État de l'art : Amélioration de la génération de légendes d'images avec RBM sur le module de captioning de BLIP (blip-image-captioning-base)

Fortuné HOUESSOU
Sous la supervision de M. Jérôme Lacaille

Mars 2025

Abstract

Cet état de l'art présente les travaux existants autour des Vision-Language Models (VLM), notamment BLIP, et explore les potentialités des Machines de Boltzmann Restreintes (RBM) pour améliorer la génération de descriptions textuelles à partir d'images.

Contents

1 Introduction **2**

2 Vision-Language Models (VLM) **2**
2.1 Prétraitement et Encodage 4
2.2 Fusion des Représentations : Alignement Visuel-Linguistique 4

3 BLIP (Bootstrapping Language-Image Pre-training) **7**
3.0.1 Fonctionnement général 7
3.0.2 Architecture globale 8

4 La Machine de Boltzmann Restreinte (RBM) **10**
4.1 Structure d'une RBM 10
4.2 Fonction énergétique 11
4.3 Probabilités associées 11
4.4 Apprentissage 11
4.5 Applications et intérêts 12

1 Introduction

La fusion entre la vision par ordinateur et le traitement du langage naturel a donné naissance à des modèles puissants capables de relier le contenu visuel aux descriptions textuelles. Parmi ces Vision-Language Models (VLM), le Bootstrapping language-image pre-training (BLIP) s'est imposé comme une référence. Cependant, malgré ses performances, les descriptions générées restent souvent génériques ou peu informatives. Pour répondre à cette limite, l'intégration de modèles probabilistes tels que la Machine de Boltzmann Restreinte (RBM) apparaît comme une piste prometteuse afin d'enrichir les représentations latentes et, in fine, améliorer la qualité textuelle.

2 Vision-Language Models (VLM)

Les modèles de Vision Langage (VLM) sont des architectures multimodales qui combinent le traitement visuel et linguistique pour accomplir des tâches variées telles que la génération de légendes d'images, la recherche d'images par texte, ou l'association image-texte. Ces modèles ont gagné en popularité grâce à leur capacité à fusionner les informations textuelles et visuelles, permettant ainsi de résoudre des problèmes complexes en vision par ordinateur et en traitement du langage naturel. Parmi les modèles les plus connus dans ce domaine figurent CLIP [11] et BLIP [8].

BLIP, en particulier, adopte une approche de pré-entraînement utilisant des données web massives et combine un encodeur d'image avec un décodeur de texte pour générer des descriptions adaptées aux images. Cette approche a démontré son efficacité en génération de légendes et interprétation multimodale. De même, CLIP [11] se distingue par sa capacité à associer des images et des descriptions textuelles, facilitant ainsi des applications comme la recherche d'images par texte et l'analyse d'images à partir de requêtes textuelles.

Les VLMs sont également présents dans des travaux comme VisualBERT [9], qui fusionne les informations visuelles et textuelles dans un modèle transformeur pour des tâches de compréhension multimodale, telles que la réponse à des questions visuelles. Ce modèle a montré qu'il est possible d'aligner les représentations visuelles et textuelles dans un espace partagé pour des performances solides sur des tâches de vision et de langage.

Un autre modèle significatif dans ce domaine est ViLT [7], qui a introduit une approche simplifiée, ne recourant ni aux convolutions ni à une supervision régionale, permettant de traiter simultanément les images et les textes tout en réduisant la complexité du modèle. Cela a ouvert la voie à des applications plus légères tout en maintenant des performances compétitives.

Dans le cadre de la conduite autonome et d'autres applications de sécurité, des travaux comme VLM-RL [6] ont exploré l'intégration des VLMs avec l'apprentissage par renforcement pour améliorer la prise de décision basée sur des informations multimodales, notamment en gestion de scénarios visuels et textuels complexes.

Les capacités de raisonnement spatial sont également un domaine d'amélioration important pour les VLMs, comme le montre SpatialVLM [3], qui cherche à doter ces modèles de la capacité à effectuer des raisonnements géométriques et spatiaux, ce qui est essentiel pour des applications telles que la navigation autonome et la manipulation d'objets en 3D.

