TensorFlow – un système d'apprentissage automatique pensé par Google

Par Guillaume et Sylvain Peyronnet

Domaine :	**Recherche**	**Référencement**
Niveau :	Pour tous	**Avancé**

Google a dernièrement annoncé RankBrain, son algorithme d'apprentissage automatique et d'intelligence artificielle (dont nous avons parlé le mois dernier) et TensorFlow, une bibliothèque logicielle autour de fonctionnalités de deep learning. Il nous semblait donc intéressant de décrire de façon plus approfondie ce nouvel outil disponible en Open Source, tout en expliquant de façon claire les notions de machine learning et deep learning, si souvent utilisées par Google pour bon nombre de ses produits.

Le mois dernier nous avons évoqué dans cette même lettre l'algorithme RankBrain annoncé par Google. Dans l'article, nous sommes restés à un niveau d'abstraction assez haut, pour éviter des détails techniques compliqués. Mais il pouvait être intéressant de parler plus en profondeur de TensorFlow, la bibliothèque de calcul au cœur de RankBrain, dont la disponibilité Open Source a été annoncée il y a peu par Google (http://www.abondance.com/actualites/20151110-15779-tensorflow-quand-google-devoile-ses-algorithmes-dapprentissage-automatique-en-open-source.html). Vous n'y couperez donc pas ce mois-ci. :-)

TensorFlow en quelques lignes

L'annonce a été très importante pour la communauté de l'intelligence artificielle : Google a rendu Open Source sa bibliothèque TensorFlow (disponible sur le site [1]). TensorFlow n'est pas un programme indépendant, ce n'est pas non plus un algorithme. Il s'agit d'une bibliothèque logicielle proposant un grand nombre de fonctionnalités différentes, toutes tournées autour du calcul sur des graphes de tenseurs (nous en parlons plus loin dans cet article). Une des applications de ce type de calcul se trouve dans le domaine de l'apprentissage automatique (*machine learning* en anglais), et Google s'en sert pour cela dans de très nombreux produits.

TensorFlow ne sort pas de nulle part : il s'agit de la seconde génération des outils de *deep learning* (voir plus loin) développés par Google. La première génération s'appelait **DistBelief** (voir [2]) et souffrait de nombreux problèmes, principalement techniques. Pour être tout à fait correct, les programmes écrits avec DistBelief ne pouvait pas être mis en

production facilement, alors que TensorFlow est 100% compatible avec les infrastructures internes de Google.

Machine learning et deep learning

Nous l'avons évoqué plus haut : TensorFlow a comme application principale le *machine learning*. Commençons donc par voir ce qu'est le *machine learning*, ainsi que le *deep learning* dont on parle tant en ce moment.

Le *machine learning* est une discipline fille de l'informatique et des statistiques. Son objectif est d'apprendre la structuration des données de manière à aider, voire à prendre complètement, les décisions sur ces données. Il existe trois grandes familles de méthodes de *machine learning* :

- L'apprentissage supervisé. Il s'agit ici d'apprendre des données qui ont été qualifiées à l'avance, généralement par des êtres humains. On va par exemple faire apprendre « à la machine » ce qui est du spam dans une boite mail, ou ce qui est une image pornographique, etc. Cette approche est très efficace, mais nécessite un travail très lourd en amont.

- L'apprentissage non supervisé. Dans cette approche, on vise à découvrir des motifs inconnus dans les données. L'exemple typique est celui du clustering opéré en segmentation clientèle dans le marketing. On en sait pas quels sont les groupes de clients qui partagent certaines caractéristiques, et on va utiliser l'apprentissage non supervisé pour les faire émerger. Très peu de problèmes peuvent être résolus efficacement avec ce type de méthode.

- L'apprentissage par renforcement. L'idée est ici d'avoir une boucle de rétroaction permanente. On va par exemple avoir un algorithme qui va apprendre à jouer au poker en jouant, avec un renforcement des stratégies selon qu'elles sont gagnantes ou perdantes. L'intuition derrière l'apprentissage par renforcement est celle de l'être humain qui apprend naturellement de cette manière pendant ses premières années. Les méthodes de bandits manchots pour l'optimisation (par exemple en marketing) sont typiquement dans ce schéma de pensée.

