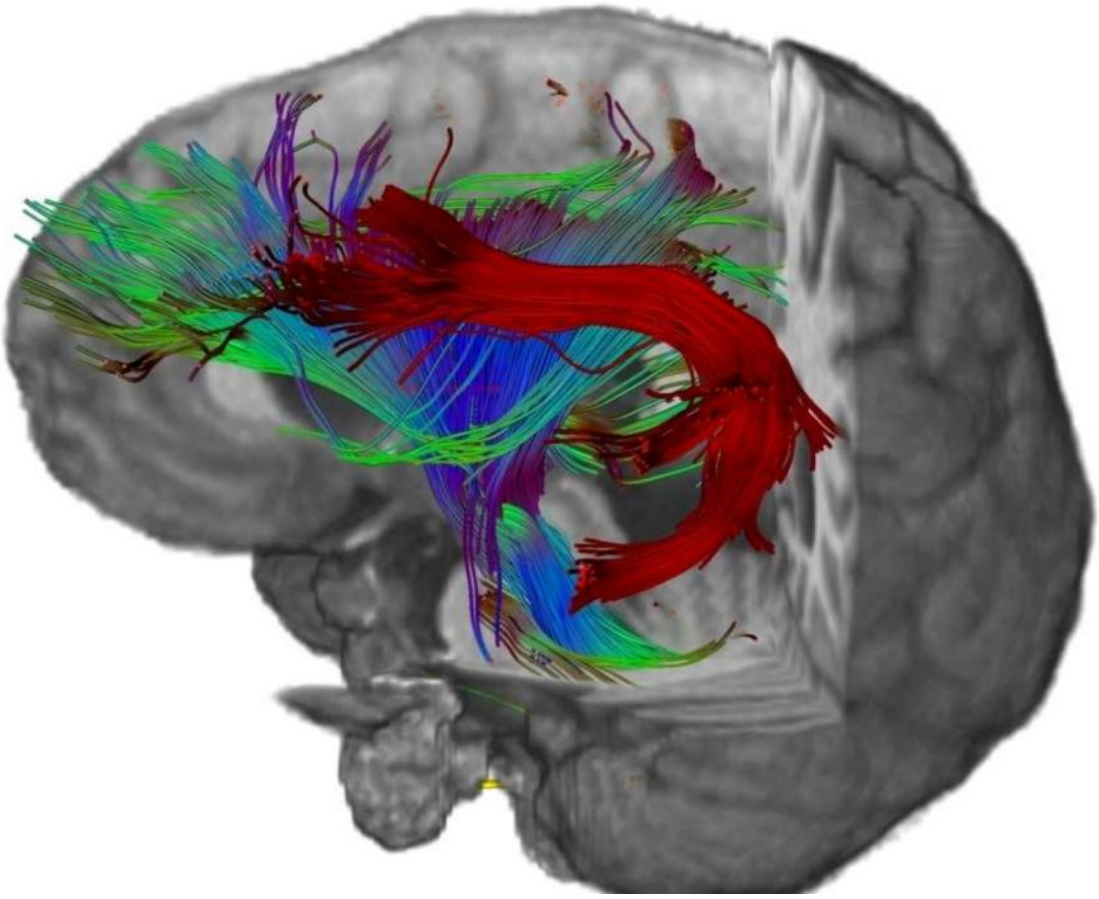


Credit:https://cerf.radiologie.fr/sites/cerf.radiologie.fr/files/files/enseignement/pdf/7 Cours Tracto 2016-2.pdf

