Date remise: Lundi 14 novembre, 23h

Instructions

- *Montrez votre démarche pour toutes les questions !*
- *Utilisez LaTeX et le modèle que nous vous fournissons pour rédiger vos réponses. Vous pouvez réutiliser la plupart des raccourcis de notation, des équations et/ou des tableaux. SVP voir la politique des devoirs sur le site web du cours pour plus de détails.*
- *Vous devez soumettre toutes vos réponses sur la page Gradescope du cours*
- *Les TAs pour ce devoir sont* ***Arian Khorasani et Nanda Harishankar Krishna***

Question 1 (2-5-5-5-3). (**Rétropropagation du gradient dans Réseau de neurones récurrents**)

Considérez l'unité récurrente suivante:

$$\boldsymbol{h}_t = \sigma(\boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{U}\boldsymbol{h}_{t-1})$$
$$\boldsymbol{y}_t = \boldsymbol{V}\boldsymbol{h}_t$$

oú σ denotes the logistic sigmoid function. Que $\boldsymbol{z}_t$ soit la vraie cible de la prédiction $\boldsymbol{y}_t$ et considère la somme des pertes au carré $L = \sum_t L_t$ oú $L_t = ||\boldsymbol{z}_t - \boldsymbol{y}_t||_2^2$.

Dans cette question, notre but est d'obtenir une expression pour les gradients $\nabla_{\boldsymbol{W}} L$ and $\nabla_{\boldsymbol{U}} L$.

1. Tout d'abord, remplissez le graphique de calcul suivant pour ce l'unité récurrente, déroulé pendant 3 étapes (de $t = 1$ à $t = 3$). Étiqueter chaque nœud avec l'unité cachée correspondante et chaque bord avec le poids correspondant.

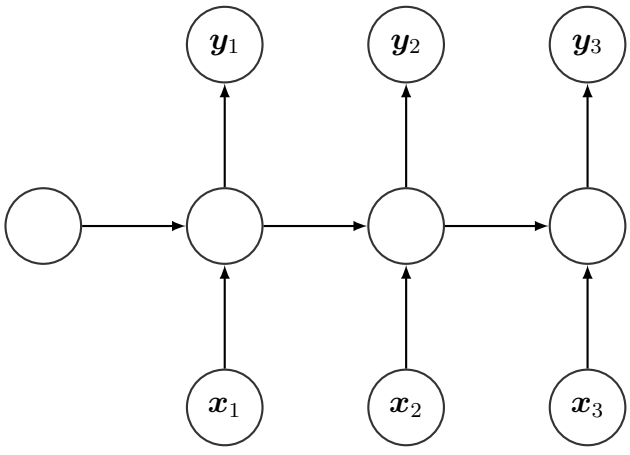

FIGURE 1 – Graphique de calcul de l'unité récurrente déroulé en trois temps.

2. Dériver une expression récursive pour le gradient total $\nabla_{\boldsymbol{h}_t} L$ en termes de $\nabla_{\boldsymbol{h}_t} L_t$ et $\nabla_{\boldsymbol{h}_{t+1}} L$.

3. obtenir une expression pour $\nabla_{\boldsymbol{h}_t} L_t$ et $\frac{\partial \boldsymbol{h}_{t+1}}{\partial \boldsymbol{h}_t}$.
4. Dérivez maintenant $\nabla_{\boldsymbol{W}} L$ et $\nabla_{\boldsymbol{U}} L$ en fonction de $\nabla_{\boldsymbol{h}_t} L$, en utilisant les résultats des deux sous-questions précédentes.
   Indice: Il pourrait être utile de tenir compte de la contribution des matrices de poids lors du calcul de la unité cachée récurrente à un moment particulier t et comment ces contributions pourraient être agrégées.
5. Éviter le problème de l'explosion ou de la disparition des gradients, nous pouvons utiliser BPTT tronqué pour calculer les gradients pour les mises à jour des paramètres dans les RNNs. Que pouvez-vous dire au sujet des estimations des gradients du BPTT tronqué – sont-ils impartiaux ou biaisés ? Pourquoi vous le pensez ? Bien que le BPTT tronqué puisse aider à atténuer l'explosion ou la disparition des gradients, Quels problèmes entrevoyez-vous avec cette approche, par exemple dans les tâches impliquant la langue ?