Pour la plupart des industriels qui utilisent le machine learning, l'objectif est de tirer partie d'une immense masse de données (comme par exemple un index contenant des milliards de pages web) à partir d'algorithmes les plus simples possible, qui ne nécessitent pas une intervention humaine lourde, et qui vont avoir la meilleure qualité de résultat possible.

Actuellement, les chercheurs et les industriels ont le sentiment que le *deep learning* peut remplir cet objectif. Dans les années 80, la recherche sur les réseaux de neurones artificiels était très dynamique (voir l'article du mois dernier pour comprendre ce qu'est un réseau de neurones). Puis, l'informatique matérielle n'étant pas à la hauteur pour faire tourner ces réseaux de neurones, la mode est passée. Et elle revient en force car maintenant,

algorithmes et matériel informatique sont en phase, grâce aux GPU et aux processeurs *multi* voire *many cores* (une dizaine de cœurs dans le premier cas, des centaines dans le second). Le *deep learning* a plusieurs avantages : pour créer un réseau profond, on va embriquer des modules simples, et c'est le comportement de l'ensemble qui va réussir des taches complexes. Par ailleurs, on peut faire tous les types de machine learning avec des réseaux de neurones profonds (supervisé, par renforcement et non supervisé).

Le deep learning est inspiré du principe de fonctionnement d'un cerveau : il n'y a pas un bloc de masse cérébrale qui fait tout le traitement d'une tache, mais au contraire, plusieurs blocs qui vont gérer plusieurs niveaux d'abstraction.