**L’intelligence artificielle : Est-ce aussi «con» qu’une planche de Galton ?**

Auteur : Jérôme Vétillard , Langue : FR, Date : April 20, 2023

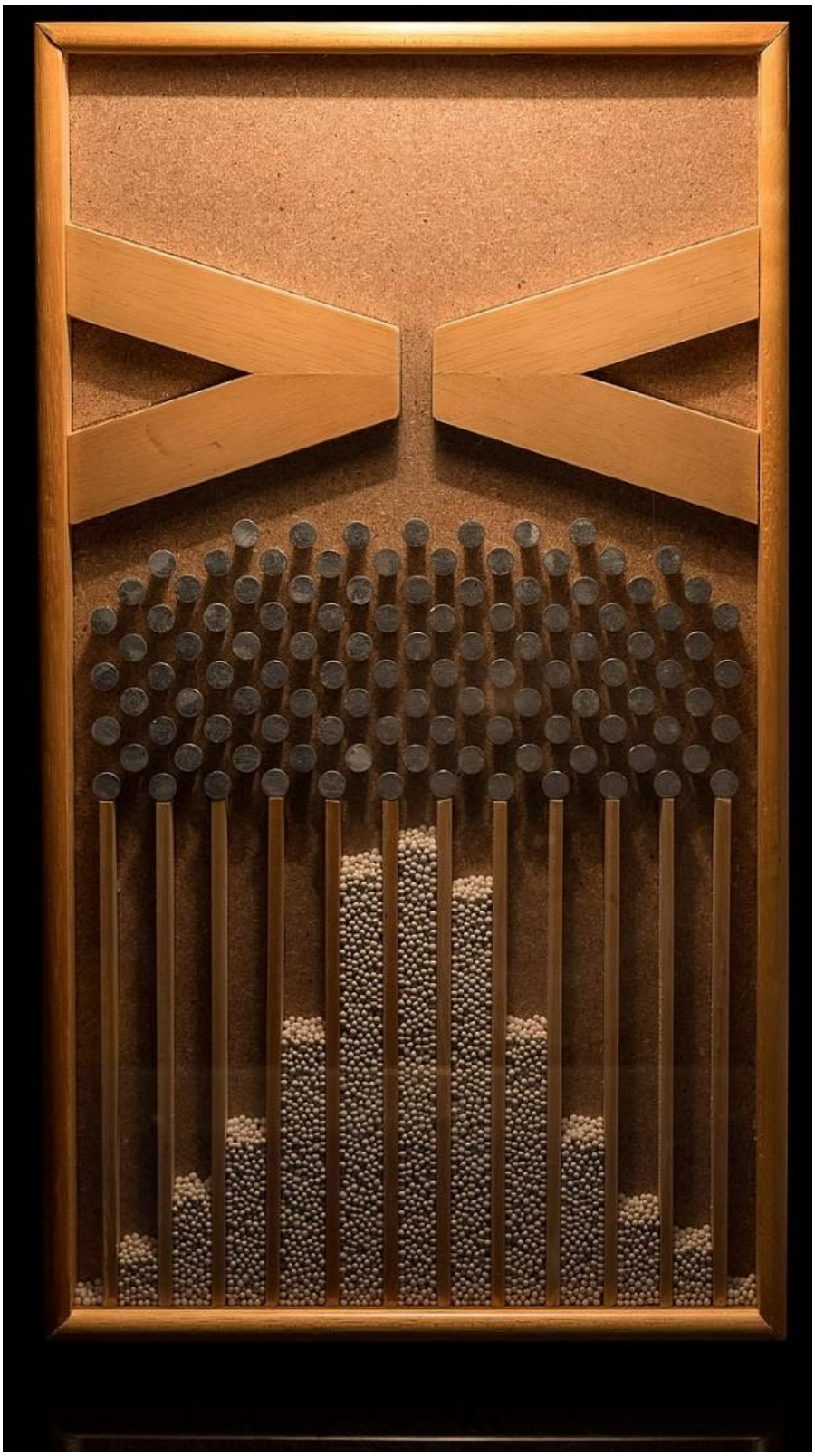
J’aurais pu écrire un haïku ou un alexandrin sur le sujet (parce que ça rime comme dans "Gaston y a le téléphone qui son". Par exemple (merci chatGPT) :

« Planche de Galton, Con et intelligence, Chaos s'épanouit. »

Bon ça manque de références aux saisons qui passent, de mélancolie et la métrique me semble approximative, mais là n’est pas le sujet. Nous allons voir, après les avoir décrits succinctement, les analogies pouvant exister entre le fonctionnement de la « planche de Galton », et un réseau de neurones.

A partir de ces analogies, nous nous questionnerons sur l’émergence de la complexité comme moteur intrinsèque de l’émergence de l’intelligence, voire de la conscience elle-même (peut être dans un autre article sinon cela risque de saturer votre mémoire attentionnelle 😊).

Une planche de Galton (voir l'article Wikepedia **Planche de Galton — Wikipédia (wikipedia.org)**) et les réseaux de neurones sont deux objets/concepts distincts, mais il est possible d'identifier quelques analogies intéressantes entre eux.



"Une planche de Galton". Source : https://fr.wikipedia.org/wiki/Planche_de_Galton#/media/Fichier:Galton_box.jpg

**La planche de Galton**

**Description du dispositif expérimental**

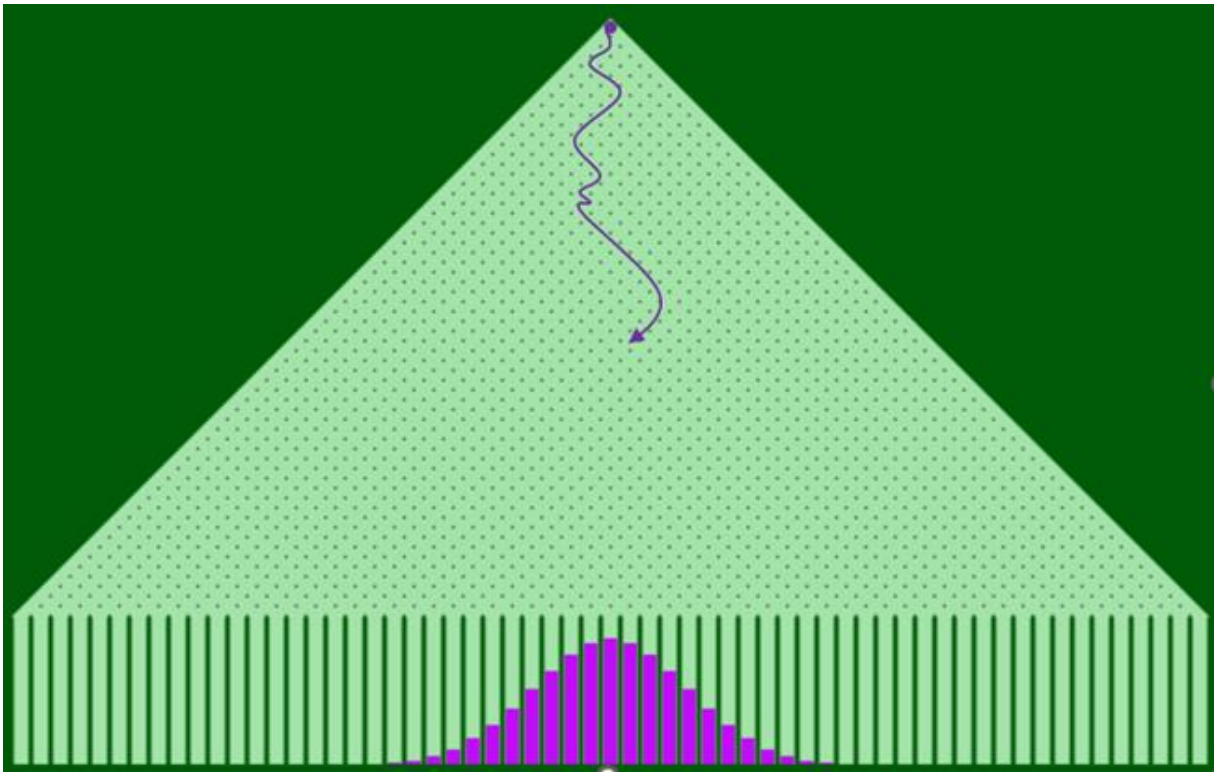
La planche de Galton est un dispositif mécanique qui illustre la distribution normale ou la loi binomiale à travers le mouvement de billes qui tombent sur des clous et se répartissent dans des bacs.

