le **bloc symbolique** – apporte une meilleure séparation des préoccupations et une flexibilité accrue dans la mise à jour de chaque sous-système, au prix d'une duplication du nombre de nœuds et d'une complexité de fusion supplémentaire.

Le choix entre ces deux approches dépend donc des contraintes architecturales, de la taille du réseau et des exigences en matière de modularité et de réapprentissage. Le reste de la section (3.5.2.2) détaillera la **formule** de synergie mixte qui permet de fusionner les informations sub-symboliques et symboliques, tandis que la section (3.5.2.3) illustrera l'application de ce schéma

dans des cas pratiques exploitant la force statistique des embeddings et la clarté du raisonnement symbolique.

3.5.2.2. Fonctions de Synergie Mixtes : $\mathbf{S}_{\text{sub, sym}}$

Dans le contexte d'un **Deep Synergy Learning (DSL)**, la nécessité de mesurer la proximité ou la compatibilité entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ se complexifie lorsqu'on intègre simultanément une composante sub-symbolique – représentée par des **embeddings** $\mathbf{x}_i$ – et une composante symbolique – exprimée par un ensemble de règles ou axiomes R_i. Afin d'obtenir un **score global** qui reflète à la fois la similarité dans l'espace vectoriel et la compatibilité logique, il convient de définir une fonction de synergie mixte, notée $\mathbf{S}_{\text{hybrid}}(i,j)$. Cette section présente diverses approches d'agrégation qui permettent de fusionner ces deux dimensions, chacune offrant des compromis entre **robustesse statistique** et **explicabilité logique**.

A. Formule Linéaire Simple

La méthode la plus intuitive consiste à combiner les deux scores de manière **linéaire**. Ainsi, la synergie globale peut être définie par :

$$\mathbf{S}_{\text{hybrid}}(i,j) = \alpha\, \mathbf{S}_{\text{sub}}\big(\mathbf{x}_i, \mathbf{x}_j\big) + (1-\alpha)\, \mathbf{S}_{\text{sym}}\big(R_i, R_j\big),$$

où $\alpha \in [0,1]$ est un paramètre de pondération ajustable. La fonction $\mathbf{S}_{\text{sub}}$ évalue la **similarité vectorielle** – par exemple, via une mesure de similarité cosinus ou une fonction de noyau Gaussien telle que

$$\mathbf{S}_{\text{sub}}\big(\mathbf{x}_i, \mathbf{x}_j\big) = \exp\big(-\gamma \parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2\big),$$

avec $\gamma > 0$ –, tandis que $\mathbf{S}_{\text{sym}}$ quantifie la **compatibilité logique** entre les blocs de règles R_i et R_j (en évaluant, par exemple, la co-occurrence des axiomes ou en appliquant des tests de contradiction). Une telle combinaison linéaire présente l'avantage d'être aisément interprétable où, en modulant α, il est possible de privilégier l'une ou l'autre des composantes en fonction du contexte d'application. Par ailleurs, une variation affine peut être introduite afin d'ajuster les échelles des scores issus des deux sous-systèmes :

$$\mathbf{S}_{\text{hybrid}}(i,j) = \alpha\, g\left(\mathbf{S}_{\text{sub}}\big(\mathbf{x}_i, \mathbf{x}_j\big)\right) + (1-\alpha)\, h\left(\mathbf{S}_{\text{sym}}\big(R_i, R_j\big)\right),$$

où les fonctions g et h assurent un recalibrage permettant d'éviter que l'une des composantes n'écrase l'autre en cas de plages de valeurs très différentes.

B. Combinaison Multiplicative ou Min–Max

Une autre approche consiste à opter pour une agrégation **multiplicative**, dans laquelle le score global est obtenu en multipliant les scores individuels :

$$\mathbf{S}_{\times}(i,j) = \mathbf{S}_{\text{sub}}(i,j) \times \mathbf{S}_{\text{sym}}(i,j).$$

Cette formulation exige que chacune des deux composantes prenne une valeur élevée pour que le produit soit important. Ainsi, si l'une des deux valeurs est proche de zéro, le score global s'en trouve fortement diminué, ce qui renforce l'idée d'une nécessité d'accord simultané entre la partie

sub-symbolique et la partie symbolique. Une alternative au produit consiste à utiliser l'opérateur min, définissant le score global comme le minimum des deux scores :

$$\mathbf{S}_{\min}(i,j) = \min\{\mathbf{S}_{\text{sub}}(i,j),\, \mathbf{S}_{\text{sym}}(i,j)\}.$$

Cette formulation, tout en partageant l'esprit de l'approche multiplicative, présente un comportement légèrement différent en ce qu'elle retient la composante la plus faible. Par conséquent, un faible résultat dans l'une des dimensions impose directement une limitation sur la synergie globale.

C. Stratification ou Pondération Conditionnelle

Une troisième piste consiste à introduire une **pondération conditionnelle** ou une stratification, c'est-à-dire à faire intervenir l'une des composantes uniquement lorsque l'autre satisfait un critère minimal de fiabilité. Par exemple, on peut définir une fonction de synergie hybride conditionnelle comme suit :

$$\mathbf{S}_{\text{hybrid}}(i,j) = \begin{cases} \mathbf{S}_{\text{sub}}(i,j), & \text{si } \mathbf{S}_{\text{sub}}(i,j) < \theta, \\ \alpha\, \mathbf{S}_{\text{sub}}(i,j) + (1-\alpha)\, \mathbf{S}_{\text{sym}}(i,j), & \text{sinon,} \end{cases}$$

où θ est un seuil prédéfini. Dans cette configuration, la composante symbolique n'intervient que lorsque la similarité sub-symbolique atteint un certain niveau, assurant ainsi que le score global reflète d'abord une exclusion basée sur le critère vectoriel avant d'intégrer une pondération combinée. De plus, il est envisageable d'introduire une fonction de pondération dynamique, $\alpha(i,j)$, qui évolue en fonction de la fiabilité ou de la cohérence de l'embedding $\mathbf{x}_i$ ou du bloc R_i, permettant ainsi un ajustement local et adaptatif de la fusion des scores.

D. Impact sur la Mise à Jour des Pondérations dans le DSL

Quel que soit le schéma d'agrégation choisi – qu'il s'agisse de la somme linéaire, du produit, du minimum ou d'une combinaison conditionnelle – la règle de mise à jour des pondérations au sein du **SCN** demeure inchangée :

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[\mathbf{S}_{\text{hybrid}}(i,j) - \tau\, \omega_{i,j}(t)\big],$$

où $\eta > 0$ est le taux d'apprentissage et $\tau > 0$ le coefficient de décroissance. Dans ce cadre, la fonction $\mathbf{S}_{\text{hybrid}}(i,j)$ remplace une mesure unidimensionnelle de similarité en offrant une évaluation composite qui prend en compte à la fois la **proximité** dans l'espace des embeddings et la **compatibilité** des ensembles de règles. Ainsi, une paire d'entités renforce sa connexion dans le réseau uniquement si les deux critères – la similarité sub-symbolique et la cohérence symbolique – sont satisfaits, renforçant ainsi l'idée d'un accord simultané nécessaire pour obtenir un score élevé. Le choix du schéma d'agrégation détermine alors la sensibilité du système aux valeurs extrêmes et la manière dont les imperfections dans l'une des composantes affectent globalement la synergie.

Conclusion

Les différentes fonctions de synergie mixtes présentées ici constituent des outils essentiels pour fusionner les informations provenant des domaines sub-symbolique et symbolique dans un DSL. La formulation linéaire simple offre une solution claire et modulable, tandis que les approches

multiplicatives ou basées sur le minimum imposent une exigence d'accord simultané plus stricte. Les stratégies de pondération conditionnelle, quant à elles, permettent une intégration adaptative qui tient compte de la fiabilité relative des deux sources d'information. Quel que soit le choix effectué, l'agrégation se traduit dans la mise à jour des pondérations du SCN, garantissant que le réseau s'auto-organise de manière à renforcer les liens entre des entités à la fois proches dans l'espace vectoriel et compatibles sur le plan logique. Cette intégration fine des deux modalités confère au DSL une robustesse et une explicabilité particulièrement appréciables dans des applications critiques, où la convergence d'une **puissance statistique** et d'un **raisonnement formel** constitue un avantage majeur.

3.5.2.3. Applications : Bases de Connaissance Enrichies, Multi-sensoriels et Concepts

Dans le cadre d'un **Deep Synergy Learning (DSL)** hybride, l'intégration simultanée des informations sub-symboliques – représentées par des **embeddings** extraits de données brutes – et des informations symboliques – traduites par des **règles** ou des axiomes formels – permet de créer des systèmes capables de traiter des environnements complexes tout en garantissant la **cohérence** et l'**explicabilité** de leurs inférences. Cette section examine plusieurs domaines d'application de ce paradigme hybride, en illustrant comment la fusion de ces deux types de représentations enrichit les capacités d'un système et améliore la qualité de ses clusters ou associations.

A. Bases de Connaissance Enrichies

Une application classique de cette approche hybride apparaît dans la construction de **bases de connaissance (Knowledge Bases, KB)**. Dans ces systèmes, chaque entité $\mathcal{E}_i$ se voit attribuer à la fois un **vecteur** $\mathbf{x}_i \in \mathbb{R}^d$ issu d'un traitement sub-symbolique – par exemple, via des méthodes d'apprentissage profond appliquées à des textes ou des images – et un **bloc** de règles R_i qui formalise des assertions ou des relations symboliques (telles que des triplets RDF ou des graphes ontologiques). L'objectif consiste à conjuguer la **fiabilité formelle** (mesurée par la cohérence et la non-contradiction des règles) avec la capacité de détection de motifs latents offerte par les embeddings.

Ainsi, lorsque le DSL évalue la synergie globale

$$S_{\text{hybrid}}(i,j) = \alpha\, S_{\text{sub}}\big(\mathbf{x}_i, \mathbf{x}_j\big) + (1-\alpha)\, S_{\text{sym}}\big(R_i, R_j\big),$$

une forte similarité vectorielle (i.e. $\mathbf{x}_i \approx \mathbf{x}_j$) peut conduire à un rapprochement même en l'absence d'un lien explicite dans le domaine symbolique. Inversement, une forte **compatibilité** ontologique, où $R_i \cup R_j$ est hautement cohérent, peut compenser un manque relatif d'information dans l'extraction d'embeddings. Des applications telles que **Wikidata** illustrent ce phénomène, chaque entité (lieu, personne, concept) étant ainsi dotée d'une double description permettant au DSL de révéler des clusters enrichis et dynamiques, facilitant la navigation et l'exploitation d'un vaste ensemble de connaissances hétérogènes.

B. Applications Multi-Sensoriels et Concepts

Dans des environnements **multi-sensoriels**, où les données proviennent de sources variées (images, vidéos, audio, signaux de capteurs physiques), la combinaison d'un **embedding** sub-symbolique et d'un bloc symbolique offre un avantage décisif pour la détection d'anomalies ou la catégorisation

de contextes. La partie sub-symbolique, obtenue par des modèles tels que les CNN pour l'imagerie ou des transformeurs pour l'audio et le texte, capture la **structure** complexe des signaux bruts. Parallèlement, le bloc symbolique R_i encode des concepts ou des règles – par exemple, des spécifications indiquant que « la température dépasse 50°C, donc il existe un risque de surchauffe » ou « cette séquence vidéo se situe dans une ZoneInterdite » – qui permet d'interpréter ces signaux de manière explicite.

Dans un tel contexte, un DSL industriel de **surveillance** ou en **robotique** peut ainsi renforcer les liens entre entités uniquement lorsque la **cohérence** est constatée à la fois sur le plan sensoriel (embeddings similaires) et sur le plan conceptuel (règles non contradictoires). Cette double validation conduit à la formation de clusters robustes, capables de détecter avec précision des événements anormaux ou des contextes critiques. Par exemple, dans un système de surveillance, les signaux analogues détectés par différents capteurs pourront être rapprochés s'ils respectent en parallèle des conditions explicites définies dans les règles de sécurité.

C. Cas d'Étude et Bénéfices

Plusieurs cas d'étude illustrent l'intérêt d'une approche hybride :

3. **En Médecine** :

   Un dossier patient peut être représenté par un **embedding $\mathbf{x}_i$** issu d'images médicales, de relevés cliniques ou de textes, complété par un bloc R_i contenant des règles diagnostiques (par exemple, « si Toux, Fièvre et saturation en oxygène $O_2 < 92\%$, alors admission en urgence »). La synergie mixte permet ainsi de regrouper des patients présentant des caractéristiques similaires tout en validant ces regroupements par la cohérence des antécédents ou des protocoles médicaux.

4. **En Droit ou en Compliance** :

   Dans la gestion de contrats ou de documents juridiques, le texte brut d'un document peut être transformé en un vecteur $\mathbf{x}_i$ et associé à des règles précises clarifiant des clauses ou des obligations. Le DSL rapproche des documents non seulement par la similitude de leur contenu, mais aussi par le respect des critères légaux ou contractuels, facilitant ainsi la vérification et la conformité.

5. **En Surveillance Industrielle** :

   Dans un environnement industriel, les signaux provenant de capteurs (température, vibrations, etc.) sont traités par des modèles neuronaux pour extraire des embeddings, tandis que des règles de sécurité précisent les seuils critiques ou les conditions d'alerte. Le DSL hybride permet alors de détecter des anomalies en associant la similarité des signaux à la validation des règles opérationnelles.

L'avantage clé d'un DSL hybride réside dans la **complémentarité** des deux approches où la partie sub-symbolique offre une grande capacité d'**apprentissage** et de découverte de nouveaux motifs en présence de données bruitées et hétérogènes, tandis que la composante symbolique assure une **cohérence** et une **explicabilité** indispensables dans des applications critiques. Le résultat est un système capable de constituer des clusters ou des associations d'entités plus fiables et interprétables.

Conclusion

Les applications d'un **DSL hybride** se déploient de manière efficace dans des domaines aussi variés que les bases de connaissances enrichies, la surveillance multi-sensorielle ou des environnements nécessitant un raisonnement formel (médical, juridique, industriel). En combinant pour chaque entité une représentation sub-symbolique $\mathbf{x}_i$ et un bloc de règles R_i, le système bénéficie simultanément de la **puissance** des techniques d'apprentissage profond et de la **rigueur** des approches symboliques. Les fonctions de synergie mixtes, telles que celles décrites dans la section 3.5.2.2, permettent d'auto-organiser un réseau – le **SCN** – où les liens $\omega_{i,j}$ se renforcent uniquement lorsque les deux dimensions convergent vers une **similitude** forte et une **compatibilité** logique élevée. Ce paradigme assure ainsi une meilleure gestion des environnements complexes, en offrant à la fois robustesse, flexibilité et explicabilité dans l'analyse et le regroupement des entités.

3.5.3. Avantages et Défis

Après avoir exploré (3.5.1) la **motivation** pour fusionner sub-symbolique et symbolique, et (3.5.2) les **stratégies** possibles (comment construire une entité hybride et définir la synergie mixte), nous abordons maintenant les **avantages** (3.5.3.1) et les **défis** (3.5.3.2) qu'implique une telle fusion dans le cadre d'un **DSL** (Deep Synergy Learning). Nous conclurons (3.5.3.3) par un **cas d'usage** illustratif (agent conversationnel logique + embedding).

3.5.3.1. Meilleure Explicabilité et Adaptabilité Statistique

La représentation hybride, qui associe un **embedding** neuronal $\mathbf{x}_i$ à un **bloc logique** R_i, confère à un **Deep Synergy Learning (DSL)** une capacité accrue à rendre ses décisions et ses regroupements à la fois **transparents** et **adaptatifs**. Dans un système purement sub-symbolique, la similarité entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ se mesure essentiellement par des critères quantitatifs, tels que la distance euclidienne

$$\| \mathbf{x}_i - \mathbf{x}_j \|,$$

ou la similarité cosinus, qui, bien que efficaces pour capturer des corrélations statistiques, restent difficiles à interpréter de manière sémantique. L'intégration d'un bloc symbolique, consistant en un ensemble de règles, axiomes ou ontologies, apporte un éclairage supplémentaire permettant de comprendre la **raison** d'un rapprochement entre deux entités. Ainsi, si deux dossiers patients présentent des embeddings similaires, c'est la présence de règles diagnostiques communes ou la cohérence de leurs antécédents qui justifiera leur regroupement. Cette double dimension – **adaptation statistique** et **explicabilité logique** – offre un justificatif explicite et vérifiable aux experts.