Answer 1. Q.1.1

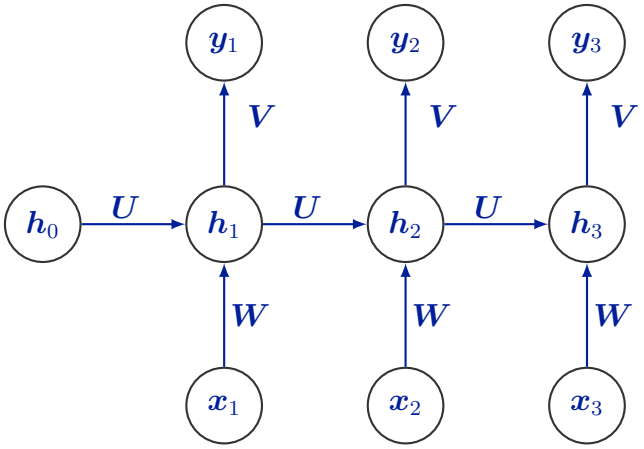

FIGURE 2 – Graphique de calcul de l'unité récurrente déroulé en trois temps.

Q1.2.

Or, h_t est construit de manière récursive croissante, ainsi pour $b < t$, $L_b = f(h_b, x_b)$ sont indépendants de h_t. Ainsi, $\nabla_{\boldsymbol{h}_t} L_b = 0$ pour $b < t$.

Part conséquent,

$$\nabla_{\boldsymbol{h}_t} L = \sum_{b \geq t} \nabla_{\boldsymbol{h}_t} L_b$$

Ainsi,

$$\nabla_{\boldsymbol{h}_t} L = \nabla_{\boldsymbol{h}_{t+1}} L + \nabla_{\boldsymbol{h}_t} L_t$$

Q.1.3.

On a que $\boldsymbol{h}_{t+1} = \sigma(\boldsymbol{W}\boldsymbol{x}_{t+1} + \boldsymbol{U}\boldsymbol{h}_t)$, ainsi

$$\begin{aligned}
\frac{\partial \boldsymbol{h}_{t+1}[i]}{\partial \boldsymbol{h}_t[j]} &= \frac{\partial \sigma(\boldsymbol{W}\boldsymbol{x}_{t+1} + \boldsymbol{U}\boldsymbol{h}_t)[i]}{\partial \boldsymbol{h}_t[j]} \\
&= \frac{\partial \sigma(\boldsymbol{W}\boldsymbol{x}_{t+1}[i] + \sum_k \boldsymbol{U}[i,k]\boldsymbol{h}_t[k])}{\partial \boldsymbol{h}_t[j]} \\
&= \boldsymbol{U}[i,j]\sigma(\boldsymbol{W}\boldsymbol{x}_{t+1} + \boldsymbol{U}\boldsymbol{h}_t)[i](1 - \sigma(\boldsymbol{W}\boldsymbol{x}_{t+1} + \boldsymbol{U}\boldsymbol{h}_t)[i]) \\
&= \boldsymbol{U}[i,j]\boldsymbol{h}_{t+1}[i](1 - \boldsymbol{h}_{t+1}[i])
\end{aligned}$$

Ainsi,

$$\frac{\partial \boldsymbol{h}_{t+1}}{\partial \boldsymbol{h}_t} = \mathrm{diag}(\boldsymbol{h}_{t+1} \odot (1 - \boldsymbol{h}_{t+1}))\boldsymbol{U}$$

On sait que

$$\frac{\partial}{\partial x}\left(\|M \cdot x\|_2^2\right) = 2 \cdot M^\top \cdot M \cdot x$$

Et que $L_t = \|\boldsymbol{z}_t - \boldsymbol{y}_t\|_2^2 = \|\boldsymbol{z}_t - \boldsymbol{V}\boldsymbol{h}_t\|_2^2$

Par conséquent,

$$\nabla_{h_t} L_t = \frac{\partial}{\partial \boldsymbol{h}_t}\|\boldsymbol{z}_t - \boldsymbol{V}\boldsymbol{h}_t\|_2^2 = -2 \cdot \boldsymbol{V}^\mathrm{T} \cdot (\boldsymbol{z}_t - \boldsymbol{V} \cdot \boldsymbol{h}_t)$$

Q.1.4. On a,

$$\nabla_{\boldsymbol{W}} \boldsymbol{L} = \sum_t \nabla_{\boldsymbol{W}} \boldsymbol{L}_t$$

Pour t fixé quelconque, on L_t qui peut s'écrire comme une fonction des résultats intérmédiaires h_k pour $k \leq t$. On applique donc la chain rule pour obtenir:

$$\begin{aligned}
\nabla_{\boldsymbol{W}} \boldsymbol{L}_t &= \sum_{k \leq t} \frac{\partial \boldsymbol{L}_t}{\partial \boldsymbol{h}_k}\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{W}} \\
&= \sum_{k \leq t} \frac{\partial \boldsymbol{L}_t}{\partial \boldsymbol{h}_t}\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_k}\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{W}} \text{ En réutilisant la chain rule sur } L_t \\
&= \nabla_{h_t} L_t \sum_{k \leq t} \frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_k}\frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{W}}
\end{aligned}$$