Enfin, des modèles comme VILA [5] et Xmodel-VLM [10] se concentrent sur des techniques de pré-entraînement et d'optimisation pour rendre les VLMs plus efficaces

et adaptés aux environnements de production, en réduisant la taille des modèles tout en préservant leur capacité à traiter des informations visuelles et textuelles de manière précise.

Dans la suite de ce rapport sur l'État de l'Art des VLMs, de Blip et RBM, nous commençons par présenter de façon générale, le fonctionnement d'un VLM puis le role des mathématiques derrière ces modèles, et enfin nous traiterons les différents modèles qui feront l'objet de ce sujet à savoir BLIP (Bootstrapping Language-Image Pre-training) et RBM (Restricted Boltzmann machine).

Les **modèles de Vision et Langage** (VLMs) sont des systèmes qui essaient de comprendre à la fois des **images** et des **mots**. Imaginons qu'on montre une image de chat à un enfant et qu'on lui dise "C'est un chat". Le cerveau de l'enfant va faire le lien entre l'image du chat et le mots "chat". C'est exactement ce que font les VLMs, mais d'une manière beaucoup plus complexe et dans un ordinateur !

Comment ça marche concrètement ?

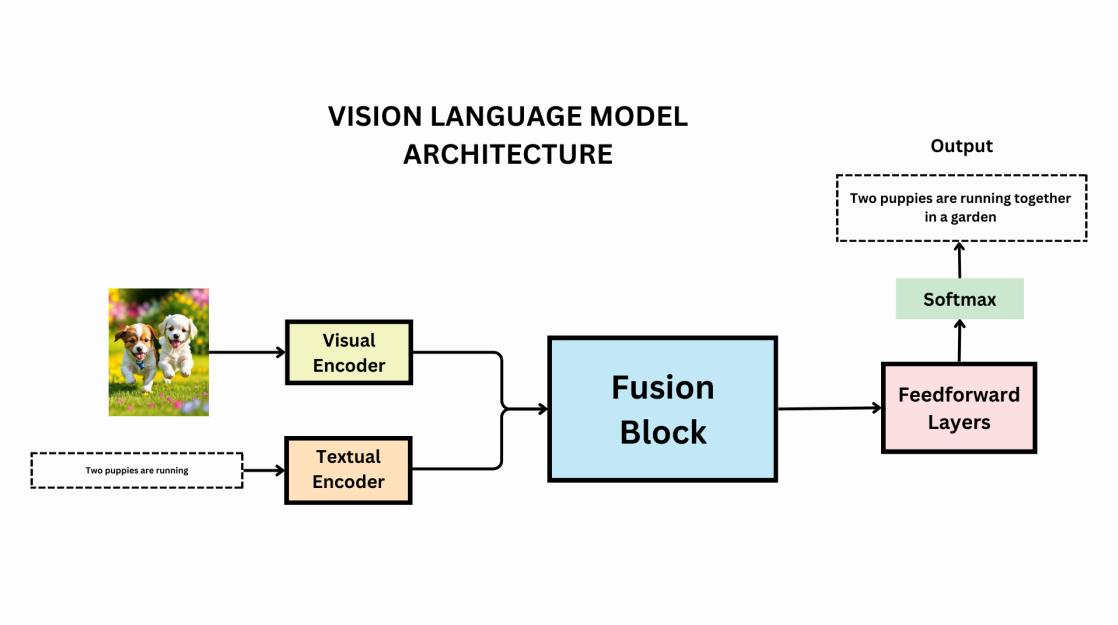

Figure 1: VLM

Le **VLM** combine la compréhension des **images** et des **mots**. Il analyse d'abord l'image pour en extraire des informations visuelles (formes, objets, etc.) via un réseau de neurones. Ensuite, les mots sont transformés en représentations que le modèle peut comprendre. Enfin, le VLM fusionne les informations des images et des mots pour accomplir des tâches comme l'association image-texte ou la génération de descriptions d'images. L'objectif est d'apprendre à lier efficacement ces deux modalités pour mieux comprendre et interagir avec le monde visuel et linguistique simultanément. C'est comme un Homme qui entends le mot "chat" et sait un peu à quelle forme s'attendre, parce que des chats il en a beaucoup vu, de différentes couleurs et dans différents horizons.

