INF8225

TP3

Hiver 2023

01

Dépelteau, Nicolas - 2083544

Soumis à Christopher Pal

Remis le 3 Avril 2023

Questions

1. Explain the differences between Vanilla RNN, GRU-RNN, and Transformers.

Un RNN vanille est un type de réseau de neurones récurrents qui traite les données séquentielles en prenant en compte l'ordre et les entrées précédentes. Cependant, il peut être difficile de paralléliser les opérations RNN et le traitement de longues séquences peut entraîner des gradients explosifs ou disparaissants. Les unités récurrentes à portes (GRU) sont une variante des RNN qui utilisent des mécanismes de portes pour retenir sélectivement l'information et sont capables d'apprendre des dépendances à long terme. Les transformateurs sont un autre type de réseau de neurones qui utilisent des mécanismes d'auto-attention pour des tâches telles que la compréhension du langage naturel, la génération de texte et la traduction automatique

2. Why is positional encoding necessary in Transformers and not in RNNs?

L'encodage de position est nécessaire dans les transformateurs car les entrées sont traitées simultanément, ce qui signifie que les transformateurs n'ont pas d'informations sur l'ordre des mots dans une phrase. L'encodage de position ajoute des informations sur l'ordre des mots dans une phrase pour que le modèle puisse comprendre la signification de la phrase. Dans les RNN, les phrases sont traitées mot par mot jusqu'à ce que la phrase soit complète. C'est ainsi que le modèle comprend quel mot vient après l'autre dans les RNN

3. Describe the preprocessing process. Detail how the initial dataset is processed before being fed to the translation models.

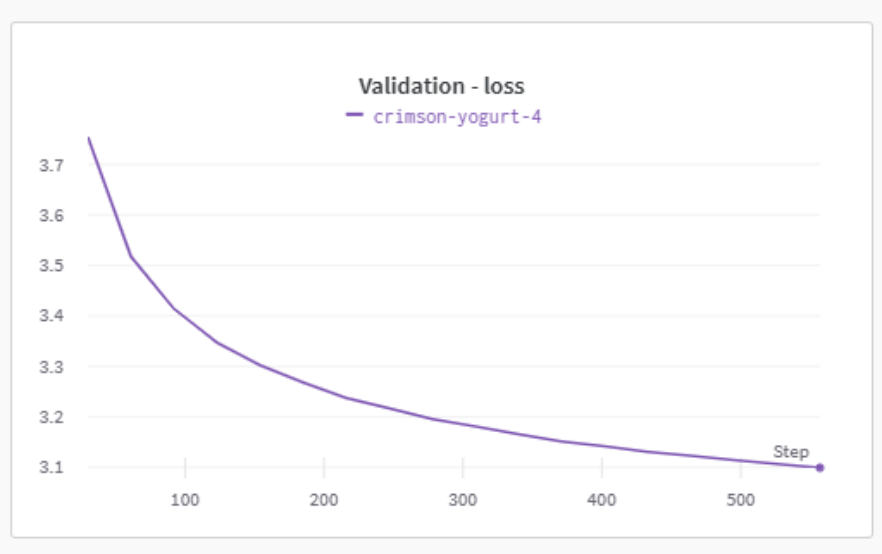
Le prétraitement des données pour la traduction automatique commence par la sélection d'un jeu de données approprié contenant des paires de phrases dans les langues source et cible. Les phrases sont ensuite tokenisées en subdivisant chaque phrase en une liste de tokens. Le vocabulaire utilisé pour la tokenisation peut inclure des tokens spéciaux tels que "unk" pour les mots inconnus, "pad" pour le rembourrage, "bos" pour le début de phrase et "eos" pour la fin de la phrase. Ces tokens spéciaux sont utilisés pour gérer les cas où les phrases ont des longueurs différentes ou contiennent des mots qui ne sont pas présents dans le vocabulaire. Ensuite, on peut injecter cette liste de token à nos modèles

Experimentation

RNN

Au cours de l'entraînement du modèle, j'ai remarqué que la perte d'entraînement (training loss) est devenue constante assez rapidement, ce qui indique que le modèle a atteint un niveau de performance stable sur les données d'entraînement. Cependant, la perte de validation (validation loss) était de 3.1, ce qui suggère que le modèle ne généralise pas aussi bien sur les données de validation. Cela pourrait être dû à un surapprentissage sur les données d'entraînement ou à un manque de capacité du modèle à capturer les dépendances à long terme dans les données.

Graphique 1. Courbe de perte sur l'ensemble validation du RNN



Graphique 2. Courbe de perte sur l'ensemble entaînement du RNN

Transformer

Dans ma deuxième expérimentation, j'ai évalué les performances de réseaux de transformateurs de différentes tailles pour résoudre une tâche de traitement du langage naturel. J'ai entraîné trois modèles différents: un petit, un moyen et un énorme. J'ai constaté que le modèle énorme avait les meilleures performances en termes de précision et de capacité à capturer les dépendances à long terme dans les données. Cependant, il était également plus lent à entraîner et nécessitait plus de ressources informatiques. Le modèle moyen avait des performances légèrement inférieures à celles du modèle énorme, mais était plus rapide à entraîner et nécessitait moins de ressources. Le petit modèle avait les performances les plus faibles, mais était le plus rapide à entraîner et nécessitait le moins de ressources.