Posons

$$\boldsymbol{S}_t = \sum_{k \leq t} \frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{h}_k} \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{W}}$$

Ainsi,

$$\begin{aligned}
\nabla_{\boldsymbol{W}} \boldsymbol{L} &= \sum_t \nabla_{\boldsymbol{W}} \boldsymbol{L}_t \\
&= \sum_t \nabla_{\boldsymbol{h}_t} \boldsymbol{L}_t \boldsymbol{S}_t \\
&= \sum_t \nabla_{\boldsymbol{h}_t} \boldsymbol{L} * \boldsymbol{S}_t - \nabla_{\boldsymbol{h}_{t+1}} \boldsymbol{L} * \boldsymbol{S}_t \\
&= \sum_{t>1} \nabla_{\boldsymbol{h}_t} \boldsymbol{L} * (\boldsymbol{S}_t - \boldsymbol{S}_{t-1}) + \nabla_{\boldsymbol{h}_1} \boldsymbol{L} * \boldsymbol{S}_1 \\
&= \sum_t \nabla_{\boldsymbol{h}_t} \boldsymbol{L} * \frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{W}}
\end{aligned}$$

Et on a

$$\begin{aligned}
\frac{\partial \boldsymbol{h}_t[i]}{\partial \boldsymbol{W}[j,k]} &= \frac{\partial \sigma\left(\boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{U}\boldsymbol{h}_{t+1}\right)[i]}{\partial \boldsymbol{W}[j,k]} \\
&= \delta_{j,k} \boldsymbol{x}_t[i] \boldsymbol{h}_t[i](1 - \boldsymbol{h}_t[i])
\end{aligned}$$

Ainsi, on obtient

$$\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{W}} = \boldsymbol{x}_t^T \otimes \operatorname{diag}\left(\boldsymbol{h}_t \odot (1 - \boldsymbol{h}_t)\right)$$

Enfin, on a :

$$\nabla_W L = \sum_t \nabla_{\boldsymbol{h}_t} L * \boldsymbol{x}_t^T \otimes \operatorname{diag}\left(\boldsymbol{h}_t \odot (1 - \boldsymbol{h}_t)\right)$$

On applique le même processus pour U,

$$\nabla_U L = \sum \nabla_{\boldsymbol{h}_t} \boldsymbol{L} * \frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{U}} = \sum_t \nabla_{\boldsymbol{h}_t} L * \boldsymbol{h}_{t-1}^T \otimes \operatorname{diag}\left(\boldsymbol{h}_t \odot (1 - \boldsymbol{h}_t)\right)$$

Q.1.5.

Pour de longues séquences de phrases ; on peut avoir une dépéndace instable entre lea première partie de la séquence et la fin de la séquence. Ainsi, durant la backprop l'instabilité se traduit dans un gradient de loss par rapport aux poids qui peut soit disparaitre ou bien être croitre énormèment.

Les estimations du gradients par Truncanted BPTT sont **biaisées**, parce que le modèle saute les étapes à partir d'un certain seuil pour tronquer . Ainsi, les dépendances lointaines entre dans les longues s"quences sont négligées.

Ainsi, pour la traduction ou les chatbots le Truncanted BPTT peut donné naissance à des erreurs quand la phrase d'input est assez longue.

Question 2 (5-5-3-5-2)**.** (**Transformateurs sont GNNs**)

L'apprentissage des représentations des intrants est le socle de tous les réseaux neuronaux. Au cours des dernières années, Les modèles de transformateurs ont été largement adaptés aux tâches de modélisation de séquence dans la vision et domaines de langue, tandis que les réseaux neuronaux graphiques (GNN) ont été efficaces dans la construction de représentations de nœuds et les bords dans les données graphiques. Dans les questions suivantes, nous allons explorer les Transformers et les GNN, et dessiner quelques connexions entre eux.

Contexte:

Examinons d'abord un modèle graphique. Nous définissons un graphique dirigé $G = \{V, E\}$ où V est l'ensemble de tous les sommets et E est l'ensemble de tous les bords. pour $\forall v_i \in V$, laissez-nous définir $\mathcal{N}(v_i)$ comme ensemble de tous les voisins de v_i avec des bords sortants vers v_i. v_i a une représentation d'état h_i^t à chaque étape de temps t.

Les valeurs de h_i^t sont mis à jour en parallèle, en utilisant le même instantané du graphique à un pas de temps donné. Les procédures sont les suivantes : Nous devons d'abord agréger les données entrantes $H'_{it} = \{f_{ji}(h_j^t)|\forall j, v_j \in \mathcal{N}(v_i)\}$ de voisins utilisant la fonction $Agg(H'_{it})$. Notez que les données entrantes de chaque voisin est une version transformée de sa représentation en utilisant fonction f_{ji}. la fonction d'agrégation $Agg(H'_{it})$ peut être quelque chose comme la somme ou la moyenne des éléments dans H'_{it}.