Un peu de formalisme :

L'entrainement du modèle BLIP s'éffectue en plusieurs étapes :

2.1 Prétraitement et Encodage

Soit x une entrée visuelle (image) et t une entrée textuelle (texte). Dans la plupart des VLMs, l'image x est d'abord passée à travers un encodeur d'image f_{image}, typiquement un réseau de neurones convolutifs (CNN) ou un modèle transformer ou vision encodeur adapté à l'image (par exemple, Vision Transformer, ViT). L'objectif est de projeter l'image dans un espace de représentation latente $\mathcal{Z}_{\text{image}} \in R^d$.

$$\mathcal{Z}_{\text{image}} = f_{\text{image}}(x)$$

De même, le texte t est encodé par un modèle de langage, tel que le Transformer, en une représentation vectorielle $\mathcal{Z}_{\text{texte}} \in R^d$. Le processus de traitement du texte suit un encodage similaire avec une fonction f_{texte}, qu'on ne connait pas à priori (BERT par exemple).

$$\mathcal{Z}_{\text{texte}} = f_{\text{texte}}(t)$$

Dans le processus d'embedding, certains modèles de VLM utilisent du : self-attention.

2.2 Fusion des Représentations : Alignement Visuel-Linguistique

Pour regrouper ces deux embeddings dans un même espace latent, il y a deux grandes stratégies :

A. Projection directe

On passe Z_{image} et Z_{texte} dans des Multi-Layer Perceptrons (MLP) [4] ou des couches linéaires afin de les projeter dans un **même espace latent**. Une fois cette projection effectuée, on calcule simplement la distance entre ces deux représentations, par exemple via une similarité:

$$\text{similarité}(Z_{\text{image}}, Z_{\text{texte}})$$

Les fonctions de similarité font l'objet d'une estimation de la distance entre deux embedding. C'est une façon de représenter par un nombre les différences ou rapprochements entre entités.

Fonctions de similarité

Basées sur la distance ou l'angle

- **Similarité cosinus (Cosine Similarity)** :

$$\text{sim}(x, y) = \frac{x \cdot y}{\|x\|\|y\|}$$

- **Distance euclidienne (Euclidean Distance)** :

$$d(x, y) = \sqrt{\sum_i (x_i - y_i)^2}$$

Souvent transformée en :

$$\text{sim}(x, y) = \frac{1}{1 + d(x, y)}$$

Basées sur des probabilités

- **Divergence de Kullback-Leibler (KL Divergence)** :
Pour deux distributions de probabilités P et Q :

$$D_{KL}(P \parallel Q) = \sum_i P(i) \log \left(\frac{P(i)}{Q(i)} \right)$$

Pas symétrique et non bornée.

- **Divergence de Jensen-Shannon (JS Divergence)** :
Variante symétrique et plus stable du KL :

$$D_{JS}(P, Q) = \frac{1}{2} D_{KL}(P \parallel M) + \frac{1}{2} D_{KL}(Q \parallel M)$$

où

$$M = \frac{1}{2}(P + Q)$$

Basées sur des ensembles

- **Similarité de Jaccard (Jaccard Similarity)** :
Pour des ensembles A et B :

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

- **Coefficient de recouvrement (Overlap Coefficient)** :

$$\text{Overlap}(A, B) = \frac{|A \cap B|}{\min(|A|, |B|)}$$

C'est précisément l'approche adoptée par le modèle CLIP.

B. Cross-Attention

Dans certains modèles comme BLIP, avant la projection des données bimodales (image/texte) dans l'espace latent,un mecanisme d'attention croisée (cross-attention) est utilisé dans les architectures de type transformer. Ce mécanisme permet de calculer une attention mutuelle entre les représentations des images et du texte afin d'extraire des correspondances pertinentes entre les modalités. Les articles [2] et [1] introduisent et

expliquent dans les détails le mécanisme d'attention. Le calcul de l'attention croisée peut être formalisé comme suit :

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

où Q (Query) est la requête (environnement textuel ou visuel), K (Key) est la clé (respectivement l'information visuelle ou textuelle), et V (Value) est la valeur : [14] et [13]. Ce mécanisme est appliqué pour apprendre les relations entre $\mathcal{Z}_{\text{image}}$ et $\mathcal{Z}_{\text{texte}}$, en ajustant les poids des connexions neuronales selon l'importance de chaque élément textuel ou visuel pour le modèle.