L'**explicabilité** se matérialise par la capacité du système à lister ou à mettre en évidence les règles partagées, à démontrer l'absence de contradictions et à exposer le recouvrement ontologique entre deux entités. Dans un contexte médical, par exemple, la partie sub-symbolique peut établir une similarité basée sur des images scannées ou des relevés vitaux, tandis que le bloc symbolique clarifie que les dossiers patients se rapprochent parce qu'ils satisfont les mêmes critères

diagnostiques, tels que définis par des règles explicites. Cette transparence renforce la confiance dans le système et facilite le contrôle des décisions prises par le réseau.

Sur le plan de l'**adaptabilité statistique**, l'association d'un embedding neuronal à un cadre logico-symbolique permet de bénéficier d'un apprentissage continu et évolutif. La partie sub-symbolique, grâce à des mécanismes tels que le fine-tuning ou le ré-entraînement partiel, est capable de s'ajuster en fonction des nouveaux flux de données, absorbant ainsi la variabilité et le **bruit** inhérents à un grand volume d'informations. Par exemple, lorsque de nouvelles données apparaissent, les vecteurs $\mathbf{x}_i$ se réajustent pour intégrer de nouveaux motifs ou corrélations, tout en restant encadrés par la composante symbolique qui assure la **stabilité conceptuelle**. En effet, la couche symbolique agit comme une contrainte qui interdit des dérives excessives en imposant des règles immuables ou des axiomes fondamentaux, ce qui évite une divergence inappropriée de l'espace des embeddings.

L'**interaction** entre ces deux composantes crée un couplage dynamique dans lequel les découvertes statistiques peuvent mener à la mise à jour ou à l'enrichissement des règles existantes. À l'inverse, des ajustements dans le bloc symbolique, par exemple la création d'un nouvel axiome fondé sur des observations répétées, peuvent orienter la réorganisation des embeddings, en forçant des rapprochements ou des éloignements dans l'espace $\mathbb{R}^d$. Ce mécanisme de « double contrainte » renforce la capacité du DSL à créer des liens solides et explicables tout en restant ouvert à de grandes variations dans les données, combinant ainsi **robustesse** et **transparence**.

En résumé, la **représentation hybride** utilisée dans le DSL permet de concilier deux objectifs essentiels dans des domaines sensibles tels que le médical, l'industriel ou le juridique. La composante sub-symbolique offre une **adaptation continue** aux nouveaux influx de données grâce à la puissance des techniques d'apprentissage profond, tandis que la composante symbolique garantit une **explicabilité** et une **cohérence** qui rendent le système compréhensible et contrôlable par des experts. Ce couplage renforce ainsi la capacité globale du réseau à s'auto-organiser de manière efficace et transparente, en assurant que les regroupements d'entités soient à la fois statistiquement pertinents et logiquement justifiés.

3.5.3.2. Complexité et Gestion des Contradictions Symboliques vs. Incertitude Sub-Symbolique

Dans un **Deep Synergy Learning (DSL)** hybride, l'intégration simultanée des informations sub-symboliques – représentées par des **embeddings neuronaux** – et des informations symboliques – issues de **règles** ou d'**ontologies** – permet de combiner la **robustesse statistique** à la capacité d'**inférence formelle**. Toutefois, cette fusion engendre également des défis majeurs, tant sur le plan **algorithmique** que conceptuel, du fait de la coexistence d'une **logique stricte** et d'une **incertitude progressive** inhérente aux représentations apprises. Nous proposons ici une analyse en deux axes complémentaires où, d'une part, la **complexité algorithmique** est induite par la comparaison conjointe des composantes sub-symboliques et symboliques, et d'autre part, la gestion des tensions entre les **contradictions symboliques** et l'**incertitude sub-symbolique** constitue un défi central.

A. Complexité Algorithmique Accrue

Dans un système purement sub-symbolique, le calcul de la similarité entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ repose sur des opérations vectorielles ou sur des fonctions noyau, par exemple

$$\mathbf{S}_{\text{sub}}(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2),$$

où $\gamma > 0$. La complexité de telles opérations est généralement de l'ordre de $O(d)$ par comparaison, ou $O(n^2 \times d)$ pour l'évaluation de toutes les paires dans un ensemble de n entités. Lorsque l'on ajoute la composante symbolique, qui implique la comparaison de deux blocs de règles R_i et R_j, la charge de calcul s'accroît significativement. En effet, le **matching** de règles ou d'axiomes, et les tests de cohérence – qui peuvent nécessiter de parcourir un espace exponentiel en fonction de la complexité du langage logique (par exemple dans des formalismes tels que OWL DL) – viennent s'ajouter au coût de calcul sub-symbolique. Ainsi, la mise à jour des pondérations dans le DSL, donnée par

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\left[S_{\text{hybrid}}(i,j) - \tau\, \omega_{i,j}(t)\right],$$

devient une opération dont la complexité globale est la somme de l'évaluation sub-symbolique (en $O(n^2 \times d)$) et d'une composante symbolique potentiellement NP-complète.

Pour pallier cette surcharge, plusieurs **stratégies** heuristiques peuvent être déployées. Par exemple, l'utilisation d'algorithmes de type **k-NN** ou d'**approximate nearest neighbor** permet de limiter les comparaisons symboliques à un voisinage restreint des entités les plus proches sur le plan sub-symbolique. De même, l'imposition d'un **seuil de similarité** permet d'effectuer des comparaisons logiques uniquement lorsque la proximité vectorielle atteint un certain niveau. Enfin, des techniques d'**indexation symbolique** peuvent organiser les axiomes ou concepts de manière hiérarchique, afin de réduire la nécessité de tests complets de contradiction pour chaque paire.

B. Contradictions Symboliques vs. Incertitude Sub-Symbolique

L'intégration de deux modalités hétérogènes dans le DSL peut générer des **signaux contradictoires** entre la partie symbolique et la partie sub-symbolique. D'un côté, la logique formelle, caractérisée par une **précision stricte**, peut conduire à rejeter la compatibilité entre deux entités si leurs blocs de règles R_i et R_j présentent des **contradictions** (par exemple, lorsque des axiomes incompatibles sont détectés dans $R_i \cup R_j$). Dans un système purement symbolique, une telle contradiction aurait pour effet d'annuler toute similarité entre les entités.

D'un autre côté, la composante sub-symbolique, fondée sur des embeddings calculés à partir de données bruyantes et complexes, introduit une **incertitude** inhérente. Il est fréquent que deux entités présentent des **embeddings** très similaires, malgré une ambiguïté dans l'interprétation logique de leurs caractéristiques. Ainsi, il peut exister un désaccord où la partie logique indique une incompatibilité tandis que la partie neuronale suggère une forte similarité.

Pour gérer cette ambiguïté, plusieurs mécanismes d'**ajustement** sont envisageables. Dans l'agrégation hybride

$$\mathbf{S}_{\text{hybrid}}(i,j) = \alpha\, \mathbf{S}_{\text{sub}}(i,j) + (1-\alpha)\, \mathbf{S}_{\text{sym}}(i,j),$$

le paramètre α peut être modulé afin de refléter la confiance relative dans chaque composante. Par ailleurs, l'implémentation d'un **feedback top-down** peut permettre d'ajuster dynamiquement les règles logiques si des incompatibilités persistantes sont détectées, ou inversement, de modifier les embeddings si la composante symbolique démontre une cohérence forte dans la formation des clusters. Une autre solution consiste à appliquer un mécanisme de **saturation** qui limite l'influence

négative d'une contradiction symbolique, de manière que celle-ci réduise la synergie sans l'annuler complètement. Cette approche permet ainsi d'obtenir un compromis où une contradiction symbolique forte diminue significativement le score, mais ne paralyse pas la formation d'un lien si les embeddings fournissent un signal robuste.

C. Synthèse et Implications Pratiques

La cohabitation dans un DSL d'une **dimension sub-symbolique** et d'une **dimension symbolique** conduit à une complexité accrue où il est nécessaire de calculer et de fusionner deux types d'évaluations. D'une part, la proximité dans l'espace vectoriel, qui est déjà coûteuse pour de grands ensembles de données, et d'autre part, la vérification logique, souvent associée à des algorithmes dont la complexité peut croître de manière exponentielle. En parallèle, la gestion des éventuelles contradictions symboliques et de l'incertitude sub-symbolique impose un arbitrage délicat, qui se traduit par le choix du paramètre de pondération α et par l'adoption d'opérateurs combinatoires adaptés (tels que le produit ou le minimum).

Dans une implémentation pratique, la stabilité d'un DSL hybride repose sur trois axes principaux :

- La **gestion algorithmique** par le biais d'heuristiques (k-NN, seuils, indexation) permettant de limiter le coût de calcul des comparaisons logiques.
- La capacité à arbitrer les conflits entre une logique stricte et une incertitude statistique en adaptant dynamiquement la contribution de chaque composante.
- L'établissement d'un **métaniveau de contrôle** (feedback top-down) qui permet, en cas de divergence trop marquée, d'ajuster soit les règles, soit les embeddings afin d'assurer une convergence cohérente du système.

Cette approche, en somme, renforce l'intérêt des systèmes neuro-symboliques en montrant que, bien que l'intégration de deux modalités de représentation introduise une complexité supplémentaire et des tensions potentielles, il est possible d'élaborer des stratégies pour en atténuer les effets négatifs tout en tirant parti de la complémentarité offerte par chaque sous-système. Le DSL hybride, s'il est correctement paramétré et optimisé, parvient ainsi à allier la **robustesse statistique** d'un apprentissage profond avec la **rigueur** et la **transparence** d'un raisonnement formel, garantissant ainsi des clusters d'entités qui sont à la fois pertinents et explicables.

En définitive, la gestion de la **complexité algorithmique** et des **contradictions** symboliques dans un DSL hybride nécessite un pilotage attentif des interactions entre les composants sub-symboliques et symboliques. La mise en œuvre de techniques d'approximation et de stratégies d'arbitrage permet d'obtenir un système capable de s'adapter à des environnements complexes, tout en maintenant une cohérence conceptuelle et une capacité d'explication indispensable dans des domaines sensibles.

3.5.3.3. Cas d'Usage : Un Agent Conversationnel Fusionnant Logique et Embedding

Dans le cadre d'un **DSL** (*Deep Synergy Learning*) hybride, l'intégration d'une composante **sub-symbolique** (embeddings neuronaux) et d'une composante **symbolique** (règles logiques, ontologies) trouve une illustration particulièrement parlante dans le domaine des **agents conversationnels**. L'idée fondamentale consiste à doter un chatbot d'une **représentation vectorielle** issue de modèles transformeurs (par exemple, BERT ou GPT) et d'un **bloc logique** structuré, capable d'assurer la cohérence et l'explicabilité du dialogue. Cette architecture permet ainsi de répondre aux requêtes en conciliant la capacité d'apprentissage à grande échelle et la transparence des raisonnements formels.

A. Architecture Hybride : Sub-Symbolique et Symbolique

Dans une configuration hybride, chaque message utilisateur, noté $\mathcal{E}_{\text{user}}$, est modélisé de manière double. La composante sub-symbolique se traduit par un **vecteur** $\mathbf{x}_{\text{user}} \in \mathbb{R}^d$, obtenu via un réseau neuronal entraîné sur un large corpus de dialogues. Parallèlement, la composante symbolique est représentée par un **bloc** de règles $\{R_{\text{user}}\}$ qui capture l'intention ou le sens explicite du message, par exemple en identifiant des concepts tels que "vol", "réservation", ou "date". De manière similaire, chaque module interne ou scénario de dialogue – tel que « achat de billets », « météo » ou « small-talk » – est également caractérisé par un embedding $\mathbf{x}_j$ et un bloc de règles $\{R_j\}$. La **synergie** entre le message utilisateur et un module spécifique est alors évaluée par la fonction hybride

$$S_{\text{hybrid}}\left(\mathcal{E}_{\text{user}}, \mathcal{E}_j\right) = \alpha\, S_{\text{sub}}\left(\mathbf{x}_{\text{user}}, \mathbf{x}_j\right) + (1-\alpha)\, S_{\text{sym}}\left(R_{\text{user}}, R_j\right),$$

où le paramètre $\alpha \in [0,1]$ module l'importance relative de la **proximité vectorielle** et de la **compatibilité logique**.

B. Exemple de Sélection de Module

Pour illustrer cette approche, considérons un scénario où un utilisateur formule la requête suivante : « Peux-tu me réserver un vol pour Barcelone la semaine prochaine ?». Le système d'extraction sub-symbolique, reposant sur un transformeur, génère un embedding $\mathbf{x}_{\text{user}}$ qui sera naturellement proche dans l'espace vectoriel du vecteur typique associé au module « FlightBooking » (réservation de vols). Parallèlement, le bloc logique $\{R_{\text{user}}\}$ extrait des indices explicites comme "vol", "destination" et "date", tandis que le module « FlightBooking » dispose d'un embedding $\mathbf{x}_j$ spécifique et d'un ensemble de règles R_j définissant, par exemple, que la présence simultanée d'une destination et d'une date déclenche un scénario de réservation. La synergie sub-symbolique, calculée par une mesure telle que le **cosinus de similarité** ou un noyau gaussien, sera élevée si la sémantique de la requête correspond bien à celle du module, tandis que la synergie symbolique validera la cohérence entre les indices extraits. La combinaison des deux permet ainsi au DSL de sélectionner le module « FlightBooking » et de justifier cette décision par l'argumentation suivante :

« La similarité sub-symbolique indique que vous parlez de billets et de dates, et la logique détecte la présence de la destination 'Barcelone' ainsi que d'une date 'la semaine prochaine', ce qui correspond aux règles du scénario de réservation. »

C. Avantage en Explicabilité et Apprentissage Continu

L'intégration de ces deux composantes confère à l'agent conversationnel une **explicabilité** accrue. En effet, la partie sub-symbolique, tirée d'un modèle neuronal, est capable de capter des variations subtiles et de s'adapter continuellement grâce à des techniques de fine-tuning sur de nouveaux corpus. En parallèle, le **bloc logique** offre une justification transparente en exposant les règles partagées ou les axiomes utilisés pour interpréter la requête, ce qui est particulièrement précieux dans des domaines sensibles. Lorsqu'un nouveau scénario de dialogue est introduit (par exemple, la gestion d'abonnements), il est possible d'ajouter un nouvel embedding $\mathbf{x}_{\text{new}}$ et un bloc de règles R_{new} associé. Cette mise à jour se répercute sur la fonction de synergie hybride, permettant au système de s'adapter en temps réel tout en maintenant une architecture explicable. De plus, si une contradiction symbolique apparaît ou si la partie sub-symbolique présente une confusion, le système peut activer des mécanismes de rétroaction (feedback) pour réviser les règles ou affiner les embeddings, garantissant ainsi une **adaptation continue** sans perte de lisibilité.

D. Conclusion

L'exemple d'un agent conversationnel illustrant l'approche hybride d'un DSL montre comment la fusion de la **partie neuronale** (embeddings) et de la **partie logique** (règles et ontologies) permet de concevoir un chatbot capable non seulement de détecter l'intention de l'utilisateur via la **proximité vectorielle**, mais également de justifier sa sélection de scénario par une **cohérence explicite**. Ce mécanisme de fusion, incarné par la fonction de synergie

$$S_{\text{hybrid}}(\mathcal{E}_{\text{user}}, \mathcal{E}_j) = \alpha\, S_{\text{sub}}(\mathbf{x}_{\text{user}}, \mathbf{x}_j) + (1-\alpha)\, S_{\text{sym}}(R_{\text{user}}, R_j),$$

confère à l'agent une capacité d'**apprentissage continu** et d'**explicabilité**. Ainsi, le DSL hybride, en orchestrant ces deux dimensions, aboutit à un système conversationnel plus **intelligent** et plus **transparent**, capable de traiter des requêtes hétérogènes et de fournir des justifications explicites quant aux décisions prises, tout en évoluant pour s'adapter aux nouveaux flux de données et scénarios.

3.6. Aspects Avancés et Études de Cas

Les sections précédentes ont détaillé les **fondements** de la représentation des entités (chap. 3.2, 3.3, 3.4, 3.5), qu'elles soient sub-symboliques, symboliques ou hybrides. Dans cette partie (3.6), nous abordons des **perspectives** plus poussées et des **exemples concrets** illustrant l'étendue du **DSL** (Deep Synergy Learning) lorsqu'il s'attaque à des configurations complexes :

- Les **hypergraphes** et la **synergie n-aire** (3.6.1) : on dépasse le lien binaire pour relier un **ensemble** d'entités, ouvrant la voie à des interactions collectives (image–son–texte, par exemple).
- Les **structures fractales** (3.6.2) : la représentation elle-même peut devenir **multi-échelle** ou **self-similar**, en lien avec la fractalité (chap. 6) ; une manière d'organiser des données en hiérarchie d'emboîtements.
- Des **études de cas** (3.6.3) : applications pratiques (analyse audio-visuelle, agent conversationnel, robotique sensorielle) démontrant comment combiner embeddings et logiques.
- Une **comparaison expérimentale** (3.6.4) : mesurer l'influence du type de représentation (sub, sym, hybride) sur la qualité de la synergie, la vitesse de convergence et la robustesse.