Disons que l'état initial à l'étape 0 est h_i^0. Définissons maintenant la règle de mise à jour pour h_i^t à un pas de temps $t + 1$ comme suit:

$$h_i^{t+1} = q(h_i^t, Agg(H'_{it})) \tag{1}$$

où q est un fonction – $Q_t : \{H_t, Agg(H'_t)\} \to H_t$, où $H_t = \{h_n^t|\forall n, v_n \in V\}$.

Maintenant, jetons un coup d'œil aux modèles Transformer. Rappelons que les modèles Transformer construisent des caractéristiques pour chaque mot en fonction des caractéristiques de tous les autres mots avec un mécanisme d'attention sur eux, tandis que les RNNs mettent à jour les fonctionnalités de manière séquentielle.

Pour représenter un mécanisme d'attention du modèle Transformer, définissons une représentation de caractéristiques h_i pour le mot i dans la phrase S. Nous avons l'équation standard pour la mise à jour de l'attention à la couche l en fonction de l'autre mot j comme suit:

$$h_i^{l+1} = \texttt{Attention}(Q^l h_i^l, K^l h_j^l, V^l h_j^l) \tag{2}$$

$$= \sum_{j \in S} \left(\texttt{softmax}_j(Q^l h_i^l \cdot K^l h_j^l) V^l h_j^l\right) \tag{3}$$

Où Q^l, K^l, V^l sont des matrices de poids pour « Query », « Key » et « Value ». Q est une matrice qui contient des représentations vectorielles d'un mot dans la phrase, Tandis que K est une matrice contenant des représentations pour tous les mots de la phrase. V est une autre matrice similaire à K qui a des représentations pour tous les mots de la phrase. Pour vous rafraîchir la mémoire au sujet du modèle de transformateur, vous pouvez vous reporter à paper.

En vous fondant sur les renseignements de base ci-dessus, répondez aux questions suivantes :

(2.1) Si l'opération d'agrégation pour $Agg(H'_{it})$ est la somme de la représentation de tous les sommets adjacents, Réécrire l'équation 1 en remplaçant $Agg(H'_{it})$) en termes de $\mathcal{N}$, f, et h.

(2.2) Considérons le graphique G de la figure 3. Les valeurs des sommets à l'étape t sont les suivantes

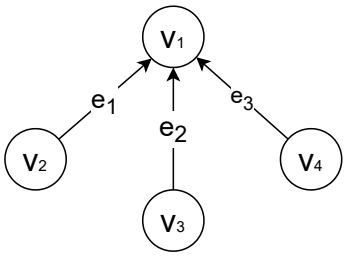

FIGURE 3 – Graphique G pour Q2.2

:

$$h_1^t = [1, -1] \quad h_2^t = [-1, 1] \quad h_3^t = [0, -1] \quad h_4^t = [1, 0] \tag{4}$$

(Ce sont des vecteurs de rangée.)
la fonction d'agrégation $Agg(H'_{1t})$ is:

$$Agg(H'_{1t}) = [0.6, 0.2, 0.2] \begin{bmatrix} f(h_2^t) \\ f(h_3^t) \\ f(h_4^t) \end{bmatrix} \tag{5}$$

Et la fonction f sur tous les bords est:

$$f(x) = 2x \tag{6}$$

Maintenant, étant donné que

$$h_1^{t+1} = q(h_1^t, Agg(H'_{1t})) = W(h_1^t)^T + max\{Agg(H'_{1t}), 0\} \tag{7}$$

où $W = [1, 1]$, quelle est la valeur actualisée de h_1^{t+1} ?
(h_i^t et W sont des vecteurs de rangée.)

(2.3) Considérons le graphique G en question (2.2). Nous voulons la modifier pour représenter la peine "Je mange des pommes rouges" (4 mots tokens) comme un graphique entièrement connecté. Chaque sommet représente un mot token, et les bords représentent les relations entre les tokens. Combien de bords au total le graphique G contient-il ? Notez que les bords sont orientés et un bord bidirectionnel compte comme deux bords.

(2.4) au moyen des équations 1 et 3, Le mécanisme d'attention à tête unique du modèle de transformateur est équivalent à un cas particulier de GNN.

(2.5) Un domaine de recherche continu dans les modèles de transformateurs pour la NLP est le défi de l'apprentissage les dépendances à très long terme entre les entités dans une séquence. Compte tenu de ce lien avec les GNNs, pourquoi pensez-vous que cela pourrait être problématique ?