Optimisation et Apprentissage

L'entraînement des VLMs repose sur la minimisation d'une fonction de perte $\mathcal{L}$ qui permet au modèle d'apprendre à associer de manière optimale les informations visuelles et textuelles. Par exemple, une fonction de perte typique dans le cas d'une tâche de classification d'images par texte est la suivante :

$$\mathcal{L} = \sum_i \left[\log P(y_i|\mathcal{Z}_{\text{image}}) + \log P(y_i|\mathcal{Z}_{\text{texte}})\right]$$

où y_i est l'étiquette associée à l'entrée x_i ou t_i. Cette formulation implique une double minimisation des erreurs sur les prédictions visuelles et textuelles.

Un autre type de perte, utilisé dans les modèles de type *contrastive learning* (comme CLIP [11]), repose sur l'idée de maximiser la similarité entre les représentations textuelles et visuelles d'une même instance tout en minimisant la similarité entre les paires d'images et de textes différentes. La fonction de perte contraste peut être formulée ainsi :

$$\mathcal{L}_{\text{contrastive}} = -\sum_i \log \frac{\exp(\text{sim}(\mathcal{Z}^i_{\text{image}}, \mathcal{Z}^i_{\text{texte}}))}{\sum_j \exp(\text{sim}(\mathcal{Z}^i_{\text{image}}, \mathcal{Z}^j_{\text{texte}}))}$$

où $\text{sim}(a, b)$ représente la similarité entre les vecteurs a et b, souvent mesurée par un produit scalaire ou un cosinus.

Architecture générale d'un VLM (Vision-Language Model)

1 Embedding

- Image → Patchs (ViT) ou CNN → Embeddings visuels.
- Texte → Tokenisation → Embeddings textuels.

2 Self-Attention(selon le modèle)

- Appliqué séparément sur les embeddings :
- Image → Self-attention entre patchs.
- Texte → Self-attention entre tokens.

3 Cross-Attention (si fusion, cas de BLIP)

- Texte attend sur l'image ou inversement.
- Q = texte, K, V = image (ou l'inverse selon le modèle).
- Permet d'associer finement les informations. (On dit que le texte et l'image se regardent)

4 Projection

- Alignement dans un même espace vectoriel via couches linéaires.
- Image → projection.
- Texte → projection.

5 Similarité

- Calcul par produit scalaire ou cosin similarity entre vecteurs projetés.

6 Contrastive Loss

- But :
  - Maximiser la similarité des bonnes paires (image, texte liés).
  - Minimiser celle des paires incorrectes.

3 BLIP (Bootstrapping Language-Image Pre-training)

BLIP (*Bootstrapping Language-Image Pre-training*) [8] est un framework conçu pour améliorer les performances sur les tâches vision-langage. Il se distingue par sa capacité à exploiter à la fois des **données bruitées web-scale** et des annotations de qualité, grâce à une architecture flexible et un apprentissage par bootstrapping.