L'entraînement a été effectué à l'aide d'une carte graphique NVIDIA RTX 2060 sur mon ordinateur portable. Cependant, le temps d'entraînement était de 15 à 30 minutes par époque, ce qui était assez long pour que je n'aie le temps que de tester trois modèles.

En résumé, trois configurations de base ont été testées: petite, moyenne et énorme. Les résultats montrent qu'à mesure que la taille de la configuration augmentait, les pertes d'apprentissage et de validation diminuent. La petite configuration avait une perte d'apprentissage de 1.475 et une perte de validation de 1.484. La configuration moyenne avait une perte d'apprentissage de 1.072 et une perte de validation de 1.247. La configuration énorme avait les pertes les plus faibles avec une perte d'apprentissage de 0.703 et une perte de validation de 0.995. Ces résultats suggèrent que des configurations plus grandes peuvent conduire à de meilleures performances dans cette expérience particulière.

Graphique 3. Courbe de perte sur l'ensemble validation des transformeurs

Graphique 4. Courbe de perte sur l'ensemble d'entraînement des transformeurs

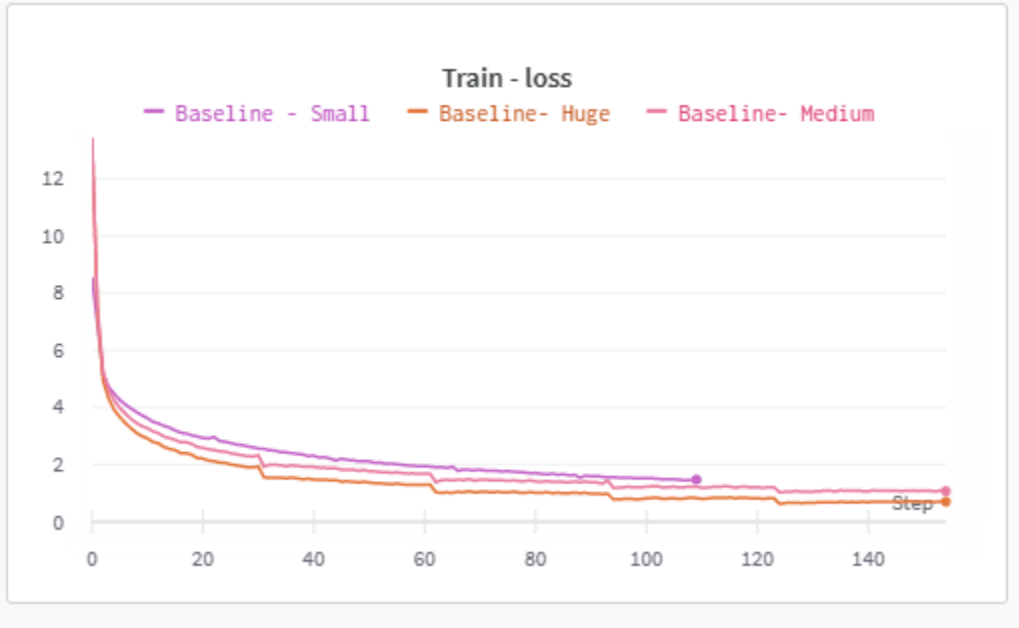


Tableau 1. Configuration du Baseline - Small

Paramètres	valeur
Batch size	128
Clip	5
dim_embedding	40
dim_hidden	60
dropout	0.001
epochs	5
learning rate	0.001
max_sequence_length	10
min_token_freq	20
n_heads	4
n_layers	1
n_tokens_src	2175
n_tokens_tgt	2635
optimizer	ADAM

Tableau 2. Configuration du Baseline - Medium

Paramètres	valeur
Batch size	128
Clip	5
dim_embedding	196
dim_hidden	256
dropout	0.001
epochs	5
learning rate	0.001
max_sequence_length	60
min_token_freq	20
n_heads	4
n_layers	3
n_tokens_src	11709
n_tokens_tgt	17645
optimizer	ADAM

Tableau 3. Configuration du Baseline - Huge

Paramètres	valeur
Batch size	128
Clip	5
dim_embedding	200
dim_hidden	256
dropout	0.001
epochs	5
learning rate	0.001
max_sequence_length	60
min_token_freq	20
n_heads	8
n_layers	3
n_tokens_src	11709
n_tokens_tgt	17645
optimizer	ADAM

Tableau 4. Perte pour chacun des modèles

Taille	Perte sur l'ensemble de validation	Perte sur l'ensemble d'entraînement
Small	1.484	1.475
Medium	1.072	1.247
Huge	0.995	0.703

Le tableau 4 présente les pertes pour chacun des modèles de différentes tailles: petit, moyen et énorme. Les résultats montrent que la perte sur l'ensemble de validation et la perte sur l'ensemble d'entraînement diminuent à mesure que la taille du modèle augmente. Le modèle petit a une perte de 1.484 sur l'ensemble de validation et une perte de 1.475 sur l'ensemble d'entraînement. Le modèle moyen a une perte de 1.072 sur l'ensemble de validation et une perte de 1.247 sur l'ensemble d'entraînement. Le modèle énorme a les pertes les plus faibles avec une perte de 0.995 sur l'ensemble de validation et une perte de 0.703 sur l'ensemble d'entraînement.
n-grammes, c'est-à-dire que la somme des poids est égale à 1 et chaque poids est positif. Le choix le plus typique, celui recommandé dans l'article original, est d'utiliser des poids égaux pour les n-grammes de 1 à 4.