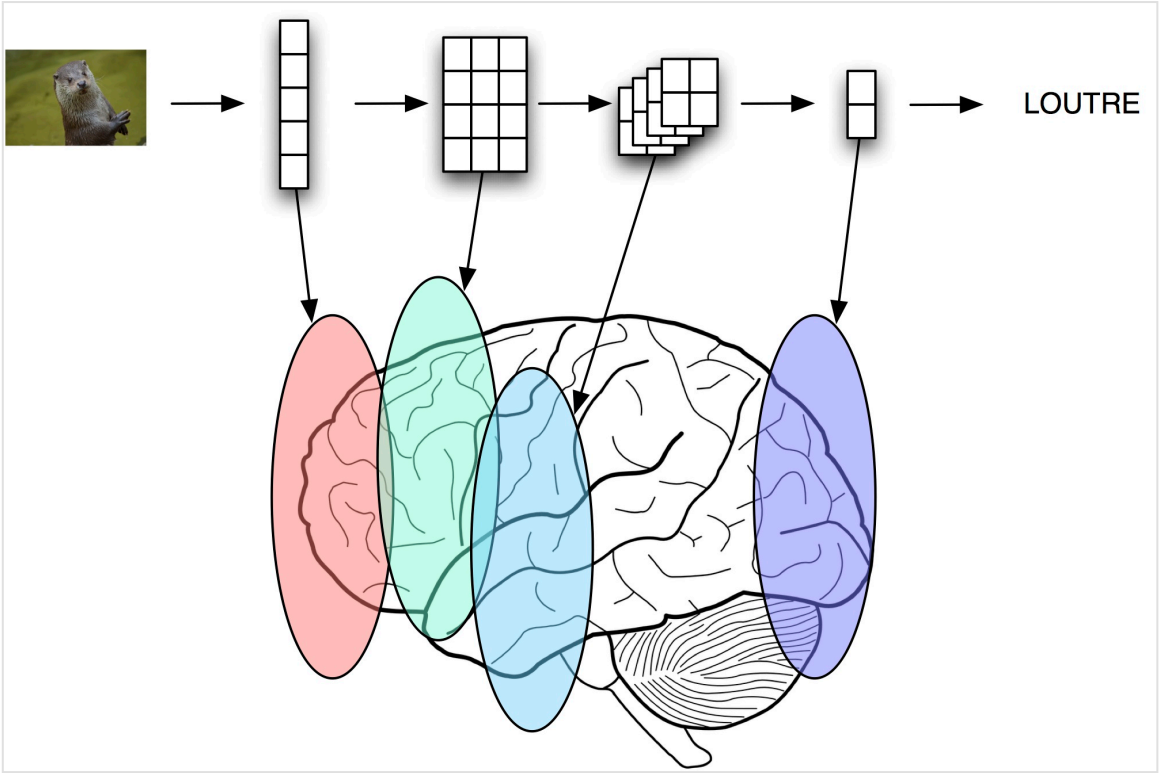

Fig. 1. Réseau profond fonctionnant grâce aux réseaux de neurones.

On voit sur la figure 1 un réseau profond avec plusieurs couches, qui permet de déterminer si un animal est une loutre ou non. Chaque couche va correspondre à un niveau d'abstraction, et intuitivement à un bloc de neurones dans un cerveau humain (l'attribution des couches au partie du cerveau est à titre d'exemple, au hasard). Lorsqu'on va fournir une photo au réseau profond, la première couche va par exemple regarder les pixels de la photo, la deuxième les blocs de pixels, la troisième la forme de l'animal, etc.

Une couche est un ensemble de neurones artificiels (voir l'article du mois dernier) qui communiquent entre eux, on peut donc représenter leur topologie par un vecteur (couche 1), une matrice (couche 2 ici) ou plusieurs matrices reliées entre elles (couche 3), etc.

Pour faire apprendre des connaissances à un réseau de neurones, on va procéder de manière itérative : on prends un exemple (une photo de loutre), on fournit l'exemple au réseau de neurones, qui va prendre une conclusion (par exemple indiquer qu'il n'y a pas

de loutres sur la photo). On va ensuite modifier les connexions entre neurones pour améliorer le résultat (si le réseau s'est trompé pour que cela ne soit plus le cas, si il est correct pour décider plus vite, etc.).

D'après Jeff Dean (voir sa bio [3]), tout ce qu'un humain peut faire en moins de 0,1 seconde est faisable par un réseau profond à 10 couches.

Nous allons maintenant pouvoir aborder TensorFlow, un ensemble de briques logicielles qui permet de réaliser ces fameux réseaux profonds.

TensorFlow : une bibliothèque de calcul sur les graphes de tenseurs

Un tenseur n'est pas quelque chose de mystérieux : il s'agit simplement d'un tableau à plusieurs dimensions. Par exemple, si la dimension est 1, c'est un vecteur, si c'est 2 c'est une matrice, si c'est 3 c'est un objet un peu plus étrange, qui peut par exemple correspondre à la modification d'une matrice en fonction du temps (le temps est alors la troisième dimension), ou encore à un encodage d'une photo en RGB (la troisième dimension est alors l'espace des couleurs R, G et B).

Un graphe (ou réseaux) de tenseurs est un graphe de flots entre tenseurs. Intuitivement, les nœuds de ce graphe sont des tenseurs et les arcs entre les nœuds vont être des opérations mathématiques effectuées sur les tenseurs.

Un réseau de neurones profond peut être encodé grâce aux graphes de tenseurs. L'avantage de cette approche dans TensorFlow est qu'elle permet de paralléliser les calculs (sur des GPUs, sur les cœurs des CPU), mais aussi de splitter le processus complet sur une plateforme hétérogène. Cela a un impact sur les petites implémentations : vous pouvez faire tourner TensorFlow pour profiter de tous les processeurs de votre machine en même temps (GPU, CPU, cartes addtionnelles, etc.). Ce parallélisme se fait de deux manières : sur le modèle (toutes les couches ne tournent pas sur les mêmes machines) et sur les données (une image va être découpée en plusieurs sous-images traitées séparément).

Jeff Dean (dans une conférence datant de 2014) évoquait ainsi un de leur modèle d'apprentissage qui tournait sur 144 machines totalisant environ 2 300 cœurs de calcul.