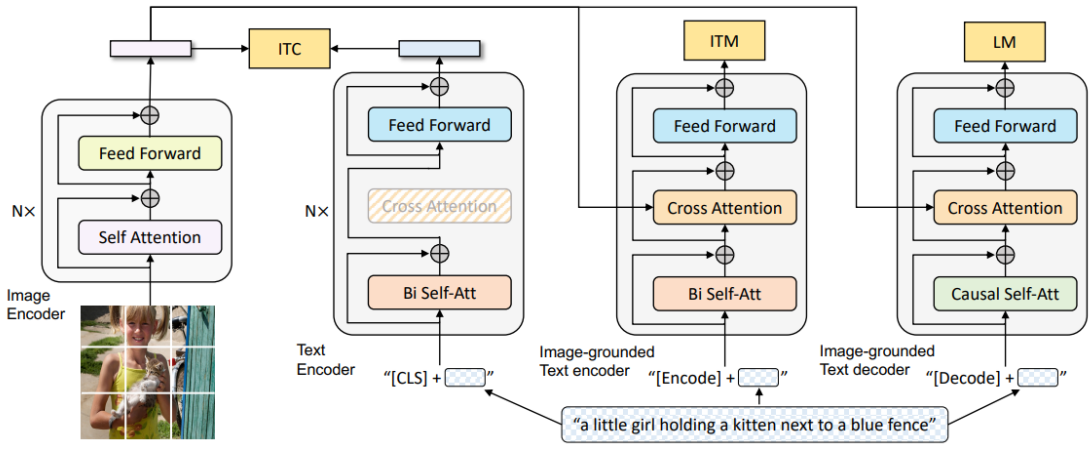

Figure 2: BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation

3.0.1 Fonctionnement général

BLIP repose principalement sur trois tâches de pré-entraînement complémentaires :

- **Image-Text Contrastive Learning (ITC)** : maximise la similarité entre les représentations d'une image et de sa légende correspondante, et minimise celles des paires non correspondantes.
  L'ITC est une fonction de perte utilisée pour aligner les représentations des images et des textes dans un espace commun. Elle est inspirée de la InfoNCE loss et fonctionne comme une perte contrastive entre les paires image-texte positives et négatives.

$$\mathcal{L}_{ITC} = -\frac{1}{N}\sum_{i=1}^{N}\left(\log\frac{e^{\text{sim}(z_i^I, z_i^T)/\tau}}{\sum_{j=1}^{N} e^{\text{sim}(z_i^I, z_j^T)/\tau}}\right) \tag{1}$$

  où z_i^I et z_i^T sont les embeddings d'image et de texte, sim est la similarité cosinus, τ est la température (temperature scaling) et N la taille du lot des données traitées.

- **Image-Text Matching (ITM)** : discrimine si une paire image-texte est correcte ou incorrecte via une classification binaire, souvent couplée à une fonction de perte d'entropie croisée classique.

$$\mathcal{L}_{ITM} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log P(y_i|I_i, T_i) + (1-y_i)\log(1 - P(y_i|I_i, T_i))\right]$$

- **Image-Conditioned Language Modeling (LM)** : génère du texte conditionnellement à une image.

$$\mathcal{L}_{LM} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T_i}\log P(w_t^i | w_{<t}^i, I_i)$$

  où :

  - N est le nombre d'exemples dans le batch.
  - T_i est la longueur de la séquence de texte pour l'exemple i. (nombre de token)
  - w_t^i est le mot à la position t dans la séquence i.
  - $w_{<t}^i$ représente tous les mots précédents dans la séquence.
  - I_i est l'image associée.
  - $P(w_t^i \mid w_{<t}^i, I_i)$ est la probabilité prédite du mot w_t^i étant donné l'image et les mots précédents.

3.0.2 Architecture globale

BLIP s'appuie sur une combinaison de :

- **Encoders visuels** (souvent des Vision Transformers - ViT)

- **Encoders textuels** (type BERT)
- **Décoders textuels** (transformer auto-régressif)

Pendant l'entrainement, les modèles apprennent dans un espace latent partagé où les embeddings d'images et de textes sont projetés, permettant une compatibilité sémantique entre les deux modalités.

Dans ce projet, nous nous interessons plus précisément au modèle BLIP-Captionner du framework BLIP, qui en sortie, génère une description textuelle en réponse à une image reçue en entrée.