Answer 2. Q.2.1.

$$\text{Agg}\left(H'_{1t}\right) = \sum_{j/v_j \in N(v_i)} f_{i,j}(h_j^t)$$

Ainsi,

$$h_i^{t+1} = q\left(h_i^t, \sum_{j/v_j \in N(v_i)} f_{i,j}(h_j^t)\right)$$

Q.2.2.

$$\text{Agg}\left(H'_{1t}\right) = [0.6, 0.2, 0.2] \begin{bmatrix} [-2,2] \\ [0,-2] \\ [2,0] \end{bmatrix} = [-0.8, 0.8]$$

$$\begin{aligned} h_1^{t+1} &= W\left(h_1^t\right)^T + \max\left\{\text{Agg}\left(H'_{1t}\right), 0\right\} \\ &= [1,1][1,-1]^T + \max\left\{[-0.8, 0.8], 0\right\} \\ &= [0, 0.8] \end{aligned}$$

Q.2.3. On a un graphe connectés de 4 mots, on a donc 12 arrêtes.

Q.2.4. On a le graphe des mots est complètement connectés comme dans la question 2.3, ainsi $N(v_i) = S$, et $H'_{it} = \left\{h_j^t | j \in S\right\}$ et $f_{ij}(x) = x$

En posant,

$$Agg(H'_{it}) = (Q^t, K^t h_j^t, V^t h_j^t)_{j \in S}$$

Ainsi en posant,

$$\begin{aligned} q\left(h_i^t, \text{Agg}\left(H'_{it}\right)\right) &= \sum_{j \in S} \left(\text{sof tmax}_j \left(Agg(H'_{it})[j,0] h_i^t \cdot Agg(H'_{it})[j,1]\right) Agg(H'_{it})[j,2]\right) \\ &= \sum_{j \in S} \left(\text{sof tmax}_j \left(Q^l h_i^l \cdot K^l h_j^l\right) V^l h_j^l\right) \end{aligned}$$

on obtient donc le résultat recherché.

Q.2.5.

Pour les longues séquences, les matrices de poids comme K^t seront très grandes pour un graphes complètement connecté, ce qui peut créer un très grand temps d'entrainement. De plus la backprop risque de prendre beaucoup de temps as well et de mémoire aussi parce que pour les grapndes séquences, on aura un graphe connecté de taille immense !

Question 3 (6-2-6-6)**.** (**Optimization et Regularization**)

(3.1) La normalisation par lots (batch normalization), la normalisation des couches (layer normalization) et la normalisation des instances (instance normalization) impliquent le calcul de la moyenne $\boldsymbol{\mu}$ et la variance $\boldsymbol{\sigma^2}$ par rapport à différents sous-ensembles des dimensions du tenseur. Étant donné le tenseur 3D suivant, calculez les tenseurs de moyenne et de variance correspondants pour chaque technique de normalisation: $\boldsymbol{\mu}_{batch}$, $\boldsymbol{\mu}_{layer}$, $\boldsymbol{\mu}_{instance}$, $\boldsymbol{\sigma}^2_{batch}$, $\boldsymbol{\sigma}^2_{layer}$, and $\boldsymbol{\sigma}^2_{instance}$.

$$\left[\begin{bmatrix}4,3,2\\1,4,3\end{bmatrix}, \begin{bmatrix}3,4,1\\4,2,2\end{bmatrix}, \begin{bmatrix}3,2,3\\4,1,2\end{bmatrix}, \begin{bmatrix}1,1,3\\4,1,2\end{bmatrix}\right]$$

La taille de ce tenseur est de 4 x 2 x 3, ce qui correspond à la taille du lot, au nombre de canaux et au nombre de caractéristiques respectivement.

(3.2) Alors que BatchNorm est très commun dans les modèles de vision par ordinateur, Il est nettement surpassé par LayerNorm dans les tâches de traitement du langage naturel. Quels pourraient être quelques problèmes avec l'utilisation de BatchNorm dans NLP ? Vous pouvez considérer une mise en en œuvre naïve de BatchNorm et aussi illustrer votre point avec un exemple si vous le souhaitez.

(3.3) Considérez un problème de régression linéaire avec les données d'entrée $\boldsymbol{X} \in \mathbb{R}^{n \times d}$, poids $\boldsymbol{w} \in \mathbb{R}^{d \times 1}$ et objectifs $\boldsymbol{y} \in \mathbb{R}^{n \times 1}$. Supposons que l'exclusion est appliquée à l'entrée (avec une

probabilité $1-p$ de laisser tomber l'unité, c.-à-d. la régler à 0). Soit $\boldsymbol{R} \in \mathbb{R}^{n \times d}$ être le masque dropout tel que $\boldsymbol{R}_{ij} \sim \text{Bern}(p)$ est échantillonné i.i.d. de la distribution Bernoulli.
Pour une fonction de perte d'erreur au carré avec dropout, on a alors :

$$L(\boldsymbol{w}) = ||\boldsymbol{y} - (\boldsymbol{X} \odot \boldsymbol{R})\boldsymbol{w}||^2$$