Les réseaux de neurones, en particulier ceux utilisant des fonctions d'activation probabilistes, peuvent également modéliser des distributions de probabilité.

Dans ce sens, les deux systèmes traitent des distributions et des probabilités en faisant appel massivement aux probabilités conditionnelles (Probabilité de l'évènement B sachant que l'évènement A est survenu).

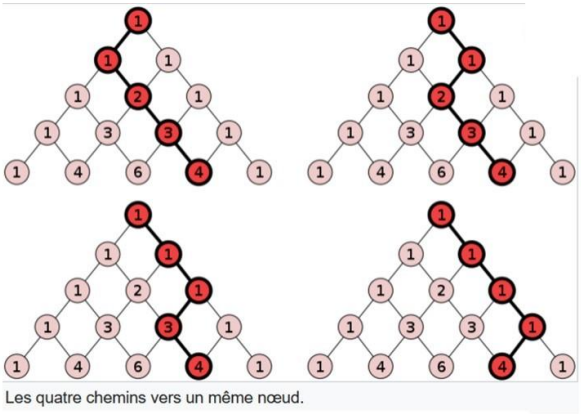
· Pour la planche de Galton, sa modélisation mathématique est décrite par l'arbre de probabilité de la loi binomiale. Elle modélise la réalisation successive (par chocs successifs entre les billes constituant le vecteur d'entrée et les clous/picots qui ségrégent le flux de billes entre deux directions avec une probabilité *p* pour l'une des directions, et *1-p* pour l'autre direction) d'une série d'épreuves de Bernoulli, c'est-à-dire *N* « tirages » indépendants d'une variable aléatoire pouvant prendre deux valeurs.

· Dans l'expérience originale, on lâche une bille dans la planche de Galton, et on observe les chocs avec les clous/picots et la gravité « décider » du réceptacle/colonne dans lequel finira par tomber la bille.



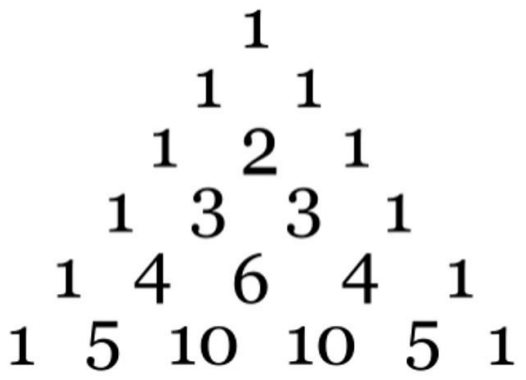
Source : https://sorciersdesalem.math.cnrs.fr/Vulgarisation/Galton/galton_plus.html

· Lorsqu'une bille est relâchée au sommet de la planche, la répartition des probabilités selon laquelle elle atterrira dans une colonne spécifique est un concept fondamental en théorie des probabilités discrètes, appelé loi binomiale. Étant donné que toutes les trajectoires possibles ont la même probabilité, la chance que la bille termine sa course dans une colonne particulière dépend du nombre de voies menant du sommet de la planche à cette colonne. En effet à chaque étage, on considère que la probabilité d'aller vers la gauche est de 0,5, et celle d'aller vers la droite est de 0,5. Quand on répète N fois ce « choc » avec le clou/picot, au fur et à mesure que la bille descend par l'effet de la gravité, la probabilité de la route au rang N est de 0,5^N et ceci est valable pour tous les chemins menant jusqu'au rang N. Toutefois, le nombre de routes menant à une position particulière est différent selon la position. Ce nombre de routes est représenté par un coefficient binomial, qui est déterminé à l'aide du (fameux) triangle de Pascal.



Les quatre chemins vers un même nœud.