3.6.1. Hypergraphes et Synergie n-aire

Jusqu'ici, la plupart des exemples de **synergie** ont porté sur des **paires** (i, j). Mais un **DSL** peut aller plus loin en considérant des **interactions collectives** entre $\{i_1, i_2, \dots, i_k\}$. Au lieu de ne mesurer que $S(i, j)$, on définit une **hyper-synergie** $S(\{i_1, \dots, i_k\})$ qui évalue la compatibilité (sub-symbolique, symbolique, ou mixte) d'un groupe d'entités dans son ensemble.

3.6.1.1. Au-delà du Lien Binaire, Possibilité de Relier $\{i_1, i_2, \dots, i_k\}$ par une "Hyper-Synergie"

Dans la plupart des *Synergistic Connection Networks* (SCN), l'architecture se fonde sur des liaisons binaires $\omega_{i,j}$ entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$. Cette structure est suffisante lorsque l'on se limite à des interactions par paires, mais on se heurte à des limites dès lors que certains phénomènes ou combinaisons essentielles ne prennent leur sens qu'en considérant plusieurs entités simultanément. C'est le cas, par exemple, dans des données multimodales (texte + audio + image), où la cohérence globale ne s'exprime pas par la somme ou la moyenne de trois liaisons binaires, mais réellement par la compatibilité collective des trois modalités.

Pour saisir de telles interactions d'ordre supérieur, on peut s'éloigner du simple concept d'arêtes binaires et se tourner vers le formalisme des *hypergraphes*, où une *hyper-arête* relie un sous-ensemble de nœuds $\{i_1, \dots, i_k\}$. Le SCN se dote alors d'**hyper-liens** dotés d'un *score* ou d'un "hyper-synergie" :

$$S(\{i_1, \dots, i_k\}) \in \mathbb{R}^+.$$

Ce score rend compte d'une cohérence globale non réductible aux simples paires. Il peut s'agir, par exemple, d'une fonction

$$S_k(\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_k}),$$

si l'on se situe dans un registre sub-symbolique (des embeddings multiples), ou encore d'une compatibilité logique *n*-aire, si l'on manipule des blocs de règles. Dans un SCN où l'on mettrait à jour les pondérations, on n'aurait plus seulement $\omega_{i,j}$ à itérer, mais potentiellement un poids $\omega_{(i_1,\dots,i_k)}$ pour chaque hyper-arête, reflétant la force de l'interaction d'ordre k.

Un premier exemple surgit dans la *fusion multimodale* : un *triplet* {texte, image, audio} n'a de sens que collectivement (une légende d'image, une bande-son, un contexte verbal), et la validité ou la cohérence de l'objet multimédia se repère à l'échelle du triplet. Un SCN classique reliant texte–image d'un côté, image–audio de l'autre, etc., ne traduirait pas la véritable interdépendance à trois entités.

Un autre exemple, dans le domaine de la **maintenance prédictive**, concerne des **capteurs multiples** où la détection d'une alarme requiert que trois capteurs, comme la température, la pression et la vibration, dépassent certains seuils simultanément. Une hyper-synergie $\omega_{(i_1,i_2,i_3)}$ se voit alors renforcée dès que l'on observe ce triple alignement.

Par ailleurs, certaines *communautés* ou *clusters* n'émergent que via des liens d'ordre supérieur. Dans la biologie, on peut étudier un système où l'interaction n'est pas captée par la somme de couples, mais par la triple ou quadruple co-occurrence de gènes ou de protéines.

Sur le plan algorithmique, étudier toutes les combinaisons de taille k dans un ensemble de n entités revient à considérer $\binom{n}{k}$ sous-ensembles, ce qui en $O(n^k)$ peut devenir prohibitif dès que $k > 2$ ou que n est grand. Les mises à jour itératives du SCN, ou l'évaluation de la synergie sur toutes ces hyper-arêtes, risquent de se révéler inenvisageables à grande échelle.

Pour conserver la **puissance** de l'hyper-synergie tout en évitant une explosion combinatoire, on adopte des heuristiques ou des restrictions où **limiter la taille de l'hyper-synergie** à 3 ou 4 entités maximum permet d'éviter une complexité inutile si l'utilité n'augmente plus au-delà. **Filtrer** via un score pair à pair avant de constituer des triplets ou quadruplets garantit que l'on ne forme un hyper-lien $\{i, j, k\}$ que si $\omega_{i,j}, \omega_{j,k}, \omega_{i,k}$ dépassent un certain seuil. Enfin, appliquer des techniques de *sampling* ou de *randomisation* permet d'échantillonner parmi les combinaisons multiples afin d'éviter de tester chaque ensemble $\{i_1, \dots, i_k\}$.

Si l'on introduit un poids $\omega_{(i_1,\dots,i_k)}$, la mise à jour prend une forme :

$$\omega_{(i_1,\dots,i_k)}(t+1) = \omega_{(i_1,\dots,i_k)}(t) \; + \; \eta \left[S(\{i_1, \dots, i_k\}) \; - \; \tau\, \omega_{(i_1,\dots,i_k)}(t)\right].$$

Chacune de ces hyper-arêtes se renforce ou s'affaiblit selon la "cohérence globale" du sous-groupe $\{i_1, \dots, i_k\}$. Les entités concernées peuvent se retrouver **fortement** reliées en tant que *tuplet*, alors même que la somme des liaisons binaires prises deux à deux ne le suggérait pas forcément. On obtient donc une *structure hypergraphique* potentiellement plus riche et plus adaptée à certaines *relations collectives*, au détriment d'une gestion plus complexe (tant dans la mise en œuvre que dans la lecture du résultat).

3.6.1.2. Représentation Plus Complexe : Capturer un “Ensemble” d’Entités Conjoints (par exemple, un Triplet Image–Texte–Audio)

Lorsqu’un *Synergistic Connection Network* (SCN) doit traiter une interaction qui ne se limite plus à de simples paires (i, j), il peut être nécessaire de considérer un *ensemble* plus vaste, comme $\{i_1, i_2, \dots, i_k\}$. Cette exigence apparaît notamment dans des scénarios multimodaux, où plusieurs entités doivent être observées conjointement (par exemple, un triplet image–texte–audio). Dans le cas d’un triplet, l’auto-organisation ne peut pas se contenter de lier séparément image–texte, texte–audio et audio–image ; une **synergie** réellement *n-aire* est requise pour évaluer la *cohérence globale* de l’ensemble.

Considérons la situation d’un article ou d’un post combinant une **image**, un **texte** descriptif et un **segment audio**. Si l’on se borne à analyser les paires (image–texte, texte–audio, audio–image), on peut rater des aspects qui n’apparaissent qu’en considérant simultanément les trois modalités. Une contradiction, par exemple, peut se glisser dans l’interaction globale (image et audio se rapprochent, texte est proche de l’audio, mais la combinaison image + texte ne correspond pas). Une vision strictement binaire ne détectera pas forcément cette incohérence, tandis qu’une **hyper-synergie** {image, texte, audio} peut en faire ressortir la contradiction ou, au contraire, la cohérence collective.

Dans une représentation hypergraphique, on remplace les arêtes binaires par des *hyperarêtes* connectant plusieurs nœuds $\{i_1, i_2, \dots, i_k\}$, chacune dotée d’un *score* ou *poids* $\omega_{\{i_1,\dots,i_k\}}$. On peut alors définir une **hyper-synergie**

$$S(\{i_1, i_2, \dots, i_k\}) \in \mathbb{R}^+,$$

qui exprime la cohérence *n*-aire. Dans le cas d’un triplet {img, txt, aud}, on peut, par exemple, construire un vecteur de *fusion* $\mathbf{z}_{\text{joint}} = \text{Fusion}(\mathbf{x}_{\text{img}}, \mathbf{x}_{\text{txt}}, \mathbf{x}_{\text{aud}})$ pour la composante sub-symbolique. Et si chacune des entités possède des **règles** ou attributs symboliques $\{R_{\text{img}}, R_{\text{txt}}, R_{\text{aud}}\}$, on agrège ces blocs en un ensemble *conjoint* $R_{\text{joint}} = \bigcup_{\ell=1}^{3} R_{i_\ell}$. Il est alors possible d’évaluer :

$$S(\{\text{img, txt, aud}\}) = \alpha\, S_{\text{sub}}\left(\text{Fusion}(\mathbf{x}_{\text{img}}, \mathbf{x}_{\text{txt}}, \mathbf{x}_{\text{aud}})\right) + (1-\alpha)\, S_{\text{sym}}\left(\text{Union}(R_{\text{img}}, R_{\text{txt}}, R_{\text{aud}})\right).$$

Si cette *hyper-synergie* est élevée, le SCN conclut à un ensemble multimodal cohérent ; dans le cas contraire, il identifie une *incohérence*.

Le passage à des hyper-synergies *n*-aires confronte le SCN à une explosion combinatoire où, pour des ensembles de n entités, tester toutes les combinaisons de taille k se chiffre en $\binom{n}{k}$. Même pour un triplet, cela peut représenter $O(n^3)$ d’hyperarêtes. Des heuristiques de **filtrage** s’avèrent alors essentielles où l’on ne retient que les ensembles $\{i_1, \dots, i_k\}$ pour lesquels les paires (i_p, i_q) dépassent un certain seuil de synergie binaire, ou bien l’on restreint la taille k à 3 ou 4 au maximum afin de contenir la complexité.

Lorsque l'on dispose d'une pondération $\omega_{\{i_1,\dots,i_k\}}$, la règle d'itération dans un SCN hybride devient :

$$\omega_{\{i_1,\dots,i_k\}}(t+1) = \omega_{\{i_1,\dots,i_k\}}(t) + \eta\left[S(\{i_1,\dots,i_k\}) - \tau\,\omega_{\{i_1,\dots,i_k\}}(t)\right].$$

Cette équation se passe désormais au niveau de l'hyperarête (ou "nœud conjugué"). Dans des applications multimodales, si les trois éléments image, texte et audio forment vraiment un ensemble cohérent, le *poids* $\omega_{\text{img,txt,aud}}$ s'accroît, reflétant l'identification d'un "micro-cluster" *n*-aire.

3.6.1.3. Implications : Algorithmes Plus Lourds, mais Patterns Plus Riches

La transition d'une synergie strictement binaire $\big(S(i,j)\big)$ à une **hyper-synergie** $\big(S(\{i_1,\dots,i_k\})\big)$ engendre un bouleversement considérable dans l'architecture et la dynamique d'un *Synergistic Connection Network* (SCN). Si les liens binaires suffisent à décrire la plupart des interactions entre deux entités, il existe des configurations où la cohérence d'un ensemble complet $\{i_1,\dots,i_k\}$ n'est pas réductible à la somme de paires, notamment dans des contextes multimodaux (image–texte–audio) ou lorsqu'un événement résulte d'un groupe de capteurs se déclenchant simultanément. Cet enrichissement se paie toutefois par une **charge algorithmique** et une gestion plus complexes, qu'il est essentiel d'analyser pour un DSL (Deep Synergy Learning) de grande dimension.

Le principal défi provient de la **combinatoire**. Dans un réseau de n entités, considérer tous les sous-ensembles de taille k impose d'en examiner $\binom{n}{k}$, ce qui peut se chiffrer en $O(n^k)$. Le simple fait de passer de $k = 2$ (paires) à $k = 3$ (triplets) fait déjà croître le nombre d'entités à évaluer de façon significative, et la situation s'aggrave pour des ordres supérieurs. Cela rend l'**exploration exhaustive** pratiquement impossible pour de grands n. Sur le plan du DSL, qui met à jour de façon itérative un score (pondération) lié aux synergies, on ne peut plus se contenter de vérifier $\omega_{i,j}$ pour toutes paires (i,j). Il faut dorénavant gérer $\omega_{\{i_1,\dots,i_k\}}$ pour les hyper-arêtes, et la recalculer chaque fois qu'une entité $\mathcal{E}_{i_\ell}$ modifie ses attributs (embedding, règles). L'effort de réévaluation peut vite s'avérer prohibitif. Des approches **heuristiques** émergent alors, consistant à filtrer ou restreindre le champ d'examen aux groupes considérés comme prometteurs.

Un hyper-lien $\omega_{(i_1,\dots,i_k)}$ s'actualise selon un schéma dérivé de la dynamique DSL habituelle :

$$\omega_{(i_1,\dots,i_k)}(t+1) = \omega_{(i_1,\dots,i_k)}(t) + \eta\left[S(\{i_1,\dots,i_k\}) - \tau\,\omega_{(i_1,\dots,i_k)}(t)\right].$$

La fonction de synergie $S(\{i_1,\dots,i_k\})$ peut être défini dans un cadre sub-symbolique (fusion des embeddings $\mathbf{x}_{i_1},\dots,\mathbf{x}_{i_k}$ et évaluation par un réseau) ou sémantique (compatibilité globale des blocs logiques $R_{i_1},\dots,R_{i_k}$), ou encore un mélange des deux. Sitôt qu'une seule entité $\mathcal{E}_{i_\ell}$ change son embedding ou ses axiomes, l'hyper-synergie se retrouve possiblement obsolète, imposant un recalcul si l'on ne met pas en place un dispositif pour restreindre son impact.

Ce surcroît de complexité, s'il représente une entrave opérationnelle, fait aussi la force d'une synergie n-aire. Il est désormais possible de détecter des *patterns* ou des phénomènes collectifs absents du spectre binaire. Dans un triplet {image, texte, audio}, la cohérence globale n'est plus strictement la somme ou la moyenne des relations deux à deux, mais véritablement un effet conjoint. De même, un ensemble de capteurs $\{\text{capteur}_1, \text{capteur}_2, \dots\}$ peut signaler un événement

critique seulement si tous se déclenchent en même temps. La synergie n-aire permet alors une modélisation plus fidèle et plus "haut niveau" de ces scénarios.

Au plan de l'implémentation, on recourt souvent à des **stratégies de filtrage** visant à n'explorer que les sous-ensembles d'entités ayant déjà prouvé une certaine proximité binaire (chacun des couples (i_ℓ, i_m) doit dépasser un seuil de similarité). Cette méthode de "candidats par paires" réduit la quantité d'hyperarêtes à évaluer en détail. Il est également envisageable d'employer des procédures d'**exploration incrémentale** (où l'on agrège peu à peu des entités sur la base d'un gain de synergie) ou des techniques de **recuit simulé** pour éviter l'énumération systématique de $\binom{n}{k}$.

La possibilité de mettre à jour un poids $\omega_{(i_1,\ldots,i_k)}$ a, dans un **Synergistic Connection Network**, d'importantes répercussions où certains **hyper-lots** d'entités, correspondant à des groupements *n*-aires, vont se renforcer et former des "hyper-clusters". On peut par exemple voir émerger un "triple cluster" $\{\mathrm{img}_1, \mathrm{txt}_5, \mathrm{aud}_3\}$ fermement relié par $\omega_{1,5,3}$ si ces trois modalités s'accordent fortement. Cette structure n'est pas réductible à un graphe usuel, puisqu'il faut appréhender la cohésion collective. Un aspect intéressant survient en logique symbolique, où la vérification d'une cohérence globale $\{R_{i_1}, \ldots, R_{i_k}\}$ peut nécessiter des tests complexes (cohérence n-aire, potentiellement NP-complets selon la expressivité logique). Tout l'enjeu consiste à contrôler la croissance de la complexité via un design parcimonieux (filtres, heuristiques, indices) et à tirer avantage de l'expressivité supplémentaire qu'offre une synergie n-aire.

3.6.2. Structures Fractales de Représentation

Au-delà des hypergraphes et de la synergie n-aire (section 3.6.1), une autre voie **avancée** pour organiser la représentation dans un **DSL** (Deep Synergy Learning) consiste à exploiter des **structures fractales** ou auto-similaires. L'idée est d'employer un **multi-niveau** ou un **multi-échelle** (en lien avec le chapitre 6) où la représentation de chaque entité (ou de chaque cluster) peut se répliquer ou s'itérer sous une forme **fractalement** organisée. Dans certains types de données, on observe en effet des **patterns** se répétant à différentes échelles — qu'il s'agisse d'images (motifs fractals), de taxonomies hiérarchiques (ontologies en forme d'arbre auto-similaire), voire d'embeddings qui se réorganisent récursivement.