Soit Γ être une matrice diagonale avec $\Gamma_{ii} = (\boldsymbol{X}^\top \boldsymbol{X})_{ii}^{1/2}$. Montrer que les *expectation (over $\boldsymbol{R}$)* de la fonction de perte peut être réécrit comme: $\mathbb{E}[L(\boldsymbol{w})] = ||\boldsymbol{y} - p\boldsymbol{X}\boldsymbol{w}||^2 + p(1-p)||\Gamma \boldsymbol{w}||^2$. *Indice: Notez que nous essayons de trouver l'attente sur un terme carré et d'utiliser* $\text{Var}(Z) = \mathbb{E}[Z^2] - \mathbb{E}[Z]^2$.

(3.4) Considérez un problème de régression linéaire avec un vecteur d-dimensionnel $\boldsymbol{x} \in \mathbb{R}^d$ comme entrée. et $y_i \in \mathbb{R}$ comme sortie. L'ensemble de données comprend n exemples de formation. $(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)$. Soit $\boldsymbol{X}$ être la $n \times d$ matrice de données formée en plaçant les vecteurs de caractéristiques sur les lignes de cette matrice c.-à-d,

$$\boldsymbol{X}^T = \left[\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\right]$$

Soit $\boldsymbol{y}$ être un vecteur de colonne avec des éléments $y_1, y_2 ... y_n$. Nous opérerons dans le cadre où $d > n$, c'est-à-dire qu'il y a plus de dimensions que d'échantillons. Ainsi, le système linéaire suivant est sous-limité/sous-déterminé:

$$\boldsymbol{X}\boldsymbol{w} = \boldsymbol{y} \tag{8}$$

Pour éviter le cas dégénéré, nous supposerons que $\boldsymbol{y}$ réside dans l'envergure de $\boldsymbol{X}$, c.-à-d. que ce système linéaire a au moins une solution.
Maintenant, rappelez-vous que le problème d'optimisation dans la régression des moindres carrés est le suivant :

$$\min_{\boldsymbol{w} \in \mathbf{R^d}} \quad f(\boldsymbol{w}) = \sum_{i=1}^{n} \underbrace{(y_i - \boldsymbol{w}^T \boldsymbol{x}_i)^2}_{\text{squared error on example } i} \tag{9}$$

Nous optimiserons (9) par descente en pente. Plus précisément, initialisons $\boldsymbol{w}^{(0)} = 0$. Et à plusieurs reprises faire un pas dans la direction de gradient négatif avec un pas-taille constant suffisamment petit η jusqu'à convergence.

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta \nabla f(\boldsymbol{w}^{(t)}) \tag{10}$$

Référons-nous à la solution trouvée par la descente en pente à la convergence comme $\boldsymbol{w}^{gd}$.
Prouver que la solution trouvée par descente en pente pour les moindres carrés est égale aux résultats de ce qui suit *différent* problème d'optimisation:

$$\boldsymbol{w}^{gd} = \arg\min_{\boldsymbol{w} \in \mathbf{R^d}} \quad ||\boldsymbol{w}||_2^2 \tag{11}$$

$$\text{such that} \quad \boldsymbol{X}\boldsymbol{w} = \boldsymbol{y} \tag{12}$$

Indice: Pour ce problème d'optimisation, vous devez travailler avec le Lagrangien $L(\boldsymbol{w}, \lambda)$, donnée par

$$L(\boldsymbol{w}, \lambda) = \|\boldsymbol{w}\|^2 + \lambda^\top (\boldsymbol{X}\boldsymbol{w} - \boldsymbol{y})$$

Obtenir $\boldsymbol{w}^*$ et λ^* à partir des conditions KKT ($\frac{\partial L}{\partial w} = 0$ and $\frac{\partial L}{\partial \lambda} = 0$) pour arriver au résultat.

Answer 3. Q.3.1.

$\mu_{batch} = \begin{bmatrix} 2.5 \\ 2.5 \end{bmatrix}$ and $\sigma^2_{batch} = \begin{bmatrix} 1.083 \\ 1.42 \end{bmatrix}$

$\mu_{layer} = \begin{bmatrix} 2.83 & 2.66 & 2.5 & 2.0 \end{bmatrix}$ and $\sigma^2_{layer} = \begin{bmatrix} 1.14 & 1.22 & 0.92 & 1.33 \end{bmatrix}$

$\mu_{instance} = \begin{bmatrix} 3.0 & 2.67 & 2.66 & 1.67 \\ 2.67 & 2.67 & 2.33 & 2.33 \end{bmatrix}$ and $\sigma^2_{instance} = \begin{bmatrix} 0.67 & 1.55 & 0.22 & 0.89 \\ 1.55 & 0.89 & 1.55 & 1.55 \end{bmatrix}$

Q3.2.