Le triangle de Pascal permet de calculer le nombre de routes menant à chaque « case » ou point topologique. Il possède de nombreuses propriétés. Chaque rangée du triangle pascal commence par un *1*, et chaque nombre suivant est la somme des deux nombres directement au-dessus de lui. Il peut servir à calculer les coefficients du polynôme $(a+b)^N$ qui vont se lire sur la ligne (j), le sommet du triangle étant la ligne 0. Pour N=2, on peut lire 1/2/1, et donc $(a+b)^2=1*a^2 + 2*a^{2-1}*b^1 + b^2$ (on reconnaît ici l'identité remarquable $(a+b)^2=a^2 + 2ab + b^2$. Ici il sert à calculer le nombre de chemins menant au point topologique choisi. Ici il y a 4 chemins/routes possibles pour arriver à la position finale.




**Le triangle de Pascal**
Les coefficients dit « binomiaux » s'obtiennent par le calcul du nombre de combinaisons de *k* éléments choisis parmi *n.* Ici, *k* est la position dans la ligne, et *n* le numéro de la ligne (la première ligne étant réglée à 0). Ici on peut vérifier que 6 ($3^{ième}$ position, $4^{ième}$ ligne) est bien égal à C(3,4).

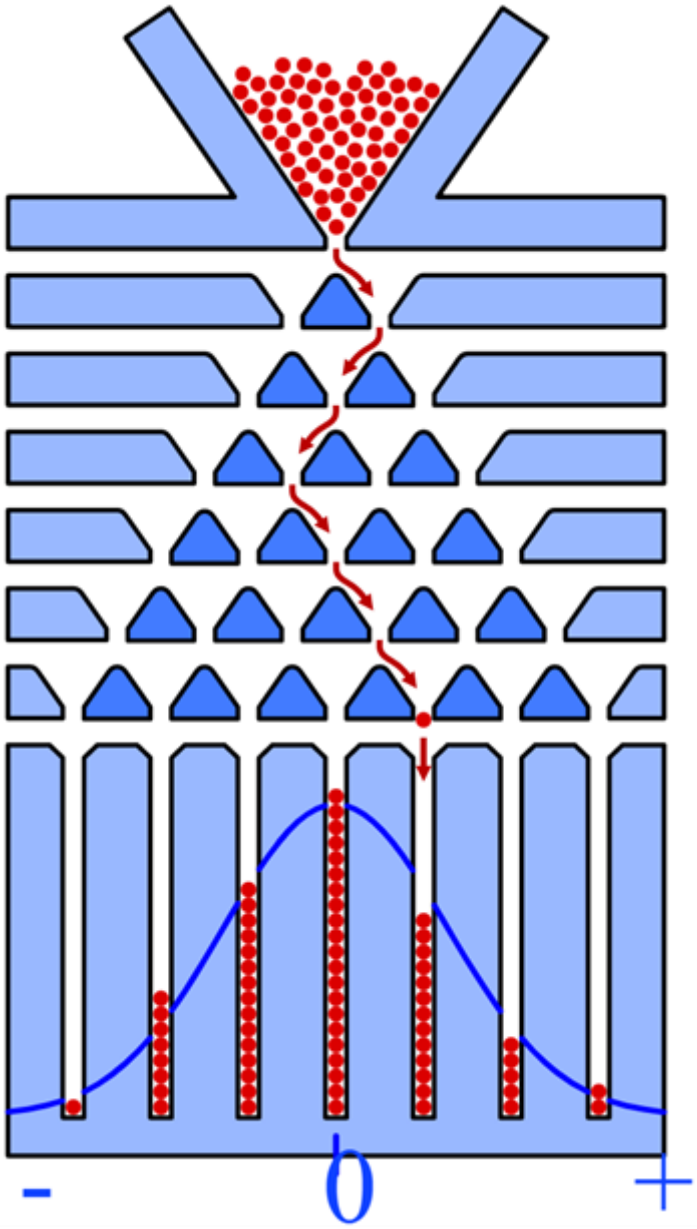
Sources : https://fr.wikipedia.org/wiki/Triangle_de_Pascal - https://commons.wikimedia.org/wiki/File:PascalSimetria.svg

**Théorie mathématique**

· Quand on répète l'expérience consistant à laisser tomber une bille unique dans la planche de Galton, on finit par observer une distribution particulière dans l'accumulation de billes dans les réceptacles/colonnes à la base de la planche.

« [...]les billes rouges (les points rouges sur la figure) empilées dans le bas de l'appareil correspondent à la **fonction de masse** de la loi binomiale, la courbe bleue correspond à la densité de la loi normale. [...] »

Source : [Loi binomiale — Wikipédia (wikipedia.org)]