3.6.2.1. Lien avec le chap. 6 (multi-échelle + fractalité) : la représentation elle-même peut s'avérer fractale (ex. embeddings récursifs, hiérarchie de concepts)

La perspective multi-échelle exposée au chapitre 6 souligne que, dans un **DSL** (*Deep Synergy Learning*), un *Synergistic Connection Network* peut se structurer en plusieurs *niveaux* ou *couches* successifs. Chaque niveau agrège ou récapitule les motifs détectés au niveau inférieur, créant ainsi un effet d'**auto-organisation hiérarchique**. Cet agencement à étages peut aller plus loin lorsque la représentation des entités exhibe un caractère **auto-similaire**, voire **fractal**, où à chaque nouvelle échelle, on retrouve un schéma semblable reproduisant le même mode d'organisation à des granularités différentes. Il en résulte la possibilité d'embeddings **récursifs** ou de hiérarchies conceptuelles répliquant la même forme à divers niveaux.

Sur le plan **sub-symbolique**, on peut concevoir que l'embedding $\mathbf{x}_i \in \mathbb{R}^d$ représentant une entité $\mathcal{E}_i$ se scinde en *mini-embeddings* itératifs, ce qui reflète une *structure fractale*. Dans certains domaines, comme la modélisation de textures ou de formes naturelles (ex. arbres, littoraux), les mêmes motifs géométriques se répètent à diverses résolutions. On formalise alors des schémas d'autoencodeurs profonds ou de CNN « en cascade », où l'opération $\mathbf{x}_i^{(\ell+1)} = F\left(\mathbf{x}_i^{(\ell)}\right)$ reproduit un *pattern* identique à chaque étape ℓ. Cette auto-similarité sous-tend une représentation fractale où le **DSL** peut l'exploiter pour repérer ou stabiliser des clusters multi-échelle, une sous-structure locale se retrouvant dans une macro-structure globale.

Sur le plan **symbolique**, les ontologies ou ensembles de règles peuvent, elles aussi, refléter une organisation fractale. Dans une ontologie de grande taille, il arrive qu'une arborescence de concepts se **duplique** partiellement dans diverses branches où le même **motif** notionnel réapparaît, générant des **sous-ontologies** répétant la même logique de liens comme subClassOf ou partOf. Un **DSL** hiérarchique peut alors reconnaître, au niveau micro, un groupement conceptuel, puis observer le *même* schéma à une échelle plus macro, renforçant la cohérence globale et la capacité de généralisation.

La dimension **multi-échelle** intervient lorsque l'on empile plusieurs *couches* dans le réseau, chacune gérant une échelle de représentation. On obtient un flux *ascendant* (bottom-up), où les motifs découverts localement se consolident, et un flux *descendant* (top-down), où les structures macro peuvent imposer une cohérence aux entités de rang inférieur. Dans un *cadre fractal*, cette interaction est d'autant plus fluide que la même forme **p** ou la même logique **r** se reproduit à chaque palier, ce qui évite de réinventer un nouveau schéma à chaque fois.

Mathématiquement, on peut décrire la fractalité dans le sous-espace $\mathbb{R}^d$ par une opération F appliquée de façon itérative à un point ou à une région, créant un ensemble invariant par F. Dans un **DSL**, si l'on modélise l'évolution d'un embedding $\mathbf{x}_i^{(\ell)}$ à mesure qu'on change de niveau ℓ, on peut écrire $\mathbf{x}_i^{(\ell+1)} = F\left(\mathbf{x}_i^{(\ell)}\right)$ pour une transformation donnée. Lorsque F est contractante, on obtient un *attracteur fractal*, dont la structure se reflète dans l'organisation multi-niveau des entités.

Le chapitre 6, centré sur l'apprentissage multi-échelle, se prête donc à accueillir ce genre de **représentation fractale** où, à chaque palier, l'**auto-organisation** exploite la même règle de mise à jour et la même forme d'ancrage, qu'elle soit sub-symbolique ou symbolique, mais appliquée à un degré de finesse différent. Ce principe peut renforcer l'efficacité du **DSL** dans des domaines où la notion de *self-similarité* est prépondérante (biologie, géologie, architecture conceptuelle, etc.).

3.6.2.2. Exemple : embeddings fractals pour des données "self-similar", ou ontologies fractales pour la taxonomie

La **fractalité** joue un rôle fondamental dès que l'on observe des motifs identiques ou très similaires qui se reproduisent à différentes **échelles**. Dans le cadre d'un **Deep Synergy Learning (DSL)**, où l'on cherche à organiser des données (vectorielles ou symboliques) de façon auto-similaire, l'exploitation de cette propriété fractale permet une **intégration** plus économe et plus robuste. Cette section illustre deux cas d'application où l'un concerne la construction d'**embeddings fractals**

pour des données intrinsèquement **self-similar** comme les images, signaux ou textures, tandis que l'autre traite des **taxonomies** ou **ontologies** présentant une structure auto-similaire récurrente.

Afin de souligner l'importance de ces approches, il convient de rappeler que la **dimension fractale** d'un objet F peut être définie, par exemple, via la formule usuelle de **self-similarité** où, si la figure est couverte par N copies d'elle-même à l'échelle $1/r$, alors

$$\dim_{\text{fractale}}(F) \;=\; \frac{\ln(N)}{\ln(r)}.$$

La réitération d'une même structure à différentes résolutions se révèle un facteur clé tant dans le cas **sub-symbolique** (données géométriques ou multi-échelle) que dans le cas **symbolique** (ontologies arborescentes). Cette auto-similarité offre un levier pour construire des **représentations** et des **mesures de similarité** adaptées au principe de fractalité.

A. Embeddings fractals pour des données self-similar

Dans de nombreux champs scientifiques, les données présentent un **comportement auto-similaire** où un motif se reproduit à plusieurs échelles, révélant une structure fractale sous-jacente. On rencontre cette propriété dans des *textures naturelles* (extraction de *patches* à différents niveaux de zoom), dans des *signaux biologiques* pouvant dévoiler la même forme d'onde à des fréquences multiples, ou dans certains *objets géométriques* (côtes maritimes, contours fractals). L'idée est alors de coder chaque entité $\mathcal{E}_i$ à l'aide d'un **embedding fractal**, c'est-à-dire d'un vecteur qui capture la redondance d'échelle.

Si l'on note $\mathbf{x}_i^{(l)}$ la description de $\mathcal{E}_i$ à l'échelle l (par exemple, la sortie d'un *patch-encoder* appliqué à un *patch* de résolution l), on peut concaténer ou fusionner ces descriptions sur plusieurs niveaux, produisant au final un **embedding** $\mathbf{x}_i \in \mathbb{R}^D$ qui agrège cette information multi-résolution. La **fractalité** de l'entité, exprimée par la répétition statistique du même motif à plusieurs granularités, se retrouve alors dans la structure même de $\mathbf{x}_i$.

Une formalisation plus mathématique introduit une fonction de type

$$\mathbf{x}_i \;=\; \left[f\left(\mathbf{x}_i^{(0)}\right), f\left(\mathbf{x}_i^{(1)}\right), \ldots, f\left(\mathbf{x}_i^{(L)}\right)\right],$$

où f est un encodeur non linéaire (par exemple un autoencodeur profond) appliqué à différentes résolutions. La **synergie** entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ peut alors se définir via une **similarité** (cosinus, noyau gaussien, etc.) entre leurs embeddings fractals respectifs :

$$S(\mathcal{E}_i, \mathcal{E}_j) \;=\; \exp\left(-\alpha \parallel \mathbf{x}_i - \mathbf{x}_j \parallel\right),$$

assurant que deux **configurations fractales** très proches (i.e. répétant le même motif à chaque niveau) se verront attribuer une **synergie** élevée. Dans un **SCN**, ces entités fortes en **self-similarité** locale formeront un **cluster** auto-organisé révélant leur communauté de forme.

B. Ontologies fractales pour la taxonomie

Si l'on se situe dans un registre plus symbolique ou conceptuel, la **fractalité** peut apparaître sous forme d'une *hiérarchie* ou *arborescence* qui se répète, quasi à l'identique, dans chaque sous-branche. De nombreuses *taxonomies* en biologie, chimie, ou dans des classifications industrielles,

reproduisent la même structure de relations et de propriétés à différents étages. On parle alors de **fractalité sémantique**, puisque la même architecture se manifeste à plusieurs niveaux de la hiérarchie, de sorte que l'on peut utiliser un *motif ontologique* unique pour chacune des sous-branches.

On peut modéliser une telle ontologie comme un graphe $\mathcal{G}$ muni d'une application récursive, par exemple

$$\phi\colon \mathcal{G} \rightarrow \mathcal{G} \times \{0, \ldots, k-1\}$$

qui identifie des *sous-graphes* isomorphes répliquant un même ensemble de classes ou de relations. Cette propriété est analogue à la fonction d'itération d'un fractal géométrique, mais appliquée aux liens sémantiques (*subClassOf*, *hasPart*, *connectedTo*, etc.).

Dans le cadre d'un **DSL**, une telle **auto-similarité** conceptuelle renforce la **cohérence** multi-niveau. La **mise à jour** des poids de synergie, notée $\omega_{ij}(t+1) = \cdots$, ou l'intégration d'une règle symbolique au niveau supérieur, peuvent automatiquement se propager aux étages fractals sous-jacents, réduisant l'effort de maintenance. Par ailleurs, la similarité sémantique entre deux *sous-domaines* peut se calculer en comparant leurs **motifs ontologiques**. Deux branches véhiculant des structures quasi identiques, à quelques variations de propriétés près, obtiendront une **distance** sémantique faible et, en conséquence, une **synergie** élevée dans le **SCN** symbolique.

C. Gains potentiels

L'approche **fractal-based** procure un **gain** en compression où un même schéma ou un même encodeur multi-résolution peut être réutilisé à diverses échelles. Elle amplifie aussi la **robustesse** où lorsqu'une révision touche la structure de base, elle s'applique à toutes les répliques, évitant ainsi les **incohérences locales**. Sur le plan purement mathématique, l'uniformisation de la **dimension fractale** ou la réplication d'un **pattern** identique signifient que la **synergie** ne fluctue pas brutalement d'un étage à l'autre, mais demeure gouvernée par un **principe** d'auto-similarité. Les mises à jour locales des poids ω_{ij} (ou l'apparition de nouvelles entités analogues) s'intègrent plus harmonieusement dans l'ensemble.

On peut inclure une *analyse dimensionnelle* pour vérifier que l'object fractal formé par la hiérarchie ou l'ensemble des embeddings est susceptible de présenter un degré de redondance réduisant la *complexité effective*. Ainsi, dans une ontologie auto-similaire, la **profondeur** L de la structure peut conditionner la **dimension** du schéma de liens, souvent inférieure à ce qu'on obtiendrait si chaque étage était dessiné de manière indépendante. Cette propriété apparaît clairement dans les constructions fractales pures, où la **taille** du système croit moins vite que l'**échelle** (puisqu'on recycle le même motif à chaque niveau).

3.6.2.3. Gains potentiels en compression ou en cohérence multi-niveaux

L'exploitation d'une **structure fractale** dans le cadre d'un **Deep Synergy Learning (DSL)** permet de renforcer à la fois la **compression** des descriptions et la **cohérence** entre différents niveaux d'analyse. Cette intégration repose sur l'idée que lorsqu'un *motif* se répète à diverses échelles, on peut le stocker ou le calculer une seule fois et le réutiliser de façon récurrente.

A. Compression et réduction de la redondance

Lorsqu'une représentation fractale ou une ontologie auto-similaire est mise en place, la **redondance** entre les différentes entités se matérialise par la récurrence du même schéma à plusieurs échelles. Supposons qu'un motif **m** se retrouve répliqué dans plusieurs branches d'un arbre ou à différents niveaux de résolution. Au lieu de décrire ce motif **m** chaque fois qu'il apparaît, on peut le factoriser dans une structure commune. Si le fractal possède un facteur d'échelle $\mathbf{r} > 1$ et qu'on retrouve le même motif dans N zones à chaque itération, la quantité totale d'information mémorisée peut se réduire à un *coût de base* plus faible que la somme naïve des contributions de chaque zone.

Une modélisation plus formelle peut invoquer une fonction de self-similarité. Si la structure globale $\mathcal{F}$ se décrit par un ensemble de copies de dimension α, on a $\dim_{\text{fractale}}(\mathcal{F}) = \alpha$, alors que la description complète (sans prise en compte de l'auto-similarité) conduirait à une dimension apparente plus grande. Dans un **SCN** (Synergistic Connection Network), un grand nombre de nœuds peuvent partager la même sous-structure interne ; la référence à cette sous-structure suffit alors, ce qui donne un **gain** en $O(\log n)$ ou $O(n^{\delta})$ (selon la fractalité) plutôt qu'en $O(n)$. D'un point de vue algorithmique, le recalcul des **synergies** $S(i,j)$ s'en trouve lui aussi allégé, car chaque motif commun **m** ne s'encode qu'une fois avant d'être propagé dans l'ensemble des entités concernées.

B. Cohérence multi-niveaux et synergie fractale

Les modèles fractals sont particulièrement adaptés à la logique **multi-niveau** qui sous-tend le DSL. Lorsque la même forme se réitère à différentes échelles, la **cohérence** entre niveaux local et global se voit considérablement renforcée. Une entité $\mathcal{E}_i$ peut se retrouver au niveau micro avec des liens $\omega_{i,j}(t)$ qui reproduisent, en "miniature", la structure générale observable au niveau macro.

Dès que l'on considère la mise à jour

$$\omega_{i,j}(t+1) \;=\; \omega_{i,j}(t) \;+\; \eta\,\big[S(i,j) \;-\; \tau\,\omega_{i,j}(t)\big],$$

on remarque que le schéma fractal induit un renforcement cohérent à chaque échelle. Si les entités $\mathcal{E}_i$ et $\mathcal{E}_j$ appartiennent à un motif reproduit dans plusieurs parties du réseau, la synergie $S(i,j)$ demeure élevée dans chacun de ces contextes, ce qui stabilise $\omega_{i,j}$ de manière récursive. Les mêmes **clusters** ou agrégations se retrouvent ainsi à divers niveaux de granularité, ce qui unifie l'auto-organisation.

Sur le plan conceptuel, cette **auto-similarité** multi-niveau peut être rapprochée d'un **zoom fractal** où l'on peut se déplacer entre une échelle fine et une échelle globale tout en constatant la même topologie répliquée. Le **DSL** bénéficie alors d'une stabilité accrue, car la redondance des liaisons synergiques évite les contradictions entre un niveau et l'autre.

C. Stabilité et scalabilité dans les applications fractales

La dimension fractale sert également de garant de **stabilité** et de **scalabilité** pour le DSL. Dans des domaines comme la **géologie**, où les images présentent des textures fractales, on peut adopter un embedding fractal afin de capturer la redondance d'échelle et réduire la dimension effective des données. Dans l'**ontologie** d'un système complexe, on peut dupliquer un même module conceptuel pour chaque subdivision, ce qui préserve la cohérence de la base de connaissances. On formalise

parfois cette récurrence par une application Φ qui envoie chaque branche vers un motif identique mais étiqueté différemment,

$$\Phi\colon \mathcal{G} \;\to\; \bigsqcup_{\ell=1}^{k} \mathcal{G}_\ell,$$

là où $\mathcal{G}_\ell$ est une copie partiellement isomorphe de $\mathcal{G}$.

Ce principe accroît la **scalabilité** du réseau, car l'augmentation de la taille (nombre de nœuds ou de classes) n'exige pas de réinventer la structure de liens, mais simplement d'appliquer la transformation Φ ou d'exploiter les mêmes blocs encodés. Les **clusters** ou groupes de nœuds se forment ainsi de façon reproductible, même lorsque l'on déploie une ontologie à plusieurs milliers de branches. Le rapport entre la complexité globale et la profondeur des niveaux demeure raisonnable, en raison de la répétition du *motif fractal* à chaque sous-niveau.

L'effet bénéfique sur la **stabilité** tient au fait qu'un motif validé localement engendre les mêmes propriétés au niveau global. Les paramétrages $\omega_{i,j}$ déjà convergés à un étage se transposent aux autres étages, conférant au **DSL** une robustesse face aux modifications et aux extensions. Un schéma local ne nécessite plus d'être redéfini dans toute la hiérarchie, ce qui évite les incohérences et limite les oscillations.