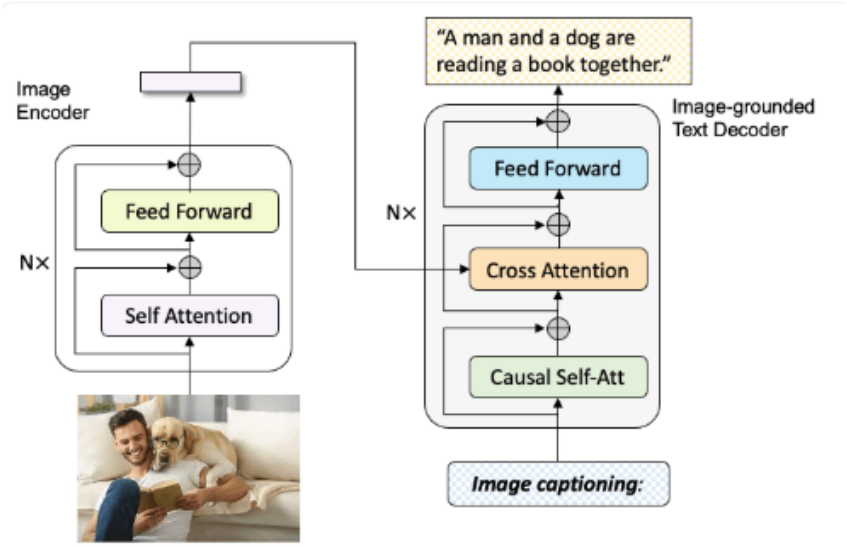

Figure 3: Blip-image-captioning-base

4 La Machine de Boltzmann Restreinte (RBM)

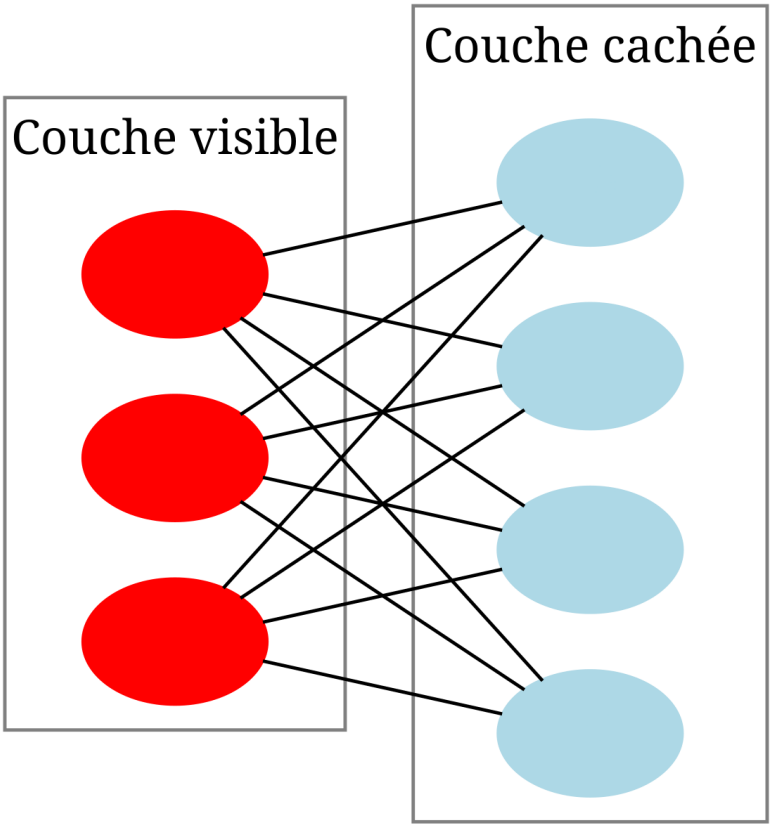

Figure 4: Restricted Boltzmann machine

La Machine de Boltzmann Restreinte (RBM), introduite par [12] et popularisée pour l'apprentissage non supervisé, est un modèle probabiliste génératif appartenant à la famille des modèles énergétiques. Elle est utilisée pour apprendre la distribution jointe de variables observables et cachées et constitue l'un des éléments fondamentaux des architectures de deep learning comme les Deep Belief Networks (DBN).

4.1 Structure d'une RBM

Une RBM est un réseau biparti composé de :

- **Une couche visible** $\mathbf{v} = (v_1, v_2, \ldots, v_m)$ représentant les données observables.
- **Une couche cachée** $\mathbf{h} = (h_1, h_2, \ldots, h_n)$ permettant de capturer des dépendances complexes (cachées).

Contrairement aux Machines de Boltzmann standards, il n'y a **pas de connexions intra-couche** :

- Aucune connexion entre les neurones visibles.
- Aucune connexion entre les neurones cachés.