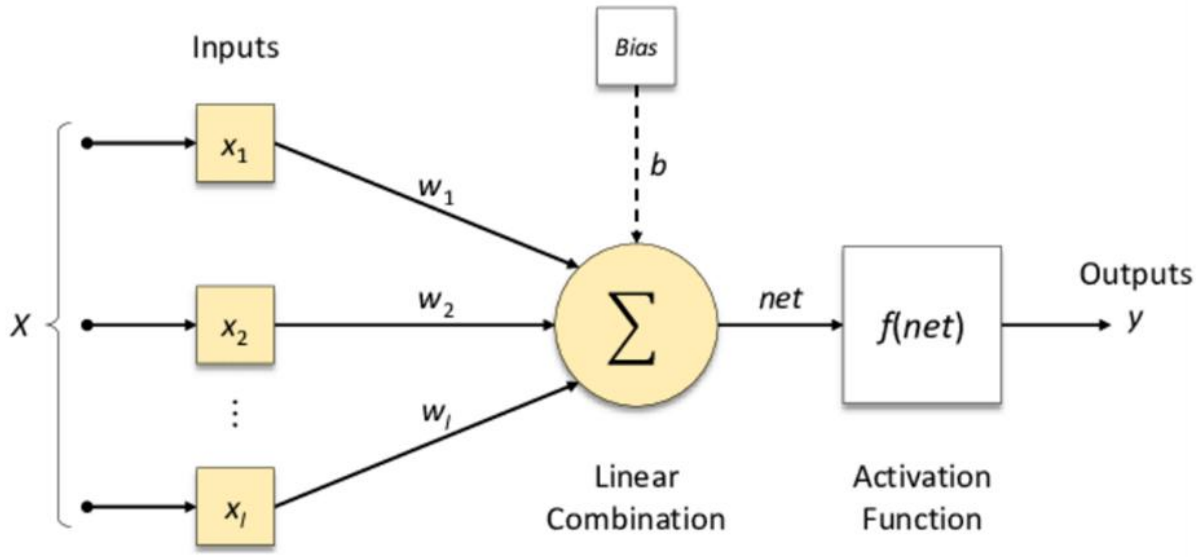



· La loi binomiale est une loi de probabilité discrète. Quand le nombre de tirages N tend vers plus l'infini, la fonction de masse de la loi binomiale tend à se « confondre » avec la densité de la loi normale (loi des grands nombres et théorème de Moivre-Laplace).

**L'architecture et fonctionnement (simplifiés) des réseaux de neurones**

**Le perceptron comme modèle de neurone numérique**

Un neurone numérique (modèle du perceptron) va recevoir un signal (numérique) amont, le traiter par une fonction mathématique interne, et selon le résultat de ce traitement, émettre un signal (numérique) aval.




Source
https://www.researchgate.net/publication/330742498_Evaluation_of_statistical_and_machine_learning_models_for_time_series_prediction_Identifying_the_state-of-the-art_and_the_best_conditions_for_the_use_of_each_model

Dans cette « architecture », nous avons trois « signaux » (pouvant être émis) par des perceptrons de la couche antérieure/supérieure X1, X2, X3 avec des pondérations w1, w2, w3 (poids synaptiques) vont être transmis au perceptron (neurone numérique) qui va réaliser la somme pondérée de ces signaux. Cette somme pondérée peut être ajutée d'un biais éventuel, et envoyé à la fonction d'activation.

Une fonction d'activation « basique » peut être :

· Si la somme pondérée ne dépasse pas un seuil S, alors ne pas transmettre de signal,

· Si la somme pondérée dépasse le seuil S, alors transmettre le signal vers la synapse suivante avec le poids wi,j (poids de la i-ième synapse de la couche j),

Les perceptrons sont organisés en couches successives :

· Couche d'entrée : constituée de N perceptrons qui va « ingérer » les valeurs de votre vecteur d'entrée, c'est-à-dire un *N-uplet*.

· (k) couches cachées pouvant avoir plus ou moins de perceptrons

· Couche de sortie : constituée de J perceptrons qui vont émettre le « signal » de sortie, c'est-à-dire un vecteur à J dimensions.

Les perceptrons entre les différentes couches sont reliés par des « synapses » qui portent une pondération (un scalaire qui va « moduler » la valeur du signal émis par le perceptron aval).

On peut avoir différent type de maillage. Par exemple si on imagine un « plat de lasagne » constitué de N-couches du même nombre K de perceptrons, chaque perceptron d'une couche, étant relié à tous les perceptrons de la couche inférieure (full mesh) :

· Entre deux couches successives de K perceptrons, on obtient alors $K^2$ synapses.

· Entre N couches successives de K perceptrons, on obtient alors K^N synapses, et donc autant de poids synaptiques, de « paramètres » du modèle.

C'est donc une architecture de traitement du signal, avec un signal d'entrée (votre « vecteur » d'entrée), multiplié par la(les) matrice(s) de pondérations des « poids synaptiques », qui sont également pour les Large Language Models (LLM) des matrices de poids probabilistes qui va donner le vecteur de sortie.

**Les fameux transformers**

Non ce n'est pas de la saga des robots de la planète Omicron dont il s'agit.

Dans le cas des LLM qui sont basés sur des transformers, il y a deux composants importants :

· L'encodeur :

Le rôle de l'encodeur est de traiter et de représenter l'information d'entrée sous une forme utile pour le décodeur. L'encodeur est composé de plusieurs couches identiques superposées, chacune ayant deux sous-modules : le mécanisme d'attention multi-têtes et le réseau de neurones feed-forward positionnel.

Mécanisme d'attention multi-têtes : Il permet au transformeur de capturer les dépendances contextuelles entre les mots à différentes positions dans la séquence d'entrée. Il se concentre sur différentes parties de la séquence pour déterminer les relations entre les mots et les attribuer des poids d'importance. C'est ce qui permet de « contextualiser » le prompt. Le poids du « sub word » « ***Va*** » n'aura pas la même valeur si l'encodeur encode «aujourd'hui il ***va*** faire beau », ou s'il encode « ***vas*** faire tes devoirs ».