3.6.3. Études de Cas Illustratives

Après avoir exploré différentes **extensions** et **approches** avancées (hypergraphes n-aires, structures fractales, etc.), il est utile de **concrétiser** ces idées à travers quelques **cas d'usage** représentatifs. Dans cette section (3.6.3), nous décrirons trois scénarios où un DSL (Deep Synergy Learning) peut trouver une application pratique et mettre en œuvre les principes de représentation (sub-symbolique, symbolique, hybride) examinés tout au long du Chapitre 3 :

- (3.6.3.1) Un **système d'analyse audio-visuelle**, combinant des **vecteurs CNN** pour l'image, des **embeddings** pour l'audio et des **règles** logiques propres à un domaine spécifique (ex. surveillance, broadcast, etc.).
- (3.6.3.2) Un **agent conversationnel**, déjà évoqué dans des sections antérieures, qui fusionne des "règles" (cohérence contextuelle) et des embeddings textuels (ex. BERT).
- (3.6.3.3) Un **scénario de robotique sensorielle**, où la représentation sub-symbolique (capteurs multiples) cohabite avec une représentation symbolique décrivant les actions, les objets et la logique de manipulation.

3.6.3.1. Système d'analyse audio-visuelle : vecteurs CNN + embeddings audio + logiques domain-specific

Dans les applications de **surveillance**, d'**indexation multimédia** ou de **diagnostic** dans un environnement audiovisuel complexe, il est souvent crucial de fusionner une **information visuelle** (généralement encodée par un **réseau de neurones convolutifs**, CNN) et une **information sonore** (par exemple encodée via des **embeddings audio**), tout en les contraignant par des **règles logiques**

spécifiques au domaine. Le **Deep Synergy Learning (DSL)**, et plus particulièrement son réseau **SCN** (Synergistic Connection Network), offre un cadre systématique pour opérer cette fusion et détecter des **événements** ou **patterns** pertinents.

Il est d'usage de représenter chaque segment temporel issu de la **vidéo** par un vecteur $\mathbf{x}_{\text{vid}} \in \mathbb{R}^{d_1}$, obtenu à l'aide d'un **CNN** pré-entraîné ou d'un autre modèle **sub-symbolique**. Parallèlement, chaque segment temporel **audio** se voit attribuer un vecteur $\mathbf{x}_{\text{aud}} \in \mathbb{R}^{d_2}$, issu d'un traitement du signal tel qu'un **spectrogramme** encodé ou un modèle **transformer audio**. Une procédure de **fusion** des deux composantes, notée

$$\mathbf{x}_{\text{joint}} \;=\; F_{\text{fusion}}(\mathbf{x}_{\text{vid}},\; \mathbf{x}_{\text{aud}}),$$

peut se contenter d'une **concaténation** ou employer des techniques plus avancées (attention croisée, somme pondérée). La dimension symbolique intervient à travers un ensemble de **règles** ou d'**axiomes** propres au domaine où, par exemple, "Si une zone interdite est franchie et qu'un son d'alarme est détecté, alors alerte." On peut modéliser ces règles par des **propriétés logiques** regroupées dans un bloc $\mathbf{R}_{\text{vid}}$ pour la partie vidéo, et $\mathbf{R}_{\text{aud}}$ pour la partie audio, ou bien un bloc commun si des règles globales couvrent l'ensemble du signal.

La **fonction de synergie** doit alors refléter à la fois la proximité sub-symbolique (cohérence **audio-visuelle**) et la compatibilité des **règles** logiques (compatibilité sémantique). Afin d'agréger ces deux critères, on définit

$$S_{\text{hybrid}}(\mathcal{E}_{\text{vid}},\, \mathcal{E}_{\text{aud}}) \;=\; \alpha\, S_{\text{sub}}(\mathbf{x}_{\text{vid}},\, \mathbf{x}_{\text{aud}}) \;+\; (1-\alpha)\, S_{\text{sym}}(\mathbf{R}_{\text{vid}},\, \mathbf{R}_{\text{aud}}).$$

La composante sub-symbolique S_{sub} peut être une **similarité** exponentielle sur la norme de la différence,

$$S_{\text{sub}}(\mathbf{x}, \mathbf{y}) \;=\; \exp(-\beta \;\; \| \, \mathbf{x} - \mathbf{y} \, \|),$$

ou tout autre indicateur mesurant à quel point la **signature visuelle** et la **signature audio** coïncident (corrélation temporelle, alignement spectral, etc.). La composante symbolique S_{sym} se fonde sur la logique du domaine où, si les axiomes audio et vidéo ne se **contredisent** pas, et mieux encore s'ils **confirment** l'un l'autre, alors $S_{\text{sym}}(\mathbf{R}_{\text{vid}},\, \mathbf{R}_{\text{aud}}) \approx 1$. En revanche, une contradiction explicite (par exemple, "alarmeActivée=TRUE" côté audio mais "alarmeInopérante=TRUE" côté vidéo) fait chuter drastiquement le score symbolique.

Le **SCN** peut ensuite **auto-organiser** les segments audio-visuels au moyen d'une **règle de mise à jour** itérative :

$$\omega_{(\text{vid,aud})}(t+1) \;=\; \omega_{(\text{vid,aud})}(t) \;+\; \eta \left[S_{\text{hybrid}}(\mathcal{E}_{\text{vid}},\, \mathcal{E}_{\text{aud}}) \;-\; \tau\, \omega_{(\text{vid,aud})}(t) \right].$$

Un lien $\omega_{(\text{vid,aud})}$ se voit **renforcé** (croît progressivement) si la synergie audio-vidéo reste élevée ; dans le cas contraire, il décroît. Après plusieurs itérations, ce mécanisme local de **plasticité** fait apparaître des **clusters** de segments fortement reliés, correspondant à des événements ou situations reconnues.

Pour décider d'une **alerte** ou d'une **étiquette** associée à un intervalle temporel, on peut fixer un **seuil** $\omega_{\min}$ où, dès que $\omega_{(\text{vid,aud})}$ dépasse $\omega_{\min}$, le **réseau** considère la relation (vid, aud) comme un événement fusionné. La logique additionnelle (par exemple, "intrusion interdite + alarme =>

situation critique") permet alors de produire un **verdict** plus précis, tant en termes de classification (type d'incident) que d'**explicabilité** (la logique indique pourquoi il y a match entre la vidéo et l'audio).

Ce **système multimodal** s'avère particulièrement robuste. Si la **vidéo** se trouve partiellement brouillée (mauvaise luminosité, occlusions), la **composante sonore** peut prendre le relai et maintenir un niveau élevé de synergie. Symétriquement, une détérioration du signal audio peut être compensée par l'analyse visuelle. Dans des applications réelles comme la **surveillance** en zones sensibles ou la **détection d'incidents** tels que les pannes, alarmes, cris ou collisions, l'association d'un bloc de **règles symboliques** vient enrichir la détection où l'on ne se contente pas d'une simple ressemblance sub-symbolique, mais on vérifie aussi la **compatibilité logique**. La structure **DSL** autorise aussi une **adaptation continue** où l'**embedding** vidéo ou audio peut être réentraîné sur de nouvelles données tandis que les **règles** du domaine se mettent à jour, qu'il s'agisse de nouvelles situations réglementaires ou de nouveaux types d'alarmes, le tout sans remettre en cause l'ensemble des acquis du **SCN**.

3.6.3.2. **Agent conversationnel : combine "règles" pour la cohérence contextuelle + embeddings BERT**

Dans le cadre d'un **Deep Synergy Learning (DSL)**, la construction d'un **agent conversationnel** (chatbot) repose sur l'intégration d'une dimension **sub-symbolique** et d'une dimension **symbolique**. L'objectif est de combiner, d'une part, la puissance d'un modèle d'**embeddings** capable de saisir la richesse et la variabilité du langage naturel – en l'occurrence, des représentations contextuelles produites par un modèle comme **BERT** (*Bidirectional Encoder Representations from Transformers*) – et, d'autre part, un ensemble de **règles logiques** qui garantissent la cohérence contextuelle et la maîtrise du dialogue. Cette approche hybride permet ainsi à l'agent d'analyser de multiples formulations textuelles tout en respectant une séquence de *slots* ou de conditions qui encadrent la progression du dialogue.

A. ***Composante sub-symbolique : embeddings BERT***

Lorsqu'un utilisateur énonce une phrase ou un segment textuel, le DSL fait appel à **BERT** pour générer un **embedding** qui représente la sémantique de l'énoncé. On définit ainsi, pour une entité textuelle $\mathcal{E}_i$,

$$\mathbf{x}_i = \mathrm{BERT}(\mathcal{E}_i) \in \mathbb{R}^d,$$

où d représente la dimension de l'espace des embeddings. Ce vecteur $\mathbf{x}_i$ encapsule les informations contextuelles et syntaxiques de la phrase. Par exemple, deux phrases exprimant une intention similaire (même signification même si formulées différemment, malgré des fautes ou l'usage de synonymes) conduiront à des vecteurs dont la **similarité cosinus** sera élevée, exprimée mathématiquement par

$$S_{\mathrm{sub}}(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\| \mathbf{x}_i \| \| \mathbf{x}_j \|}.$$

La robustesse de BERT, particulièrement lorsqu'il est **fine-tuned** sur des corpus spécifiques à un domaine (assurance, banque, médical, etc.), garantit que même des variations linguistiques (fautes,

abréviations) n'altèrent pas significativement la capture de l'intention. Ce niveau de détail permet au DSL de comparer de manière fine les intentions des utilisateurs et de sélectionner le module de dialogue le mieux adapté, en se fondant sur le **rapprochement sémantique** des embeddings.

B. Composante symbolique : règles de cohérence contextuelle

Outre la représentation sub-symbolique, un agent conversationnel doit maintenir une **cohérence logique** dans le déroulement du dialogue. Pour ce faire, le DSL intègre un ensemble de **règles logiques** regroupé dans un bloc R_i associé à chaque entité ou état de dialogue. Ces règles, élaborées en fonction des contraintes du domaine, définissent des conditions d'activation ou de transition. Par exemple, dans un scénario de réservation d'hôtel, il est possible de spécifier que l'utilisateur doit d'abord fournir la date et le lieu avant de passer à la sélection du mode de paiement. Formellement, la **synergie symbolique** peut être représentée par une fonction qui attribue un score en fonction de la conformité des règles, telle que

$$S_{\text{sym}}(R_i,\, R_j) = \begin{cases} 1, & \text{si les règles se valident mutuellement,} \\ 0, & \text{en cas de contradiction totale,} \\ s_{ij}, & \text{pour une satisfaction partielle,} \end{cases}$$

où $s_{ij} \in (0,1)$ représente un score intermédiaire mesurant le degré de compatibilité entre les blocs de règles associés aux entités $\mathcal{E}_i$ et $\mathcal{E}_j$. Cette approche assure que l'agent conversationnel ne franchit jamais des étapes incohérentes, préservant ainsi la logique du dialogue et la non-répétition de questions redondantes.

C. Calcul de la Synergie Mixte et Mise à Jour des Pondérations

La force du DSL réside dans l'intégration des deux composantes précédentes afin de définir une **synergie hybride** qui guide la mise à jour des **pondérations** dans le SCN. La synergie hybride, qui évalue simultanément la **proximité sémantique** et la **compatibilité logique**, est formulée comme suit :

$$S_{\text{hybrid}}(i,j) = \alpha\, S_{\text{sub}}(\mathbf{x}_i,\, \mathbf{x}_j) + (1-\alpha)\, S_{\text{sym}}(R_i,\, R_j),$$

où le paramètre $\alpha \in [0,1]$ ajuste l'importance relative accordée à la composante sub-symbolique par rapport à la composante symbolique. La **mise à jour** des pondérations du SCN se fait alors selon la règle

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[S_{\text{hybrid}}(i,j) - \tau\, \omega_{i,j}(t)\big],$$

où η représente le **taux d'apprentissage** et τ le **coefficient de décroissance**. Cette équation traduit la **plasticité locale** du système où, lorsque la synergie hybride entre deux entités est élevée, indiquant que leurs intentions sont à la fois **sémantiquement proches** et **logiquement cohérentes**, la pondération $\omega_{i,j}$ est renforcée, favorisant ainsi la formation d'enchaînements de dialogue pertinents. Inversement, des valeurs faibles de $S_{\text{hybrid}}(i,j)$ conduisent à un affaiblissement des liaisons, évitant ainsi des transitions incohérentes.

*D. **Bénéfices et Retours d'Expérience***

L'approche hybride présente plusieurs **avantages** clés. La **robustesse linguistique** est assurée grâce à BERT, qui, par son fine-tuning, permet de gérer efficacement les variations dans les formulations textuelles, qu'il s'agisse de fautes d'orthographe, d'abréviations ou de synonymes. La **cohérence** du dialogue est maintenue par l'intégration d'un ensemble de **règles logiques** qui structurent la progression du dialogue et garantissent que les transitions respectent des contraintes préétablies. Par ailleurs, la combinaison de ces deux dimensions permet d'atteindre une **explicabilité** accrue, car chaque décision d'enchaînement peut être justifiée à la fois par une correspondance sémantique (le score sub-symbolique élevé) et par la validation d'un ensemble de règles (le score symbolique). Enfin, l'**évolutivité** du système est favorisée puisque le SCN, par le biais de la mise à jour continue des pondérations, s'adapte progressivement aux nouvelles données et aux révisions des règles, assurant une continuité du dialogue tout en permettant une mise à jour fluide des modules. En pratique, ces avantages se traduisent par un agent conversationnel capable de gérer des requêtes complexes et variées tout en maintenant un fil de dialogue cohérent et explicable, ce qui est particulièrement pertinent dans des domaines spécialisés.

3.6.3.3. Robotique sensorielle : capteurs sub-symboliques + représentation symbolique des actions/objets

L'association de **capteurs multiples** (caméra, LiDAR, capteur de force, etc.) avec un **ensemble de règles** décrivant les actions et les objets manipulés constitue un scénario riche pour le **Deep Synergy Learning (DSL)**. Le robot, plongé dans un environnement parfois incertain (lumière changeante, bruit sur les mesures), doit à la fois **analyser** les informations sub-symboliques fournies par ses capteurs et **respecter** la logique imposée par ses règles (scripts, protocoles, axiomes de sécurité). Cette double contrainte façonne un réseau **SCN** où chaque état ou action intègre à la fois un **embedding sub-symbolique** et un **bloc** de conditions symboliques, et où la **synergie** commande la mise à jour adaptative des liens.

A. Contexte : un robot multisensoriel

Imaginez un **robot logistique** évoluant dans un entrepôt, équipé de multiples capteurs :

- **Caméra** fournissant un flux d'images. Un **CNN** (ou réseau de vision spécialisé) produit un vecteur $\mathbf{x}_{\text{cam}} \in \mathbb{R}^{d_1}$ décrivant l'information visuelle (objets détectés, caractéristiques de la scène).
- **LiDAR** fournissant un nuage de points 3D. Un encodeur (type *PointNet* ou *autoencodeur 3D*) calcule $\mathbf{x}_{\text{LiDAR}} \in \mathbb{R}^{d_2}$.
- **Capteur de force** détectant la pression exercée par un bras manipulateur, ou la force-torque globale, résumé dans un vecteur $\mathbf{x}_{\text{force}} \in \mathbb{R}^{d_3}$.

Par ailleurs, le robot se réfère à un **bloc symbolique** explicitant des règles de fonctionnement où "si la zone scannée est interdite, le robot doit contourner", "si l'objet est fragile, la force de préhension doit rester sous un certain seuil" et "si la distance est < 0.5 m et que l'objet est inconnu, refuser l'avancée".. Ces règles logiques forment un corpus **R**, scindé éventuellement entre celles

attachées à l'état (quelles conditions valent pour la position, la posture ?), et celles décrivant les actions (pré-requis pour saisir un objet, interdictions particulières, etc.).

Dans l'esprit d'un **DSL** hybride, chaque entité $\mathcal{E}_i$ peut être un **état** sensoriel, un **objet**, ou une **action** potentielle, muni à la fois :

– d'un **embedding** sub-symbolique $\mathbf{x}_i \in \mathbb{R}^d$ (obtenu, par exemple, en concaténant $\mathbf{x}_{\text{cam}}$, $\mathbf{x}_{\text{LiDAR}}$, $\mathbf{x}_{\text{force}}$ ou un encodeur fusionné),

– d'un **bloc** de règles symboliques $\mathbf{R}_i$, exprimant, par exemple, la classe d'objet (fragile, volumineux), des restrictions de mouvement, ou un protocole d'action (*pick*, *place*, *push*).