Q.3.3.

$$\begin{aligned} \mathbb{E}[L(w)] &= \mathbb{E}[\|\boldsymbol{y} - (\boldsymbol{X} \odot \boldsymbol{R})\boldsymbol{w}\|^2] \\ &= \|y\|^2 - 2 * \mathbb{E}(y^T(\boldsymbol{X} \odot \boldsymbol{R})\boldsymbol{w}) + \mathbb{E}(\boldsymbol{w}^T(\boldsymbol{X} \odot \boldsymbol{R})^T(\boldsymbol{X} \odot \boldsymbol{R})\boldsymbol{w}) \end{aligned}$$

et on a l'espérance est linéaire et que $\mathbb{E}[\boldsymbol{X} \odot \boldsymbol{R}][i,j] = \mathbb{E}[x_{i,j} * R_{i,j}] = p * x_{i,j}$

Et pour $i \not\!\!j$,

$$\begin{aligned} \mathbb{E}[(\boldsymbol{X} \odot \boldsymbol{R})^T(\boldsymbol{X} \odot \boldsymbol{R})][i,j] &= \mathbb{E}[\sum_k x_{k,i} R_{k,i} x_{k,j} R_{k,j} \\ &= \sum_k x_{k,i} x_{k,j} \mathbb{E}(R_{k,i} R_{k,j}) \text{ avec les } R_{i,j} \text{ qui sont i.i.d.} \\ &= p^2 \boldsymbol{X}^T \boldsymbol{X}[i,j] \end{aligned}$$

$$\begin{aligned} \mathbb{E}[(\boldsymbol{X} \odot \boldsymbol{R})^T(\boldsymbol{X} \odot \boldsymbol{R})][i,i] &= \mathbb{E}[\sum_k x^2_{k,i} R^2_{k,i}] \\ &= \sum_k x^2_{k,i} \mathbb{E}(R^2_{k,i}) \\ &= p \boldsymbol{X}^T \boldsymbol{X}[i,i] \end{aligned}$$

Donc

$$\mathbb{E}[(\boldsymbol{X} \odot \boldsymbol{R})^T(\boldsymbol{X} \odot \boldsymbol{R})] = p^2 \boldsymbol{X}^T \boldsymbol{X} + p(1-p)\mathbf{diag}(\left(\boldsymbol{X}^\top \boldsymbol{X}\right)_{ii})$$

Ainsi

$$\mathbb{E}[L(w)] = \|\boldsymbol{y}\|^2 - 2p\boldsymbol{y}^T\boldsymbol{X}\boldsymbol{w} + p^2\boldsymbol{w}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} + p(1-p)\boldsymbol{w}^T\mathbf{diag}(\left(\boldsymbol{X}^\top\boldsymbol{X}\right)_{ii})\boldsymbol{w}$$

Or,

$$\|\boldsymbol{y} - p\boldsymbol{X}\boldsymbol{w}\|^2 + p(1-p)\|\Gamma\boldsymbol{w}\|^2 = \|\boldsymbol{y}\|^2 - 2p\boldsymbol{y}^T\boldsymbol{X}\boldsymbol{w} + p^2\boldsymbol{w}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} + p(1-p)\boldsymbol{w}^T\mathbf{diag}(\left(\boldsymbol{X}^\top\boldsymbol{X}\right)_{ii})\boldsymbol{w}$$

Conclusion :

$$\mathbb{E}[L(\boldsymbol{w})] = \|\boldsymbol{y} - p\boldsymbol{X}\boldsymbol{w}\|^2 + p(1-p)\|\Gamma\boldsymbol{w}\|^2$$

Q 3.4.

Résoudre le problème (11)

$$\frac{\partial L}{\partial w} = 2\boldsymbol{w} + X^T\lambda$$

$$\frac{\partial L}{\partial \lambda} = X\boldsymbol{w} - y$$

Ainsi en utilisant les conditions KKT,

$$\begin{cases} 2\boldsymbol{w}^* + X^T\lambda^* &= 0 \\ X\boldsymbol{w}^* &= y \end{cases}$$

Et dans la realtion de la descente du gradient (10), en faisant tendre t vers $+\infty$, on a : $\nabla f\left(w^{(gd)}\right) = 0$

Et $\nabla f\left(w^{(gd)}\right) = -2 \cdot X^\top \cdot (y - X \cdot w^{(gd)})$ Ainsi,

$$X^\top y = X^\top X \cdot w^{(gd)}$$

En réutilisant le système obtenu par les conditions KKT,

$$X^T X(w^{gd} - w^*) = 0$$

Si $X^T X$ est positive donc

$$w^{gd} = w^*$$

Question 4 (20 pts)**.** (**Question de révision d'article**)

dans cette question, vous allez écrire **une page** révision de the vision transformer paper. Veuillez structurer votre examen dans les sections suivantes:

1. **Sommaire [5 pts]**:
   (a) De quoi parle cet article ?
   (b) Quelle est la principale contribution ?
   (c) Décrivez l'approche principale et les résultats. Seulement des faits, pas encore d'opinions.
2. **Forces [5 pts]**:
   (a) Y a-t-il un nouvel aperçu théorique ?
   (b) Ou un progrès empirique important ? Ont-ils résolu un problème permanent ?
   (c) Ou une bonne formulation pour un nouveau problème ?
   (d) Des résultats concrets (code, algorithme, etc.) ?
   (e) Les expériences sont-elles bien exécutées ?
   (f) Utile pour la communauté en général ?
3. **Faiblesses [5 pts]** :
   (a) Que peut-on faire de mieux ?
   (b) Des bases de référence manquantes ? Des ensembles de données manquants ?
   (c) Des choix de conception bizarres dans l'algorithme ne sont-ils pas bien expliqués ? Qualité de l'écriture ?
   (d) Est-ce qu'il y a suffisamment de nouveauté dans ce qu'ils proposent ? Variation mineure de travaux antérieurs ?
   (e) Pourquoi devrait-on s'en soucier ? Le problème est-il intéressant et important ?
4. **Reflets [5 pts]**:
   (a) Quel est le lien avec d'autres concepts que vous avez vus dans la classe ?
   (b) Quelles sont les prochaines orientations de recherche dans ce domaine ?
   (c) Quelles nouvelles idées (directement ou indirectement liées) ce document vous a-t-il donné ? Qu'est-ce que tu voudrais essayer ?

Answer 4. Report in English to avoid mistranslating key words

This paper introduces the use of Transfromers in the image processing tasks. This technique is usually used for NLP related tasks, which uses the embeddings of the tokens(words) of the sequences as input. In this paper, the parallelism is conserved with the NLP Transfromer, in fact the article uses almost the same Transformer. The difference being the input. **Their contribution** is the first layer that transforms the image to be adapted to be the input of a Transformer. Here, tthe image is cut into small patches that are flattened. To conserve the information of their placement, they aded a case to the embedding indicating the original position of the patch within the image. Only then that the image is fed to Transformer Encoder. **The obtained results,** are intersting, the model out perfroms the state of the art ResNet classic models, but also takes less time to pre-train. However, the model does not out perform the ResNet for small datasets like ImageNet.

(4.2) In fact, there is no new theoritical insight. The main idea was to split the image into small patches that would be the tokens of the Transformer. However, one must admit that the fact that this new approach outperforms the ResNet model is quite an empirical progress. This paper could actually be perceived as a first good formulation of the use of Transformers in Computer Vision. The pseudo code and algorithms were explained in good details, which would allow the reader to code it if he wishes to. And the graphs and experiences were given in good details, since the different types of the model were executed ("Base","Large", "Huge") on datasets with different scales from ImageNet to JFT. They even added the pretraining time which is one of the strong points of this new method. Actually The years that followed would know various contibutions of the same domain. The article was cited 8916 times according to Google Scholar, and that in the span of two years.

(4.3) The article praises itself on outperforming on very large scale datasets, however, this remains **unreproducible** for the largest dataset they used. The JFT is an internal Google dataset, so it would be hard to access if not working inside the google. The thing that I don't get the most is the reason behind using 16x16 image patches for Large and Huge models. This hyperparameter was never really discussed in this paper. Also, the code and the algorithm that is proposed is not revolutionnary since it just adds a new pre-processing layer that gives the input the required shape for a Transformer. Furthermore, this article major strength comes from the access they had (TPUs and private datasets). This article doesn't actually have a really big added value when we deal with medium size datasets.

(4.4) This article calls in the Transfromer configuration on a computer vision tasks, it somewhat reminds of the first assignement for example when we used MLP on images but we completly flatten them beforehands. Actually, the article does that as well in the Hybrid Architecture. So this technique of thinking of an image differently have been discussed in class. Also the base of the model used, meaning the MultiHeadTransfomer is the one we coded with GRUs in the practical code of the second assignement. Also the article briefly mentions how the Group normalization (which is a hybrid between Instance Normalization and Layer Normalization) improves the ResNet. **The next steps** in this domain would be to try and find new ways in image representation that does not rely on CNN, maybe some methods that might be more complex and more effecient on low-scale data. CI currently work on a min-cut problem that uses 3D graph manipulation. And I think that the idea of patching the image and **adding position embeddings might help me in my project.** In fact, I want to navigate the graph to link all the coupled weights (some weights are -1, some 1 , the rest is 0). 2D is obviousd with Steiner Trees, and I think labeling the trees with positional

mebeddings might help me link them in a 3D configuration.