Réseau de neurones feed-forward positionnel : Il s'agit d'un réseau de neurones simple, qui traite les représentations des mots après le mécanisme d'attention multi-têtes. Il ajoute des informations non linéaires et capture des relations positionnelles entre les mots.

Après avoir traversé toutes les couches de l'encodeur, l'information d'entrée est transformée en représentations continues, qui sont transmises au décodeur.

· Le décodeur :

Le décodeur a pour rôle de générer la séquence de sortie à partir des représentations continues fournies par l'encodeur. Il est également composé de plusieurs couches identiques superposées, avec trois sous-modules : le mécanisme d'attention multi-têtes (auto-attention), l'attention multi-têtes entre l'encodeur et le décodeur, et le réseau de neurones feed-forward positionnel.

Mécanisme d'attention multi-têtes (auto-attention) : Il fonctionne de la même manière que dans l'encodeur, permettant au décodeur de capturer les dépendances contextuelles entre les mots de la séquence de sortie.

Attention multi-têtes entre l'encodeur et le décodeur : Ce mécanisme d'attention permet au décodeur d'accéder aux représentations continues de l'encodeur et de les utiliser pour générer la séquence de sortie.

Réseau de neurones feed-forward positionnel : Il traite les représentations des mots après les mécanismes d'attention, ajoutant des informations non linéaires et capturant des relations positionnelles entre les mots.

Finalement, le décodeur génère la séquence de sortie en utilisant une couche de sortie linéaire suivie d'une fonction softmax, qui attribue des probabilités aux mots du vocabulaire.

**Attention !!! Vous avez dit attention ?**

Les têtes de lecture dans le mécanisme d'attention multi-têtes des transformers utilisent trois matrices pour effectuer leurs calculs : la matrice de clés (K), la matrice de valeurs (V) et la matrice de requêtes (Q).

· Matrice de requêtes (Q) : Cette matrice est obtenue en projetant les représentations des mots d'entrée sur un ensemble de poids appris, spécifique à chaque tête de lecture. La matrice de requêtes sert à interroger les relations contextuelles entre les mots de la séquence.

· Matrice de clés (K) : Elle est également obtenue en projetant les représentations des mots d'entrée sur un ensemble de poids appris, spécifique à chaque tête de lecture. La matrice de clés est utilisée pour évaluer l'importance relative des autres mots dans la séquence par rapport au mot interrogé par la matrice de requêtes.

· Matrice de valeurs (V) : Comme pour les matrices de requêtes et de clés, la matrice de valeurs est obtenue en projetant les représentations des mots d'entrée sur un ensemble de poids appris, spécifique à chaque tête de lecture. La matrice de valeurs contient les informations contextuelles à partir desquelles le mécanisme d'attention construit les nouvelles représentations de mots.

Le mécanisme d'attention multi-têtes fonctionne en calculant le produit scalaire entre la matrice de requêtes et la matrice de clés, suivi d'une normalisation par la racine carrée de la dimension des clés. Oui fallait écouter pendant les cours de maths 😊

Ensuite, on applique une fonction softmax pour obtenir les poids d'attention, qui sont utilisés pour pondérer la matrice de valeurs. Finalement, la matrice résultante est utilisée pour construire les nouvelles représentations de mots.

Ce processus est réalisé en parallèle pour chaque tête de lecture, et les résultats sont concaténés et transformés par une autre projection linéaire pour obtenir les représentations finales utilisées par les couches suivantes du transformer.

**Ce qu'il faut en retenir pour l'analogie !**

Le GPT (transformer pré-entrainé) va donc utiliser des probabilités conditionnelles pour « deviner » le prochain sub-word qui doit compléter votre « prompt d'entrée ». Son mécanisme attentionnel, lui permet même d'effectuer des calculs complexes de probabilités conditionnelles, lui permettant de « deviner » la prochaine phrase… qui elle-même s'ajoutera aux données d'entrée (en flux continu) pour « deviner » la prochaine, prochaine phrase etc…

Ah bah tu aurais pu commencer par là….

**Les analogies entre la planche de Galton et les réseaux de neurones**

**Distribution et probabilité**

La planche de Galton est un dispositif mécanique qui illustre la distribution normale ou la loi binomiale à travers le mouvement de billes qui tombent sur des clous et se répartissent dans des bacs.

Les réseaux de neurones, en particulier ceux utilisant des fonctions d'activation probabilistes, peuvent également modéliser des distributions de probabilité.

Dans ce sens, les deux systèmes traitent des distributions et des probabilités. Les deux font aussi massivement appel aux probabilités conditionnelles (Probabilité de l'évènement B sachant que l'évènement A est survenu).

Comme nous l'avons vu :

· La planche de Galton va attribuer la probabilité de se trouver en un point topologie particulier, en fonction du chemin parcouru par la bille jusqu'à lors. Forcément dans notre exemple avec une équiprobabilité des routes, cela ne saute pas aux yeux, mais il s'agit bien de probabilités conditionnelles. Il suffit pour s'en convaincre de décider que *p* est différent de 0,5. Par exemple qu'on a 0.75 de probabilité d'aller à gauche, et 0.25 d'aller à droite pour voir que désormais toutes les routes ne sont pas équiprobables.