B. Synergie entre capteurs et règles d'action

La fusion sub-symbolique permet d'évaluer la **proximité** ou la **compatibilité** entre deux états sensoriels, en calculant une distance ou une similarité sur leurs embeddings $\mathbf{x}_{\text{state1}}$, $\mathbf{x}_{\text{state2}}$. Parallèlement, si l'on compare un état $\mathcal{E}_i$ et une action $\mathcal{A}$, la **synergie** inclut la vérification logique où l'on évalue si l'action $\mathcal{A}$ est autorisée dans l'état $\mathcal{E}_i$ et quelles conditions $R_{\mathcal{A}}$ doivent s'appliquer, par exemple "si l'objet est fragile, vérifier que la force < 10 N". On écrit :

$$S_{\text{hybrid}}(\mathcal{E}_i, \mathcal{A}) \;=\; \alpha\, S_{\text{sub}}(\mathbf{x}_i, \mathbf{x}_{\mathcal{A}}) \;+\; (1-\alpha)\, S_{\text{sym}}(R_i, R_{\mathcal{A}}).$$

Ici, $\mathbf{x}_i$ représente l'**embedding** agrégé de l'état sensoriel du robot (caméra, LiDAR, force), tandis que $\mathbf{x}_{\mathcal{A}}$ peut (ou non) exister selon la nature de l'action (certaines actions se dotent d'un vecteur sub-symbolique décrivant leurs contraintes). Les règles $\mathbf{R}_i$ et $R_{\mathcal{A}}$ reflètent la logique symbolique, comme "ne pas déplacer un objet fragile au-delà de 2 m de hauteur" ou "nécessite zone autorisée = TRUE".

Ce score de **synergie** sub-symbolique + symbolique conditionne l'**auto-organisation** dans le **SCN** où, si $\mathcal{A}$ est fréquemment validée au vu des capteurs (S_{sub} élevé) et respecte toutes les règles ($S_{\text{sym}} \approx 1$), alors le lien $\omega_{(i,\mathcal{A})}$ se **renforce**. Dans la mise à jour hebbienne :

$$\omega_{(i,\mathcal{A})}(t+1) \;=\; \omega_{(i,\mathcal{A})}(t) \;+\; \eta\left[S_{\text{hybrid}}(\mathcal{E}_i, \mathcal{A}) \;-\; \tau\, \omega_{(i,\mathcal{A})}(t)\right].$$

Si, en revanche, l'action apparaît incohérente (la force nécessaire dépasse un seuil, la zone est interdite, etc.), la pondération décroît, évitant que le robot ne réitère cette transition. Avec le temps, le **SCN** se stabilise en **clusters** ou en **chaînes** d'états-actions validées, traduisant un *plan* de manœuvre flexible qui reflète la fois la perception neuronale et les axiomes symboliques.

C. Exécution auto-organisée dans un DSL robotique

Dans un **SCN** robotique, chaque **état** ou **situation** $\mathcal{E}_i$ encapsule un **embedding** (issu des capteurs) et un **bloc** de règles (posture autorisée, zone permise, classe d'objet détecté, etc.). Les **actions** (prendre, poser, avancer, reculer) apparaissent comme des nœuds à part ou comme des transitions rattachées à un état. La force des liaisons $\omega_{(i,\mathcal{A})}$ se met à jour au fil des expériences, chaque essai ou simulation renforçant ou affaiblissant le couple (état, action) selon la synergie $\alpha\, S_{\text{sub}} + (1-\alpha)\, S_{\text{sym}}$.

- **Sub-symboliquement**, si le robot "voit" un objet de forme X (via l'encodage de vision) et ce même objet a été identifié comme Y dans la base, on obtient un haut degré de **similarité**.

- **Symboliquement**, si une règle R impose de ne pas saisir cet objet sans vérifier la force, alors la conformité de la force de préhension agit sur S_{sym}.

Ainsi, la commande “saisir l’objet” depuis un état $\mathcal{E}_i$ (caractérisé par un ensemble de mesures) ne sera validée que si le rapprochement sub-symbolique indique qu’il s’agit bien de l’objet visé (pas de confusion visuelle), et si la règle ne détecte pas d’incohérence (non-fragile, autorisé, zone libre, etc.). Le robot suit alors une **dynamique auto-organisée** — les **liaisons** ω se stabilisent sur des transitions conformes, et s’évanouissent pour celles jugées peu cohérentes.

Exemple

Un **robot picking** manipule divers objets sur un tapis roulant où il peut traiter des boîtes en carton fragile, des pièces métalliques lourdes, et d’autres types d’objets. Les règles imposent que si l’objet est fragile, la force de pincement doit demeurer < 10 N. Le sous-système sub-symbolique (caméra + force sensor) encode la forme et la masse apparente. Quand le robot tente l’action “pincer fort” pour un carton fragile, la synergie symbolique sera quasi nulle (contradiction avec la règle), même si sub-symboliquement la forme du carton correspond à un objet saisissable. Le poids $\omega_{(i,\text{pincerFort})}$ se réduit, tandis que “pincerDoucement” reçoit un renforcement positif. Progressivement, le **SCN** converge vers des gestes adaptés à la nature de chaque objet, tout en respectant la cohérence logicielle.

D. Bénéfices

Les **bénéfices** sont multiples.

La **robustesse** du système repose sur les capteurs sub-symboliques comme la vision, le LiDAR ou les capteurs de force, qui gèrent la **variabilité** de l’environnement. Si la luminosité change, le CNN peut toujours encoder un vecteur cohérent, et le LiDAR un nuage de points exploitable, permettant ainsi à l’**auto-organisation** du DSL de poursuivre son travail sans interruption.

Le **contrôle explicable** est garanti par l’intégration de **règles**, assurant une transparence de l’architecture. Il est possible d’expliquer pourquoi une action est refusée, en raison d’une contrainte de sécurité, ou validée, grâce à la confirmation des capteurs et à la satisfaction des règles établies.

L’**adaptation continue** est facilitée par la flexibilité du DSL. L’ajout d’un nouveau capteur, comme une caméra infrarouge ou un système RFID, s’intègre directement dans l’embedding sub-symbolique. De même, l’introduction d’une nouvelle règle de sécurité dans la base symbolique ne perturbe pas le fonctionnement global, le **DSL** assurant la mise à jour progressive de ω sans rupture du système.

La **scalabilité** du modèle permet une gestion efficace dans des environnements industriels complexes. Un robot peut traiter un grand nombre d’états tout en s’appuyant sur les mêmes principes de synergie. La matrice ω indique quels enchaînements **état → action → nouvel état** sont les plus performants, optimisant ainsi l’efficacité du système.

3.6.4. Comparaison Expérimentale

Tout au long de ce chapitre (3), nous avons présenté divers **types de représentation** (sub-symbolique, symbolique, hybride) et plusieurs **approches** avancées (hypergraphes, fractales). Une question cruciale demeure où il s'agit de déterminer **comment** comparer expérimentalement les performances d'un **DSL** (Deep Synergy Learning) en fonction du **choix** de la représentation. Dans cette section (3.6.4), nous abordons les éléments clés d'une **étude comparative** :

- (3.6.4.1) L'**impact** du type de représentation sur la **qualité** de la synergie,
- (3.6.4.2) Les **mesures** de clustering, le temps de convergence et la robustesse face à l'insertion de nouvelles entités,
- (3.6.4.3) L'**approche HPC** pour gérer un grand nombre de dimensions ou un grand volume de données.

3.6.4.1. **Impact du type de représentation (sub, sym, hybride) sur la qualité de la synergie**

Dans le cadre d'un **Deep Synergy Learning (DSL)**, le choix de la représentation des entités constitue un paramètre déterminant pour la qualité de la synergie entre celles-ci, c'est-à-dire pour l'efficacité de l'auto-organisation du réseau de connexion. Selon qu'on opte pour une approche **sub-symbolique** (basée sur des **embeddings neuronaux**), une approche **symbolique** (fondée sur des **règles**, des **axiomes** ou des **ontologies**), ou une approche **hybride** qui combine ces deux modalités, la structure des liaisons $\{\omega_{i,j}\}$ ainsi que la formation des clusters peuvent varier de manière significative. En effet, ces différences se manifestent principalement en termes de **robustesse** face au bruit, de **lisibilité** ou **explicabilité**, de **capacité de raisonnement** et de **scalabilité**.

A. Comparaison entre les représentations sub-symbolique, symbolique et hybride

Dans une approche **sub-symbolique**, chaque entité $\mathcal{E}_i$ est représentée par un **embedding** $\mathbf{x}_i \in \mathbb{R}^d$ issu d'un modèle d'apprentissage profond, lequel convertit des données brutes (images, extraits audio, phrases textuelles, etc.) en vecteurs qui condensent les caractéristiques les plus saillantes. La similarité entre deux entités est évaluée par des mesures classiques telles que la **similarité cosinus** ou la **distance euclidienne**. Par exemple, la similarité cosinus est donnée par

$$S_{\text{sub}}(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\| \mathbf{x}_i \| \| \mathbf{x}_j \|},$$

ce qui reflète le degré de rapprochement angulaire entre $\mathbf{x}_i$ et $\mathbf{x}_j$. Cette approche se caractérise par une **souplesse** et une **robustesse** intrinsèque, en particulier pour gérer un grand volume de données potentiellement bruitées ; toutefois, elle présente l'inconvénient d'être peu **explicable**, car il est difficile de déterminer explicitement quelles caractéristiques contribuent à la proximité entre deux embeddings, hormis par la valeur numérique obtenue.

En revanche, dans une approche **symbolique**, chaque entité $\mathcal{E}_i$ est décrite par un **bloc** de règles, d'axiomes ou d'attributs logiques, noté R_i. La similarité entre deux entités se mesure alors par un

score $S_{\text{sym}}(R_i, R_j)$ qui quantifie la **compatibilité** entre les ensembles de règles ou les structures logiques associées. Par exemple, une fonction de compatibilité peut être formulée comme

$$S_{\text{sym}}(R_i, R_j) = \begin{cases} 1, & \text{si les règles se valident mutuellement,} \\ 0, & \text{en cas de contradiction totale,} \\ s_{ij}, & \text{pour une compatibilité partielle,} \end{cases}$$

où s_{ij} est un score intermédiaire compris entre 0 et 1. Cette approche apporte une **explicabilité** élevée, puisque la logique sous-jacente offre la possibilité de retracer précisément pourquoi deux entités sont considérées comme compatibles, bien que le traitement symbolique soit souvent plus rigide et moins tolérant aux variations ou au bruit inhérent aux données.

Le **modèle hybride** vise à combiner le meilleur des deux approches précédentes. Dans ce cas, chaque entité est dotée à la fois d'un **embedding** $\mathbf{x}_i$ et d'un bloc de règles R_i. La synergie globale entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$ est alors définie par la formule

$$S_{\text{hybrid}}(i, j) = \alpha\, S_{\text{sub}}(\mathbf{x}_i, \mathbf{x}_j) + (1 - \alpha)\, S_{\text{sym}}(R_i, R_j),$$

où $\alpha \in [0,1]$ est un paramètre qui ajuste l'importance relative de la **dimension sub-symbolique** par rapport à la **dimension symbolique**. Ce schéma de fusion permet d'exploiter la **robustesse** des embeddings pour absorber le bruit et les variations lexicales, tout en bénéficiant du **raisonnement formel** et de la **traçabilité** qu'apporte la couche symbolique. Néanmoins, il nécessite un double stockage des informations (vecteur et règles) et des calculs supplémentaires, ce qui augmente la **complexité** de l'implémentation. La sélection judicieuse de α est cruciale pour obtenir le bon compromis entre **souplesse** et **précision**.

B. Métriques pour évaluer la qualité de la synergie

Pour évaluer la **qualité** des liaisons $\{\omega_{i,j}\}$ établies dans un DSL, il est important d'utiliser des **métriques** qui permettent de quantifier à la fois la **compacité** des clusters (mesurée par des indices tels que l'indice de Silhouette ou le Davies–Bouldin Index) et la **séparation** entre ces clusters. Lorsque l'approche sub-symbolique est employée, la justification d'un regroupement se base essentiellement sur la proximité numérique entre les embeddings. En revanche, une approche hybride offre la possibilité de lier la **proximité sémantique** à des critères explicites, c'est-à-dire de fournir une explication formelle à la raison pour laquelle deux entités se trouvent dans le même cluster. Cette capacité à fournir une **explication** riche renforce la lisibilité des résultats, en permettant par exemple d'associer à un regroupement une liste de règles communes ou un indice de cohérence formelle qui soutient la formation des clusters.

C. Observations empiriques et compromis

Dans la pratique, l'approche **purement sub-symbolique** se distingue par sa **robustesse** et sa capacité à absorber de grandes quantités de données bruitées, tout en permettant un calcul rapide des similarités à l'aide de fonctions vectorielles telles que le cosinus ou la distance euclidienne. Toutefois, cette méthode reste limitée en termes d'**explicabilité**, car elle ne fournit qu'un score numérique sans indication sur les motifs formels qui sous-tendent la similarité. En revanche, une approche **purement symbolique** offre une **cohérence** et une **transparence** importantes, puisque chaque regroupement peut être expliqué par des règles logiques précises, mais elle est souvent rigide et sensible aux incohérences, rendant sa mise en œuvre coûteuse en calcul lorsque la base de

règles est très large. Le modèle **hybride** se présente alors comme un compromis idéal, car il combine la **flexibilité** et la **robustesse** sub-symbolique avec la **rigueur** et la **clarté** symboliques. Cependant, cette approche hybride impose une augmentation du coût de stockage et de calcul, et requiert un réglage fin du paramètre α pour assurer le bon équilibre entre les deux composantes. En général, les expérimentations montrent qu'un DSL hybride parvient à générer des clusters d'entités plus **cohérents** et plus **explicables** que les approches purement sub-symboliques, tout en offrant une meilleure tolérance au bruit que les systèmes strictement symboliques.

3.6.4.2. **Mesures de clusters, temps de convergence, robustesse à l'insertion de nouvelles entités**

Dans le cadre d'un **Deep Synergy Learning (DSL)**, l'évaluation du comportement du système ne se limite pas à l'analyse statique de la structure auto-organisée, mais englobe également des aspects dynamiques essentiels, notamment la **qualité des clusters**, le **temps de convergence** du réseau ainsi que la **robustesse** face à l'insertion de nouvelles entités. Ces trois dimensions permettent de mesurer de façon approfondie la stabilité et la scalabilité du **Synergistic Connection Network (SCN)**. L'analyse de ces critères s'appuie sur des **indices mathématiques** et des **métriques** qui quantifient la compacité des regroupements, le temps nécessaire à la stabilisation des pondérations, ainsi que la capacité du système à intégrer de nouvelles informations sans perturber la structure existante.

*A. **Mesures de qualité des clusters***

Pour évaluer la qualité des clusters obtenus dans le SCN, on se réfère à des **indices de partition** qui mesurent la compacité intra-cluster et la séparation inter-clusters. Par exemple, l'**indice de Silhouette** est défini pour une entité $\mathcal{E}_i$ par

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i),\, b(i)\}},$$

où $a(i)$ représente la distance moyenne entre $\mathcal{E}_i$ et les autres entités du même cluster, et $b(i)$ la distance moyenne entre $\mathcal{E}_i$ et les entités du cluster le plus proche. Un score global s proche de 1 indique que les clusters sont bien définis et distincts. De même, l'**indice de Davies–Bouldin** s'exprime par

$$\mathrm{DB} = \frac{1}{k}\sum_{i=1}^{k} \max_{j \neq i}\left\{\frac{\sigma_i + \sigma_j}{d(c_i, c_j)}\right\},$$

où k est le nombre de clusters, σ_i la dispersion moyenne des entités dans le cluster i, et $d(c_i, c_j)$ la distance entre les centres c_i et c_j de clusters différents. Un indice DB plus faible signifie des clusters compacts et bien séparés. Par ailleurs, l'**indice Calinski–Harabasz** se définit comme

$$\mathrm{CH} = \frac{\mathrm{Tr}(B_k)}{\mathrm{Tr}(W_k)} \times \frac{n - k}{k - 1},$$

où $\mathrm{Tr}(B_k)$ et $\mathrm{Tr}(W_k)$ représentent respectivement la trace de la matrice de dispersion inter-clusters et intra-clusters, n étant le nombre total d'entités. Ces indices, appliqués aux représentations

obtenues par le DSL – qu'elles soient sub-symboliques, symboliques ou hybrides – permettent d'évaluer de manière quantitative la **qualité des regroupements**. Dans un système hybride, par exemple, la combinaison d'un score basé sur la **proximité vectorielle** et d'un score issu des **règles logiques** offre une mesure plus riche et explicable, où l'on peut retracer l'origine d'un regroupement à la fois par la similarité des embeddings et par la cohérence des règles associées.

