Seminaire Turing

g.dhimoila

February 2023

Abstract

Ce rapport présente le projet fait dans le cadre du Séminaire Turing. On y retrouve une implémentation de l'architecture des transformer, des toy model pour voir apparaître des induction head, et du fine tuning de gpt-2 small pour voir l'évolution du circuit des COI.

1 Introduction

Le but à lon terme de ce projet est d'étudier l'exaptation dans les transformer et l'effet du fine tuning sur des circuits ayant un rôle précis. L'exaptation est le phénomène qui conduit, par exemple, les plumes (initialement dédiée à la régulation de la chaleur) à changer de rôle pour permettre le vol, un objectif tout nouveau mais auquel elles sont particulièrement bien adaptée.

J'ai choisit le circuit servant à la gestion des COI, mis en évidance dans $Interpretability\ In\ The\ Wild: \ A\ Circuit\ for\ IOI\ in\ GPT-2\ small$.

2 Transformers et têtes d'induction.

Le but ici est de mettre en évidence un circuit simple, composé de deux têtes :

- la première fournit le token précédent au token actuel (PTH, pour previous token head),
- la deuxième est la tête d'induction. Elle regarde dans le début de la phrase les tokens qui sont proche du token courant, et grâce au travail de la PTH, ce sont exactement les tokens qui apparaîssent après le token courant dans un motif.

Ce circuit permet donc de détecter et de compléter des motifs récurrents dans un texte. Typiquement, dans la langue anglaise, "pot" est le plus souvent suivit de "entiel", mais dans un texte mentionnant souvent Harry Potter, "Pot" sera le plus souvent suivit de "ter".

L'éxpérience que j'ai imaginée pour faire apparaître facilement des induction head est d'entraîner un transformer à prédire le next token sur un ensemble de séquences aléatoires périodiques. Il s'agit d'une tâche très simple, dont la seule

composante déterministe et utile est la répétition de motifs. Une méthode simple et efficace pour le transformer est alors de développer une induction head. On peut donc s'attendre que le simplicity prior de la gradient descente nous en fasse apparaître de manière plutôt consistante. En pratique, deux couches, une tête par couche, suffisent à en voir apparaître.

Au départ, j'ai fait des expériences avec des transformer contenant un encodeur et un décodeur, en utilisant l'implémentation que l'on avait réalisé lors du ML4G. Cependant, cette architecture est spécifiquement bonne pour les taches qui se rapportent à de la traduction, et ce n'était pas du tout adapté lors de mon expérience, puisque l'encodeur a accès à la phrase entière, donc une attention centrée sur le token actuel, puis tous le reste qui ne fait rien, suffit amplement ... J'ai donc essayé de transformer le problème de plusieurs manières différentes, pour que ce soit un problème de traduction et que le seul moyen de le résoudre soit avec des induction head - en introduisant du bruit spatial, par exemple, pour que l'encodeur arrête d'apprendre par coeur les positions intéressantes. Ce fut un lamentable échec. J'ai donc commencé à questionner mon implémentation, puisqu'elle n'arrivait pas à faire apparaître un circuit aussi simple et adapté à une tâche aussi simple.

Après quelques recherches, et un joker Charbel, il s'est avéré que l'architecture de GPT n'était pas un encodeur décodeur, mais un simple décodeur. J'ai donc apporté les modifications nécessaires à mon implémentation pour supprimer l'encodeur. Ca a tout de suite, bizarrement, beaucoup mieux marché ! On voit presque systématiquement l'apparition d'une previous token head, et d'une induction head. Cependant, les résultats sont étranges. En effet, dans *A Mathematical Framework for Transformer Circuits*, on peut voir les diagrammes d'attention des têtes de ce circuit, et les leurs contiennent des belles diagonales bien complètes, alors que dans les miens, les diagonales se coupent après environ le tier de la séquence...

J'ai longtemps cherché une explication de pourquoi ce "comportement" apparraissant, et de manière consistante qui plus est. Voici mon raisonnement : la matrice de positionnal embedding est apprise lors de l'entrainement. La tête d'induction n'a pas besoin de la position. La previous token head, elle, a besoin de la position. Cependant, tous les motifs apparaissent dans la première moitié de la phrase. De plus, la seule et unique composante du problème à apprendre est de trouver ces motifs. A quoi bon, donc, apprendre à coder la position des tokens trop avancés ? Ou à quoi bon apprendre à utiliser ces tokens, alors que les précédents contiennent déjà toute l'information que l'on recherche ?

J'ai donc supposé que la descente de gradient a eu la flemme d'apprendre à coder la position passé une certain point dans les séquences. Pour vérifier cette hypothèses, rien de plus simple : regardons la matrice du positionnal embedding ! Ah... C'est... Euh... D'accord, faisons comme ça.

Comme le problème est très simple, et que la descente de gradient punit la complexité de la solution (weight decay), je suppose que seul un tout petit sous espace a été nécessaire pour coder la position, et le reste n'étant pas utilisé, est aléatoire.

3 Fine Tuning de gpt-2

Il ne reste plus qu'à trouver une tâche sur laquelle fine tune gpt-2, et pour laquelle on peut raisonnablement s'attendre à une modification des têtes faisant partie du circuit de détection des COI.

Ce circuit comporte une première phase qui sert à "compter" le nombre d'occurrence des mots, affin de ne retenir que le nom le moins courrant, qui sera alors sûrement le COI. Nous pouvons essayer d'exploiter cette première phase en demandant à gpt-2 d'extraire le set des tokens apparaissant dans une séquence aléatoire. Cependant un problème avec cette tâche est que si il extrait bel et bien le set mais pas dans l'ordre que l'on voulait, on détectera une erreur alors qu'il n'y en a pas. Une autre tâche pourraît être de simplement lui demander quel token est apparut le plus souvent. Cette tâche est bien plus simple à mettre en oeuvre, et c'est celle que j'ai retenu.

Je n'ai malheureusement pas réussi à aboutir. En effet, lorsque j'essaye de regarder les diagrammes d'attention des têtes indiquées par l'article, il n'y a rien d'ontéressant, tous les diagrammes sont à quelque chose près les mêmes, et les têtes ne semblent pas faire quoi que ce soit qui soit en rapport avec le rôle que l'article leur prête (par exemple, pour les previous token head, on s'attendrait à voir une diagonale décalée de un). On ne voit qu'un expèce de dégradé.

Après le fine tuning, les diagrammes semblent se discerner les uns des autres, mais on n'y voit rien de particulièrement intéressant.

4 Continuation

Pour le futur de ce projet, il me reste donc à comprendre pourquoi j'ai que des dégradés tout moche dans les diagrammes d'attention, comprendre les éventuelles autres techniques de détection du rôle des têtes, parce que les diagrammes d'attention c'est bien joli, mais c'est pas très rigoureux de juste regarder et voir si on a une belle diagonale là où on s'attend à en voire une...

Si jamais j'arrive à avoir des trucs assez propre, je pourrais éventuellement fine tune sur des problèmes plus concrets : courts poèmes et longs articles : pour les poèmes on s'attend à ce que les têtes d'induction s'effacent un peu, puisqu'elles seront moins utiles, et pour les articles en revanche, comme ils parlent souvent de trucs spécifiques qui reviennent souvent, on s'attend à ce qu'elles soient beaucoup plus utiles et soient renforcées. Si ça marche, on verrait l'impact du fine tuning de manière beaucoup plus intéressante que sur des toy tasks.