· Un Large Language Model va essayer de « deviner » avec son encodeur, son décodeur et son mécanisme attentionnel le mot, la phrase la plus probable qui viendra compléter votre « prompt d'entrée ». Pour ce faire, il dispose de plusieurs matrices avec des pondérations probabilistes qui ont été définies par son apprentissage/entrainement sur son jeu de données d'apprentissage.

OK une granny smith est verte, la pelouse aussi (quand nous ne sommes pas en privation hydrique), mais bon ça n'a quand même rien à voir. D'aucuns diront que les poids de la table de Galton sont fixes, souvent égaux à 0,5 chacun… que la table de Galton est pyramidale/triangulaire ce qui n'a rien à avoir avec un plat de lasagne, et que le vecteur d'entrée constitué par une bille qui tombe dans les couches de « picots », ça manque de raffinement. Et ils auraient raison !

Toutefois, faisons une expérience de l'esprit comme les physiciens théoriques du début du siècle aimaient en faire… Oui le chat à la fois mort et vivant :

· Nous pouvons modifier la topologie de la table de Galton, en adoptant une forme rectangulaire avec des clous / picots posés en quinconce d'une couche à l'autre (couche 2n = K clous, couche 2n+1 = K+1 clous). On s'approche là du modèle du plat de lasagne ce qui fait sauter la première objection.

· Ensuite plutôt que de balancer une bille, et d'attendre qu'elle soit dans les couches inférieures pour jeter une nouvelle bille, nous pourrions disposer de N colonnes pré-remplies d'une certaine quantité de billes, différente pour chaque colonne (une distribution d'entrée) qui nous permettrait d'alimenter notre planche de Galton «personnalisée façon plat de lasagnes » avec une plus grande variété de « vecteurs d'entrée ».

Oui c'est bien beau tout ça, mais quid de l'équiprobabilité des chemins ?

· Pour modifier le pattern de probabilité, on pourrait modifier la taille des clous/picots voire leur géométrie (rond, triangulaire, patatoïde, etc..). Et là la physique appliquerait d'elle-même la fonction *softmax* puisque la somme des probabilités pour un seul picot/clou serait toujours de 1, avec comme probabilité de l'évènement A, p(A), et celle de l'évènement A-barre (linkedin ne permet pas d'écrire un A avec une barre dessus), p(A-barre), telle que p(A)+ p(A-barre) =1.

**Agrégation d'effets**

Les deux systèmes agrègent des effets pour produire des résultats.

Dans la planche de Galton, chaque bille subit une série de collisions avec les clous, ce qui détermine finalement la position de la bille dans les bacs.

De même, dans les réseaux de neurones, les signaux d'entrée sont transformés et pondérés par des poids avant d'être transmis à travers les couches du réseau, ce qui

aboutit à une réponse en sortie. D'ailleurs on voit là que les réseaux de neurones font avant tout du traitement de signal numérique.

Dans le cadre des réseaux de neurones, l'apprentissage permet d'ajuster les poids afin de minimiser la différence entre l' « output » (le résultat produit par le réseau de neurones) et le résultat attendu (notamment dans le cadre d'un apprentissage supervisé).

Dans le cas de la planche de Galton, c'est plus « compliqué ». L'apprentissage on le fait comment ?

· Il "suffit" d'associer des paires de vecteurs entrée/sortie, c'est-à-dire une distribution de billes en entrée, et une distribution de billes en sortie. Ensuite, ça risque d'être un peu fastidieux, mais en modifiant la taille, la forme et éventuellement la position des « picots », on modifiera les probabilités des différents chemins, jusqu'à obtenir la distribution de sortie qu'on désire.

Et après ?

· Ca risque d'être pifométrique, et très très long… mais heureusement nous sommes dans le cadre d'une expérience de l'esprit. Mais une fois entraîné, le modèle ne nécessite que de l'énergie gravitationnelle.

Enfin oui peut être… mais ce n'est pas avec une centaines de picots/clous qu'on arrivera à s'adapter à N paires de vecteurs d'entrée et de sortie !

· Et avec 175 Milliards de clous/picots ?

Bien entendu développer des processus attentionnels tels les têtes de lecture avec une table de Galton, serait plus compliqué. Mais si on estime pouvoir approximer/émuler une matrice de pondération à l'aide d'une table de Galton "lasagnifiée", il suffirait d'assembler en cascade plusieurs tables avec leurs pondérations propres pour émuler ce processus d'attention.

**Complexité émergente**

La planche de Galton et les réseaux de neurones montrent tous deux des comportements complexes qui émergent à partir d'interactions simples.

Dans la planche de Galton, des interactions "simples" (en apparence, ou du moins dans leur modalité conceptuelle) de type choc mécanique entre les billes et les clous génèrent une distribution "complexe" de billes dans les bacs. Même si paradoxalement, la planche de Galton semble ordonner le chaos, puisqu'à partir d'une distribution non ordonnée de billes, elle réalise la fonction de masse de la loi binomiale. Ah si seulement j'avais une table de Galton pour ranger mon bureau !

De même, les réseaux de neurones sont composés de neurones simples interconnectés qui, ensemble, sont capables de modéliser des fonctions et de « résoudre » des problèmes complexes.