B. Temps de convergence

Le **temps de convergence** du SCN est une mesure dynamique qui reflète la rapidité avec laquelle les pondérations $\omega_{i,j}(t)$ se stabilisent. La mise à jour des poids dans le DSL s'exprime par l'équation

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) + \eta\big[S(i,j) - \tau\,\omega_{i,j}(t)\big],$$

où $S(i,j)$ représente la synergie entre les entités, η le taux d'apprentissage et τ le coefficient de décroissance. La convergence se produit lorsque $\omega_{i,j}(t+1) \approx \omega_{i,j}(t)$ pour toutes les paires (i,j). On peut définir un critère de convergence, par exemple, lorsque

$$\max_{i,j} |\omega_{i,j}(t+1) - \omega_{i,j}(t)| < \varepsilon,$$

où ε est un seuil de tolérance fixé. Dans les systèmes **sub-symboliques**, le calcul des similarités vectorielles est généralement rapide, ce qui permet une convergence relativement rapide, à condition que la dimension d ne soit pas excessive. En revanche, dans les systèmes **symboliques** et **hybrides**, les opérations logiques et la fusion des scores augmentent la complexité, rallongeant ainsi le temps de convergence. L'évaluation empirique du temps de convergence permet de mesurer la **vitesse d'auto-organisation** et d'ajuster les hyperparamètres η et τ de façon à optimiser la stabilisation des clusters.

C. Robustesse à l'insertion de nouvelles entités

Un critère fondamental pour l'évolutivité d'un DSL est la capacité du SCN à intégrer de nouvelles entités sans perturber la structure auto-organisée préexistante. Si une nouvelle entité $\mathcal{E}_{\text{new}}$ dotée d'un embedding $\mathbf{x}_{\text{new}}$ (et, dans le cas d'un modèle hybride, d'un bloc de règles R_{new}) est insérée, il est souhaitable que les nouveaux liens $\omega_{\text{new},i}$ soient établis de manière **locale** et que la structure globale ne soit pas remise en cause. En pratique, le DSL compare $\mathbf{x}_{\text{new}}$ aux embeddings existants et, éventuellement, vérifie la compatibilité des règles associées pour calculer la synergie

$$S(\mathcal{E}_{\text{new}}, \mathcal{E}_i),$$

afin d'initier les pondérations $\omega_{\text{new},i}$. Cette approche permet de garantir une **insertion locale** sans nécessiter une re-synchronisation globale du SCN. La robustesse du système peut être évaluée par des métriques telles que le **changement moyen** des pondérations suite à l'ajout de nouvelles entités ou par des tests de stabilité de la structure de clusters, par exemple en mesurant l'évolution de l'indice de Silhouette ou du Davies–Bouldin Index avant et après l'insertion. Un DSL robuste sera capable de s'adapter de manière incrémentale, de sorte que l'ajout d'une entité provoque des ajustements locaux limités plutôt qu'une réorganisation massive du réseau.

3.6.4.3. Approche HPC si le DSL manipule un grand nombre de dimensions

Lorsqu'un **Deep Synergy Learning (DSL)** se déploie à large échelle, que ce soit en raison de la **dimension** élevée des vecteurs de représentation (embeddings sub-symboliques de plusieurs centaines ou milliers de composantes) ou du **nombre** massif d'entités manipulées, la charge de calcul pour maintenir la **synergie** $\{S(i,j)\}$ et la mise à jour itérative des $\{\omega_{i,j}\}$ peut rapidement croître au point de devenir prohibitive. Dans de telles configurations, une **approche HPC** (*High Performance Computing*) devient indispensable pour gérer de manière parallèle ou distribuée l'important volume d'opérations. L'idée consiste à recourir à des ressources matérielles avancées (GPU, clusters multiprocesseurs) et à des algorithmes spécifiques permettant de limiter l'impact de la croissance en $O(n^2 d)$, où n est le nombre total d'entités et d la taille des embeddings.

A. Échelle et complexité dans un DSL de grande dimension

Si l'on considère un **DSL** manipulant $\mathbf{x}_i \in \mathbb{R}^d$ pour chaque entité $\mathcal{E}_i$, et qu'on souhaite évaluer une synergie sub-symbolique via un **produit scalaire** ou une **distance** de la forme

$$\| \mathbf{x}_i - \mathbf{x}_j \|^2 = \sum_{k=1}^{d} (x_{i,k} - x_{j,k})^2,$$

le coût se chiffre rapidement en $O(n^2 d)$ si on compare toutes les paires (i,j). Pour un simple examen exhaustif de quelques centaines de milliers d'entités (ou davantage) en embeddings de dimension 768, le nombre d'opérations requises s'avère colossal. La **composante symbolique**, lorsqu'elle est présente, peut ajouter une vérification de règles dont la **complexité** peut être encore plus élevée, parfois de nature exponentielle selon la logique mise en œuvre. Il devient alors essentiel d'exploiter des **moyens de calcul parallèle** et de **techniques d'approximation** pour préserver la faisabilité du DSL.

La mise à jour répétée des pondérations, exprimée par exemple sous la forme

$$\omega_{i,j}(t+1) = \omega_{i,j}(t) \; + \; \eta \left[S(i,j) - \tau \, \omega_{i,j}(t) \right],$$

exige de réévaluer partiellement ou globalement la **synergie** $\{S(i,j)\}$ à chaque itération, d'où la nécessité d'un **High Performance Computing**. Sans ce levier, il est inenvisageable de mettre à jour une matrice de taille $O(n^2)$ dès que n atteint plusieurs centaines de milliers d'entités, surtout si la logique impose un raisonnement ponctuel pour vérifier la compatibilité de certaines règles $\mathbf{R}_i, \mathbf{R}_j$.

B. Parallélisation sub-symbolique et distribution symbolique

La partie **sub-symbolique** (calcul de similarité entre embeddings) peut être largement parallélisée sur GPU ou CPU multicœurs. Le produit scalaire

$$\mathbf{x}_i \cdot \mathbf{x}_j = \sum_{k=1}^{d} \left(x_{i,k} \, x_{j,k} \right)$$

peut se répartir par blocs matriciels ou par batchs, chaque nœud HPC traitant un sous-groupe de vecteurs. Dans un DSL géant, il est fréquent d'employer des algorithmes d'**approximate nearest neighbors** (Faiss, Annoy, HNSW) pour éviter un balayage exhaustif en $O(n^2)$. On réduit ainsi la construction du **SCN** aux seuls couples (i, j) dont la distance vectorielle est jugée potentiellement faible, ce qui autorise la parcimonie.

La partie **symbolique**, impliquant des règles ou des axiomes logiques $\mathbf{R}_i$, bénéficie parfois d'un déploiement sur des *reasoners* distribués ou des bases de connaissances échelonnées en grappes de serveurs. La compatibilité logique $\mathbf{R}_i, \mathbf{R}_j$ s'examine alors de manière partiellement parallèle, avec synchronisation des conclusions (contradiction, recouvrement) uniquement lorsqu'un sous-groupe d'entités sub-symboliquement proches exige une vérification fine. Cette stratégie "two-layers" consiste à filtrer sur la proximité neuronale et ne procéder au test symbolique complet que pour quelques paires pertinentes.

C. Organisation du SCN à grande échelle

Le **SCN** (Synergistic Connection Network) se construit de manière incrémentale et partiellement distribuée. Pour un lot de données, un ensemble de machines HPC calcule les voisinages k-NN dans l'espace $\mathbb{R}^d$. Chaque entité se voit attribuer une liste de candidats $\mathcal{N}_i \subset \{1, \dots, n\}$ correspondant à ses proches en embedding. Pour chaque couple $(i, j) \in \mathcal{N}_i$, on évalue la **composante symbolique** éventuelle $\mathbf{R}_i, \mathbf{R}_j$. On obtient ainsi

$$S_{\text{hybrid}}(i, j) = \alpha\, S_{\text{sub}}\big(\mathbf{x}_i, \mathbf{x}_j\big) + (1 - \alpha)\, S_{\text{sym}}\big(R_i, R_j\big),$$

que l'on exploite pour initialiser ou ajuster la pondération $\omega_{i,j}$. La matrice $\{\omega_{i,j}\}$ reste parcimonieuse, car on n'active que les liens dépassant un certain seuil, ou répondant à une condition d'intérêt sub-symbolique ou symbolique. À l'issue de chaque vague d'itérations, un protocole de synchronisation (barrière, agrégation) peut consolider les résultats.

Dans un tel schéma **HPC**, la **synchronisation** demeure un facteur critique où, si les nœuds traitent des partitions disjointes, la cohérence globale du **DSL** requiert un échange périodique des informations sur les couples inter-partitions. Les **paramètres** η (taux d'apprentissage), τ (décroissance) et α (poids sub-symbolique vs. symbolique) nécessitent des ajustements pour que l'on évite les oscillations dues à la latence ou à l'asynchronie.

Cette organisation rend cependant possible le traitement d'entités en nombre très élevé (jusqu'à des dizaines ou centaines de millions si l'on dispose d'assez de ressources), car on exploite des **librairies** HPC spécialisées (MPI, CUDA, OpenMP) et des *frameworks* (Spark, Hadoop) adaptés aux graphes distribués. Les opérations en $O(n^2 d)$ sont ainsi contournées par des mécanismes de filtrage (approximate k-NN) et de parallélisation.

D. Gains, contraintes et perspectives

L'approche HPC ouvre la voie à un **DSL** capable de gérer simultanément de larges volumes de données sub-symboliques et un grand nombre de règles. On préserve la **richesse** de l'auto-organisation (renforcement local $\Delta\omega_{i,j}$) sans voir les temps d'exécution exploser, grâce à une exploitation massive du calcul parallèle. Le principal **gain** est la capacité à mettre à jour la synergie $\{\omega_{i,j}\}$ pour plusieurs millions d'entités en un temps acceptable, tout en intégrant les compatibilités logiques.

Les **limites** résident dans la **complexité de déploiement** où il est nécessaire de configurer des clusters GPU ou multicœurs, de mettre en place des reasoners distribués et de synchroniser les partitions du **SCN**. Les **coûts** (financiers et en ingénierie) peuvent être considérables, réservant cette mise en œuvre à des instituts de recherche ou à des organisations gérant déjà des infrastructures HPC. De plus, l'approche reste conditionnée à des **heuristiques** de réduction (approximate k-NN, restrictions sur la portée des règles) pour rendre la dynamique $\omega_{i,j}(t)$ réellement scalable.

Néanmoins, ces méthodes **HPC** s'avèrent cruciales pour faire évoluer un **Deep Synergy Learning** vers des scénarios *big data*, où le DSL incorpore des embeddings dimensionnels imposants (issus de modèles neuronaux à grande échelle) et une **composante symbolique** non triviale. Il devient alors possible de concilier la **puissance** statistique (sub-symbolique) et la **cohérence** ou l'**explicabilité** (symbolique), malgré la croissance spectaculaire de la dimension d et du nombre n de nœuds, le tout orchestré par l'infrastructure HPC.

3.7. Conclusion

Au terme de ce **Chapitre 3**, nous avons examiné en profondeur la **représentation** des entités dans un cadre **DSL** (Deep Synergy Learning). Du choix d'une **approche sub-symbolique** (vecteurs, embeddings) à celui d'une **approche symbolique** (règles, ontologies), en passant par les **solutions hybrides** mêlant les deux, nous avons souligné à quel point la **qualité** de cette représentation influe directement sur la **synergie** calculée (et, partant, sur la structure du Synergistic Connection Network). Nous avons aussi abordé des **extensions avancées** (synergie n-aire, modèles fractals, multimodalité, etc.) et illustré ces principes à travers plusieurs **études de cas** (analyse audio-visuelle, agent conversationnel, robotique sensorielle).

3.7.1. Synthèse des Contributions du Chapitre

La question de la **représentation** d'une entité $\mathcal{E}_i$ dans un **Deep Synergy Learning (DSL)** a émergé comme un élément fondamental, dont dépend l'intégralité de la **fonction** de synergie $S(i,j)$ et, par voie de conséquence, la dynamique de mise à jour $\omega_{i,j}(t)$. Les analyses conduites dans ce chapitre ont démontré l'extrême diversité des **approches** possibles, depuis des **embeddings** neuronaux (sub-symboliques) jusqu'aux **règles** logiques (symboliques), en passant par des schémas **hybrides** plus complexes.

A. Diversité des approches de représentation

L'exploration a porté sur trois **familles** principales. D'abord, la **représentation sub-symbolique** s'appuie sur un **embedding** dans $\mathbb{R}^d$, souvent calculé via des réseaux neuronaux (par exemple, BERT, GPT, ViT). Cette forme de modélisation met l'accent sur la **similarité vectorielle**, qu'il s'agisse d'une distance euclidienne ou d'un produit scalaire normalisé (cosinus). Une telle approche assure une **tolérance** appréciable au bruit et aux variations sémantiques, ce qui s'avère précieux pour de grandes bases de données textuelles ou visuelles. En revanche, l'absence de structure explicite rend l'interprétation plus délicate et limite les possibilités de **raisonnement** formel.

Une seconde voie consiste en des représentations **symboliques**, articulées autour de **règles**, d'**axiomes** ou d'**ontologies**. Cette méthode repose sur une **cohérence déclarative** où la synergie $S_{\text{sym}}(i,j)$ naît de la compatibilité ou de l'absence de contradiction entre les blocs logiques attachés à $\mathcal{E}_i$ et $\mathcal{E}_j$. Pareille approche garantit une **transparence** conceptuelle, un **raisonnement** potentiellement exact, et la capacité de vérifier la non-contradiction de façon formelle. On lui reproche cependant une certaine rigidité face à un monde bruyant ou incertain, et une complexité algorithmique parfois élevée si la logique est riche (OWL, règles complexes).

Enfin, la troisième famille est **hybride** où elle associe le meilleur des deux précédentes en combinant la robustesse statistique des embeddings sub-symboliques et la force explicative ou normative de la logique. Le calcul de synergie devient alors un **mélange**, par exemple

$$S_{\text{hybrid}}(i,j) = \alpha\, S_{\text{sub}}\big(\mathbf{x}_i, \mathbf{x}_j\big) + (1-\alpha)\, S_{\text{sym}}\big(R_i, R_j\big),$$

de sorte que l'on bénéficie d'une tolérance au bruit couplée à une explicabilité formelle. Cette combinatoire exige néanmoins un paramétrage plus lourd et un certain surcoût algorithmique.

Cette **diversité** d'approches reflète la pluralité des **domaines** auxquels se confronte un DSL. Dans certains cas (images, flux sensoriels massifs), l'embedding sub-symbolique domine ; dans d'autres (règlementaire, médical), les règles symboliques fournissent le cadre essentiel ; les scénarios mixtes justifient l'usage d'une représentation hybride.

B. Importance de la qualité de la représentation

L'acte même de calculer la synergie $S(i,j)$ dépend étroitement de la **qualité** de la représentation employée pour $\mathcal{E}_i$ et $\mathcal{E}_j$. Si celle-ci est incomplète, trop approximative ou mal adaptée, la fonction de similarité ou de distance devient peu fiable, et les pondérations $\omega_{i,j}$ s'en ressentent où la **stabilité** de la mise à jour, la cohésion des clusters et l'émergence d'une structure interprétable dans le **SCN** se voient compromises. À l'inverse, une **représentation** minutieusement pensée – qu'il s'agisse d'un embedding « calibré » ou d'une base de règles sans redondances inutiles – renforce la **pertinence** des liens et la **lisibilité** du réseau.

Le chapitre a insisté sur le fait que le choix initial (sub-symbolique pur, logique pure, ou mixte) influence la manière dont évolue la distribution des $\omega_{i,j}$. Une représentation essentiellement vectorielle engendre une organisation liée aux distances dans $\mathbb{R}^d$ (clustering géométrique), tandis qu'une représentation purement symbolique façonne des **groupes** souvent déterminés par la logique d'inclusion, de contradiction ou de compatibilité d'axiomes. Les systèmes hybrides donnent des clusters plus sélectifs où l'on exige à la fois une proximité numérique et une cohérence de règles, ce qui affine la formation d'ensembles stables.

C. Aspects avancés : synergie n-aire, fractalité, embeddings transformers

Au-delà de la traditionnelle comparaison binaire, le chapitre a montré que la **synergie** peut se généraliser à des **structures n-aires** (hypergraphes) où l'on compare simultanément $\{\mathcal{E}_{i_1}, \dots, \mathcal{E}_{i_k}\}$. Cette extension s'avère primordiale dès que l'on veut saisir la configuration conjointe de plusieurs modalités (ex. audio–vidéo–texte). Elle impose une représentation capable de décrire des entités multiples au sein d'un même **bloc** de calcul.