4.2 Fonction énergétique

Le cœur de la RBM repose sur une fonction d'énergie définie par :

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{v}^\top \mathbf{W}\mathbf{h} - \mathbf{b}^\top \mathbf{v} - \mathbf{c}^\top \mathbf{h}$$

où :

- $\mathbf{W}$ est la matrice des poids entre les couches visible et cachée.
- $\mathbf{b}$ et $\mathbf{c}$ sont les biais des couches visible et cachée respectivement.
- $\mathbf{v}$ et $\mathbf{h}$ sont les états des neurones sur les couches visible et cachée respectivement.

4.3 Probabilités associées

La probabilité jointe d'un état $(\mathbf{v}, \mathbf{h})$ est donnée par la dtistribution de Gibbs :

$$P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} e^{-E(\mathbf{v},\mathbf{h})}$$

avec Z la **partition** :

$$Z = \sum_{\mathbf{v},\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}$$

Le modèle apprend ainsi à approximer la distribution des données en maximisant la vraisemblance $P(\mathbf{v})$ par :

$$P(\mathbf{v}) = \frac{1}{Z} \sum_{\mathbf{h}} e^{-E(\mathbf{v},\mathbf{h})}$$

4.4 Apprentissage

L'apprentissage des paramètres $(\mathbf{W}, \mathbf{b}, \mathbf{c})$ s'effectue typiquement via l'algorithme **Contrastive Divergence (CD)** proposé par Hinton. Il repose sur une estimation approchée du gradient du logarithme de la vraisemblance :

$$\Delta w_{ij} = \epsilon(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model})$$

où :

- ϵ est le taux d'apprentissage.
- $\langle \cdot \rangle_{data}$ est l'espérance sous la distribution empirique des données.
- $\langle \cdot \rangle_{model}$ est l'espérance sous la distribution modélisée.

Le processus implique des étapes de **Gibbs Sampling**, alternant entre la mise à jour de $\mathbf{h}$ et $\mathbf{v}$:

$$P(h_j = 1|\mathbf{v}) = \sigma\left(\sum_i w_{ij} v_i + c_j\right)$$

$$P(v_i = 1|\mathbf{h}) = \sigma\left(\sum_j w_{ij} h_j + b_i\right)$$

où $\sigma(x)$ est la fonction sigmoïde.

4.5 Applications et intérêts

Les RBM sont particulièrement adaptées pour :

- La réduction de dimensionnalité.
- L'initialisation de réseaux profonds (pre-training).
- La modélisation générative de données complexes (images, textes...).
- L'extraction de caractéristiques non supervisée.
- Elles peuvent aussi être combinées avec d'autres modèles, comme les VLM, pour apprendre des représentations conjointes multi-modales.

Ces trois dernières applications sont au cœur de notre étude...

References

[1] Noam Shazeer Ashish Vaswani et al. Attention is all you need. *https://arxiv.org/abs/1706.03762*, 2023.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[3] Yul Choi et al. Spatialvlm: Integrating spatial reasoning with vision-language models. *arXiv preprint arXiv:2105.08752*, 2021.

[4] Simon Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 1998.

[5] Jialei Hou et al. Vila: Vision-and-language pre-training with large-scale datasets. *arXiv preprint arXiv:2103.12824*, 2021.

[6] Haesu Hwang et al. Vlm-rl: Visual-language models for reinforcement learning. *arXiv preprint arXiv:2004.06592*, 2020.

[7] Wonjae Kim et al. Vilt: Vision-and-language transformer without convolution or region supervision. *arXiv preprint arXiv:2102.03334*, 2021.

[8] Junnan Li et al. Blip: Bootstrapping language-image pre-training. *arXiv preprint arXiv:2201.12086*, 2022.

[9] Luowei Li et al. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019.

[10] Wei Liu et al. Xmodel-vlm: Efficient cross-modal representation learning. *arXiv preprint arXiv:2104.06923*, 2021.

[11] Alec Radford et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021.

[12] Paul Smolensky. Information processing in dynamical systems: Foundations of harmony theory. *Parallel distributed processing: Explorations in the microstructure of cognition*, 1:194–281, 1986.

[13] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*, 2019.

[14] Yao-Hung Hubert Tsai et al. Multimodal transformer for unaligned multimodal language sequences. *arXiv preprint arXiv:1906.00295*, 2019.