***Et c'est bien ce phénomène d'émergence qui est le plus « fascinant ».***

Ce phénomène initialement évoqué dans la théorie du chaos, avec des systèmes d'équations différentielles non linéaires, décrivant des systèmes dynamiques non linéaires, et qui sont extrêmement susceptibles aux modifications même mineures des conditions initiales, trouve ici une autre forme d'illustration. On peut aussi penser aux générateurs « simplistes » de fractales qui donnent naissance à des structures complexes d'une beauté intrinsèque qui captive. Dans le même ordre d'idée, la théorie du chaos évoque les attracteurs étranges. Ces attracteurs sont des états métastables vers lesquels les systèmes dynamiques non linéaires peuvent converger. D'ailleurs ils ont pour principales caractéristiques :

· La non-linéarité : Les attracteurs étranges se forment dans des systèmes dynamiques non linéaires, où les variables interagissent de manière complexe et imprévisible (même si unitairement chaque interaction peut apparaître "simple" au point d'être modélisable mathématiquement).

· D'être très sensibilité aux conditions initiales : Les systèmes présentant des attracteurs étranges sont extrêmement sensibles aux conditions initiales, ce qui signifie que des différences infimes dans les conditions de départ peuvent entraîner des résultats radicalement différents. C'est ce qu'on appelle l'effet papillon.

· De présenter des structures fractales : Les attracteurs étranges ont souvent une structure fractale, c'est-à-dire qu'ils présentent des motifs similaires à différentes échelles. Les fractales sont des objets mathématiques qui possèdent une dimension fractale, une mesure de leur complexité géométrique, qui est généralement un nombre non entier. Pour l'anecdote, si on colorie les nombres pairs et impairs du triangle de Pascal, on obtient une structure de type fractal appelée Triangle (ou napperon) de Sierpinski (Sierpinki gasket). Aussi connu sous le nom de joint de culasse (nom donné par Mandelbrot, celèbre pour ses courbes fractales éponymes).

· D'avoir un comportement chaotique : Les attracteurs étranges sont associés à des comportements chaotiques, où des trajectoires infiniment proches peuvent diverger exponentiellement avec le temps. Cela rend la prévision du comportement du système à long terme extrêmement difficile, voire impossible.

D'ailleurs c'est peut être là, la différence fondamentale entre une planche de Galton (surtout si elle possède 175 milliards de picots) et un Large Language Model :

· Pour le Large Language Model, l'émergence de cette « complexité » n'est pas fondée mathématiquement sur un système d'équations différentielles décrivant un système dynamique non linéaire mais sur des matrices de poids probabilistes, et des minimums locaux dans les différents calculs de gradients que les calculs matriciels opèrent (notamment dans les trois matrices de pondération des têtes de lecture attentionnelles).

.

Pour la planche de Galton, la position initiale de la bille, son vecteur quantité de mouvement, la façon dont elle va interagir avec le clou/picot, la forme, la taille, la « texture » de ce dernier sont autant de paramètres qui vont influer un système dynamique non linéaire dont les conditions initiales vont grandement modifier le « vecteur de sortie ». L'émergence dans le cas de la planche de Galton serait donc plus mathématiquement liée à la théorie du chaos.

**Conclusion**

Pour ceux qui arriveront jusqu'à là… J'ai un peu abusé de votre attention contextuelle, il faudra vous mettre à jour vers GPT 10.0 avec 100k tokens.

Dans les deux cas, c'est à la fois « simple » en termes d'architecture du système, voire même des lois qui en gouvernent les mécanismes unitaires. Et la complexité qui émerge de cette simplicité conceptuelle semble liée à un effet de masse.

C'est bien le nombre de ces paramètres / synapses / « picots » qui ajoute autant de « degrés de liberté » au modèle pour s'adapter aux différentes paires « vecteur d'entrée / vecteur de sortie » qui semble permettre cette émergence.

Avec 175 milliards (pour GPT 3.5) et 5 fois plus pour GPT4… on voit alors apparaître ces inférences probabilistes « magiques » qui font penser à certains qu'on a là l'embryon d'une intelligence artificielle générale.

L'intelligence reposerait donc sur une architecture simple que seule la massification des synapses permettrait de faire émerger pour ordonner le chaos ?

On remarquera toutefois, que l'architecture de notre cerveau bien que reposant sur une massification du nombre de synapses, ne ressemble en rien à un plat de lasagne. Au contraire c'est une architecture de type « fractale » qui interconnecte des unités spécifiques de « calculs » (aire visuelle, aire du langage, aires motrices…) avec de grands faisceaux d'axones (voir image illustrative de cet article).

De la même manière que pour le cerveau, la massification synaptique des réseaux de neurones (ou de la table de Galton) rend illusoire la capacité de comprendre en détail le fonctionnement du système, et donc notamment d'expliquer A Posteriori, un résultat, une "décision". Si on peut modéliser une à quelques interactions de façon quasi

déterministe, la masse immense des successions d'interaction fait apparaître cette émergence de la complexité. Dès lors, les Large Language Models seront difficilement compatibles avec la démarche d'une IA éthique : énormes jeux de données difficiles à décrire avec des métadonnées, robustesse pouvant être compromise par un "attracteur étrange", impossibilité de déterminer le comportement du modèle probabiliste à ces échelles de paramètres. Mais cela est une autre histoire (article).