La **fractalité**, ou **auto-similarité**, constitue une autre dimension abordée où certaines représentations fractales, comme les embeddings multirésolution ou les ontologies fractales, offrent une **compression** et une **récurrence** conceptuelle à plusieurs niveaux. Les bénéfices se révèlent dans des scénarios multi-échelle (Chap. 6) où l'on exploite la réitération systématique d'un même schéma à différentes granularités.

La question des **embeddings avancés** issus de **Transformers** (BERT, GPT, ViT) a souligné l'opportunité d'accéder à des **vecteurs** plus riches et plus contextuels, tout en augmentant la complexité de calcul et le besoin potentiel d'**approches HPC** pour gérer d'énormes ensembles de données.

Synthèse

L'axe majeur de ce **Chapitre 3** tient donc à la mise en évidence du rôle central de la **représentation** dans un **DSL** où toutes les formules $\omega_{i,j}(t+1)$ et tous les patterns $\{\omega_{i,j}\}$ observés à la fin de la convergence découlent in fine de **comment** chaque entité $\mathcal{E}_i$ est encodée. L'**embedding** sub-symbolique, la base de **règles** logiques ou l'approche **hybride** imposent un langage qui influence :

– La robustesse au bruit (approximation, flexibilité neuronale),

– La cohérence explicable (raisonnement symbolique, absence de contradiction),

– La complexité (c'est-à-dire le coût temporel et matériel de calcul),

– La forme même des "clusters" ou "hyper-clusters" émergents.

Les **prochaines** parties (Chapitres 4, 5, 6) s'appuieront précisément sur cette **base** pour décrire la manière dont un **SCN** organise ses entités, étudie la dynamique de mise à jour (convergence, attracteurs) et peut s'étendre à des architectures distribuées ou multi-échelle, confirmant que les **choix** opérés dans le présent chapitre quant à la **représentation** constituent l'ossature de toute la suite du **Deep Synergy Learning**.

3.7.2. Limites et Pistes Futures

Malgré la diversité d'approches abordées au **Chapitre 3** – qu'il s'agisse de représentations **sub-symboliques** par embeddings neuronaux, de descriptions **symboliques** formées de règles ou d'axiomes, ou encore de modèles **hybrides** fusionnant les deux dimensions –, il demeure manifeste qu'aucune formulation n'est universellement optimale. Chaque méthode se plie à des **besoins** précis, des **données** particulières et des **ressources** de calcul données, si bien que le choix d'une représentation dépend étroitement du contexte. Il subsiste aussi de nombreux axes de **recherche**, qu'il s'agisse d'unification, de standardisation ou de perfectionnement des mécanismes de synergie au sein d'un **DSL** (*Deep Synergy Learning*).

A. Pas de représentation "unique" optimale

La première observation tient au fait qu'il n'existe pas de **format** apte à couvrir tous les cas d'usage avec la même efficacité. Les **embeddings** sub-symboliques (calculés dans $\mathbb{R}^d$) s'imposent lorsque l'on traite un grand volume de données bruitées, telles que des images, du son ou des segments textuels en langage naturel. Ils offrent une **robustesse** notable aux variations aléatoires et aux erreurs mineures, mais n'apportent pas la même explicabilité formelle que des règles logiques. À l'inverse, une approche strictement **symbolique** excelle pour des tâches exigeant un **raisonnement** strict, un contrôle de la **cohérence** ou une traçabilité explicite des décisions (domaine médical, légal, etc.). Cependant, elle se révèle rigide dès lors qu'apparaissent des informations ambiguës ou bruitées que la logique pure gère difficilement. L'option **hybride** tente de réconcilier ces deux mondes en calculant la **synergie** de façon mixte, ce qui peut apporter une meilleure **interprétabilité** tout en conservant une certaine tolérance aux imprécisions, au prix d'une infrastructure plus complexe et d'un temps de calcul plus important.

Un second aspect a trait aux **ressources** de calcul où des représentations sub-symboliques de grande dimension ou des bases de règles logiques massives peuvent exiger des **moyens HPC** (High Performance Computing) pour atteindre la performance requise. Dans certaines situations où l'on dispose de peu de capacité machine, il est parfois plus judicieux de réduire la complexité des modèles, soit en limitant la taille des embeddings, soit en simplifiant l'ontologie ou les règles symboliques, afin de maintenir le **DSL** dans des délais acceptables. Enfin, la **capacité d'évolution** du **DSL** joue un rôle où, lors d'un apprentissage continu, de nouvelles entités et de nouveaux attributs apparaissent. Une représentation sub-symbolique est souvent plus facile à "raffiner" (par

un fine-tuning local) qu'une base de règles, dont la moindre modification peut exiger de recomposer la cohérence globale.

B. Recherche : standardisation des formats, outils pour la synergie multi-modal ou multi-domaine

Les différentes méthodes rencontrées dans le **Chapitre 3** soulignent la nécessité de **standards** où chaque domaine, qu'il s'agisse de vision, de son, d'ontologies, de textes ou de robotique, recourt à un système distinct, comme **RDF, OWL** pour la partie symbolique et **ONNX ou TF SavedModel** pour la partie neuronale. L'élaboration d'un format **commun** – permettant de décrire à la fois un bloc symbolique $\mathbf{R}_i$ et un embedding $\mathbf{x}_i$ de façon unifiée – pourrait fluidifier l'intégration de modules hétérogènes dans un **DSL**. Cette standardisation faciliterait également la réutilisation de bibliothèques ou l'échange de modèles entre différents domaines.

Les expériences en **multimodalité** (où l'on fusionne plusieurs types de signaux) et en **multidomaine** (où un même **SCN** chapeaute des entités relevant de divers secteurs) suggèrent de développer des **kernels** ou des **fonctions de synergie** plus avancées, aptes à combiner plusieurs embeddings à la fois tout en tenant compte d'éventuelles règles logiques. Il devient alors crucial de définir :

$$S_{\text{hybrid}}(\mathcal{E}_i, \mathcal{E}_j) = \alpha_1 \, S_{\text{sub},1}(\mathbf{x}_{i,1}, \mathbf{x}_{j,1}) + \cdots + \alpha_m \, S_{\text{sub,m}}(\mathbf{x}_{i,m}, \mathbf{x}_{j,m}) + (1 - \alpha_\Sigma) \, S_{\text{sym}}(R_i, R_j),$$

où $\{\mathbf{x}_{i,k}\}$ représentent différents **embeddings** issus de modalités ou de domaines divergents, et $\{R_i\}$ décrivent leurs attributs symboliques. La détermination optimale des $\{\alpha_k\}$ constitue un enjeu de calibration, pour équilibrer la part d'influence de chacune des modalités face à la couche logique. Il s'agit d'une **piste de recherche** encore largement ouverte.

C. Lien possible avec la sécurité (Chap. 11) et la convergence (Chap. 4, 7)

Les représentations discutées ici interfèrent avec d'autres volets du **Deep Synergy Learning**. Du point de vue de la **sécurité** (Chap. 11), la robustesse d'un modèle sub-symbolique s'avère cruciale pour repérer des signaux anormaux (par exemple dans des embeddings modifiés de manière adversaire), tandis qu'une base de règles symboliques permet de justifier pourquoi l'on suspecte un comportement malveillant, ou de repérer un usage qui contrevient à certaines politiques. Un DSL "hybride" renforce donc cette défense en alliant détection statistique et validation logique.

Sur le plan de la **convergence** (Chapitres 4 et 7), la présence d'une logique symbolique, parfois discontinue (une légère modification peut changer radicalement la compatibilité de deux blocs de règles), peut introduire des points de non-lissité dans la fonction de synergie globale, ce qui soulève des problèmes de stabilité et de temps de convergence. Les algorithmes d'optimisation et les garanties de stabilisation du SCN peuvent devenir plus complexes que dans un cadre sub-symbolique lisse (où la distance euclidienne donne une fonction continue). On entrevoit ici une **piste d'investigation** consistant à examiner la dynamique $\{\omega_{i,j}(t)\}$ lorsque la composante symbolique introduit des mises à jour discrètes et non linéaires.

3.7.3. Liens avec les Chapitres Suivants

Les développements du **Chapitre 3** ont mis en lumière la **diversité** des **représentations** – sub-symboliques, symboliques, hybrides, voire plus avancées (n-aires, fractales) – et la **manière** dont

elles peuvent se combiner dans un **DSL** (*Deep Synergy Learning*). Il reste cependant à comprendre **comment**, à partir de ces entités ainsi représentées, se déploie toute la **dynamique** d'auto-organisation et **comment** cette architecture s'inscrit dans un cadre global multi-échelle. Les chapitres ultérieurs (4, 5, 6) poursuivront précisément dans cette direction, en s'appuyant sur les notions introduites ici.

A. Chapitre 4 : Dynamique d'Auto-Organisation

Le **Chapitre 4** approfondira la **dynamique** propre au **SCN** (Synergistic Connection Network), c'est-à-dire le **mécanisme** régissant l'évolution des pondérations $\omega_{i,j}$ au fil des itérations. Cette dimension dynamique n'a été que brièvement abordée jusqu'ici, puisque le présent chapitre (3) se focalisait surtout sur la **nature** des entités $\{\mathcal{E}_i\}$ et la **forme** de leur représentation $\mathbf{r}(i)$. Les différentes **représentations** suggèrent en effet des **fonctions de synergie** distinctes, et donc des **équations** de mise à jour différentes.

Lorsqu'un DSL recourt à des embeddings sub-symboliques, la fonction $S_{\text{sub}}(i,j)$ repose sur une distance continue (euclidienne, cosinus), engendrant une mise à jour "lisse" qui tend à éviter les discontinuités. À l'inverse, une représentation symbolique peut amener des discontinuités logiques (contradiction vs. non-contradiction), rendant la dynamique plus délicate à analyser. Le **Chapitre 4** explicitera comment chaque type de représentation influe sur la **vitesse** et la **stabilité** de la convergence ($\Delta\omega_{i,j} \approx 0$), et examinera la notion d'**attracteurs** ou de **clusters** stables qui se forment lorsque le SCN atteint l'équilibre.

B. Chapitre 5 : Architecture SCN et Exploitation Pratique

Dans le **Chapitre 5**, on s'intéressera à la **structure globale** du **Synergistic Connection Network**, c'est-à-dire la manière dont on assemble concrètement les entités dans un graphe adaptatif. Les entités que l'on a décrites ici (qu'elles soient sub-symboliques, symboliques ou hybrides) y prennent la forme de *nœuds* ou de modules, reliés par des pondérations $\omega_{i,j}$. Ce chapitre décrira plus avant les **algorithmes** d'**exploitation** du SCN, par exemple pour :

– Partitionner le réseau en sous-groupes (macro-clusters),

– Mettre en place un système distribué ou modulaire,

– Gérer la dynamique en parallèle (HPC, chap. 3.6.4.3).

On clarifiera aussi les **formes** que peut prendre le **SCN** où il peut s'agir d'un simple graphe non orienté, d'un hypergraphe ou d'une architecture hiérarchique où les liens s'organisent en couches. Les choix de **représentation** exposés au Chap. 3 déterminent en partie la **manière** dont on code les attributs ou règles associées aux nœuds, et influencent la **conception** même de l'architecture SCN.

C. Chapitre 6 : Multi-Échelle, Fractalité et Hiérarchies

Le **Chapitre 6** approfondira la **dimension multi-échelle** déjà évoquée. Les représentations fractales (section 3.6.2.2) et les constructions n-aires se prêteront à une **organisation** hiérarchique ou **fractal-like**. Il sera montré que l'on peut concevoir le DSL comme un système où plusieurs **niveaux** (micro, macro) interagissent, chacun recourant aux mêmes principes de **synergie**, mais adaptés à l'échelle correspondante. On y détaillera également comment un SCN fractal, ou multi-

échelle, peut renforcer l'**auto-organisation**, et comment la formation simultanée de **clusters** locaux et globaux démultiplie la puissance de l'approche.

Le lien avec les représentations décrites au **Chapitre 3** est direct où un embedding fractal, par exemple, prend tout son sens dans un réseau multi-niveaux, et une ontologie fractale trouve également sa place dans la construction d'un **SCN** hiérarchique. Le **Chapitre 6** montrera comment ces idées se combinent pour donner au DSL une **cohérence** à la fois locale et globale, fortement inspirée de la théorie des systèmes complexes et de la cognition distribuée.

3.7.4. Vision Globale

Lorsqu'on aborde la construction d'un **Deep Synergy Learning (DSL)** et de son **Synergistic Connection Network (SCN)**, la notion de **représentation** occupe une place centrale. Cette représentation, qu'elle soit sub-symbolique, symbolique ou hybride, agit comme la **pierre angulaire** de toute la dynamique subséquente. Les analyses précédentes ont montré que la fonction $S(i,j)$, calculée entre deux entités $\mathcal{E}_i$ et $\mathcal{E}_j$, dépend directement de la manière dont ces entités sont **modélisées**. Il s'avère ainsi que la richesse, l'adaptation ou au contraire la pauvreté d'un schéma de représentation se répercutent sur la qualité des *clusters*, la stabilité de la mise à jour $\omega_{i,j}(t)$, et la capacité du DSL à s'ajuster à un environnement complexe.

A. Impact sur la qualité de la dynamique et des clusters

Les expériences de ce chapitre confirment que si la représentation initiale est imprécise ou non adaptée, les résultats d'auto-organisation peuvent se révéler fragiles ou incohérents où les $\omega_{i,j}$ se stabilisent difficilement, et l'on observe des *clusters* dont la pertinence s'avère discutable. À l'inverse, une représentation bien pensée, qu'il s'agisse d'un **embedding** sub-symbolique pertinent ou d'un **ensemble** de règles logiques cohérentes, induit une fonction de synergie plus robuste. Cette robustesse se traduit par une formation de partitions ou de groupements lisibles, et par une dynamique de stabilisation plus rapide ou plus fiable.

Pour certains types de données très bruitées (images, enregistrements audio, textes informels), un embedding neuronal s'impose souvent comme un compromis efficace, offrant une métrique continue dans $\mathbb{R}^d$. Dans d'autres domaines plus réglementés (domaines médical, légal, etc.), la nature symbolique d'une représentation par **règles** ou **axiomes** facilite l'explicabilité et permet un raisonnement formel, tout en résistant mal au bruit et aux variantes imprévues. Les systèmes **hybrides** puisent leur force dans la combinaison de ces deux registres où ils assurent une flexibilité face à la variabilité sub-symbolique tout en maintenant un contrôle conceptuel par la logique, ce qui rejaillit directement sur la **cohérence** des clusters et la **lisibilité** de la carte du réseau $\{\omega_{i,j}\}$.

B. Choix de modélisation pour un DSL lisible et performant

Le panorama offert par les sections précédentes rappelle qu'**il n'existe pas de solution unique** optimisant simultanément la rapidité d'exécution, l'explicabilité, la gestion du bruit et la capacité de raisonnement. Pour des applications volumineuses, les embeddings sub-symboliques dominent souvent en pratique, mais au détriment de la traçabilité formelle. Pour des tâches exigeant un raisonnement exact, les règles symboliques priment, moyennant parfois une lourdeur de calcul ou

une rigidité de mise à jour. Les solutions **hybrides** visent un équilibre, souvent gagné au prix d'une architecture plus complexe et de ressources plus importantes.

Dans une architecture DSL couvrant plusieurs **modalités** (vision, audio, textes) et intégrant une **couche** de logique (règles métier, ontologies fractales), la diversité des **outils** (fractalité, synergie n-aire, HPC) aide à maintenir un certain degré de performance, même lorsque la dimension d des embeddings ou le nombre n d'entités croît fortement. Le système se révèle alors capable de faire face à des **scénarios** réalistes, où plusieurs types de données coexistent, où l'on exige à la fois la tolérance au bruit et l'aptitude à justifier des décisions.

C. Perspectives et intégration dans les chapitres suivants

Si l'on admet que la **représentation** est la fondation d'un **DSL**, il apparaît naturel que les chapitres ultérieurs (4, 5, 6) exploitent et prolongent ce travail. Le **Chapitre 4** décrira la *dynamique* de la mise à jour $\Delta\omega_{i,j}$, montrant comment la *nature* même de la fonction $S(i,j)$ influe sur la stabilité ou la vitesse de convergence. Le **Chapitre 5** s'attachera à la *structure* d'un SCN et à sa mise en œuvre pratique, en déclinant notamment les algorithmes de construction et d'exploitation. Enfin, le **Chapitre 6** abordera la *multi-échelle* et la *fractalité* dans l'organisation du DSL, qui constituent des cas d'application sophistiqués pour les représentations hybrides ou auto-similaires évoquées dans ce chapitre.