Il est important de noter que TensorFlow peut avoir d'autres utilisations que le *machine learning*, même si nous n'en parlerons pas ici.

A quoi cela sert-il ?

Le nombre d'applications potentielles de ce type d'outil est réellement très important. En voici quelques-unes venues directement de chez nos amis de Google.

- Reconnaissance de la parole

Il s'agit d'une application qui montre que Google travaille sur le sujet depuis longtemps puisqu'elle a été mise en production en 2012 dans la release Jellybean d'Android. Avec un entrainement de moins de 5 jours sur un cluster de 800 machines de calcul, ils ont réussi à créer un modèle acoustique pour la reconnaissance de la langue qui a diminué de 30% le taux d'erreur de reconnaissance (en langue anglaise).

- Reconnaissance d'objets dans les images

Il s'agit de loin de l'application qui a le plus fait parler d'elle. Les résultats sont édifiants.

Fig. 2. Reconnaissance d'objets dans une image.

Google a ainsi la capacité à reconnaître des fleurs différentes, mais aussi à généraliser à partir de caractéristiques moins claires. Par exemple, dans la figure 3, le réseau de neurones reconnaît qu'on parle de repas, alors qu'il y a peu de caractéristiques communes entre les photos.

Fig. 3. Reconnaissance d'images autour du « repas ».

Ceci étant, l'algorithme mérite parfois le bonnet d'âne (fig. 4)...

Fig. 4. Ooops...

Les résultats ont l'air (et ils sont) impressionants et l'algorithme de classification est déployé dans l'algorithme de recherche de photos de Google+. On notera que les premiers travaux sur ce sujet sont dûs à Yann Lecun, qui est directeur du laboratoire d'intelligence artificielle chez Facebook. L'approche standard était un réseau à 7 couches qui avait un taux d'erreur de 16%, tandis que Google a présenté en 2014 un réseau à 24 couches avec un taux d'erreur de presque 7%.

- Skipgram texte model.

Voici un nom qui pourrait faire peur, mais on ne parle de rien d'autre que de la base de RankBrain. L'objectif est ici d'apprendre des relations rares dans l'immense masse des données. Un exemple de relation rare ? C'est assez simple, les mots « Sylvain » et « Guillaume » sont en relation rare car on rencontre quelques phrases, dans le corpus documentaire du web, du type « je suis allé à l'apéro SEO et j'ai rencontré les frères Guillaume et Sylvain Peyronnet ».

Blague à part, on peut apprendre des relations de type co-occurrences grâce à un réseau de neurones (voir l'article [4]). Jeff Dean donne l'exemple des mots « bronx », « manhattan » et « brooklyn » qui sont « nearby » (pas loin) de « new york » selon le réseau de neurones. On ne va pas trop rien rajouter de plus ici car le sujet à déjà été abordé en détail dans notre précédent article. On notera simplement que les propriétés de l'outil de reconnaissance sont intéressantes, avec par exemple toutes les conjugaisons d'un verbe qui sont automatiquement regroupées (auparavant, on procédait par lemmatisation, perdant ainsi une partie de l'information). Ce modèle s'étend aux paragraphes et aux textes, comme vue dans la lettre Recherche et Référencement du mois d'octobre 2015.

On trouve d'autres applications chez Google : la traduction automatique et la description textuelle d'images. Cette dernière est d'ailleurs largement perfectible car Google voit, par exemple, un skateboardeur là où un humain voit un cerf-volant. Sur certaines autres images, la description est techniquement bonne mais pas très utile (« un homme tient une raquette sur un terrain de tennis », tandis que la description par un humain est « un tennisman est en train de servir »).

On voit cependant un très grand potentiel dans l'utilisation du deep learning, rendu possible grâce aux progrès techniques et à l'émergence de bibliothèque logicielle comme TensorFlow.

Est-ce que TensorFlow est la seule bibliothèque pour faire du deep learning efficacement ?

Soyons clair, encore une fois Google est loin d'être le premier sur la brêche. C'est Facebook qui avait en premier cru au *deep learning* en embauchant Yann Lecun (même si Google a depuis recruté Geoffrey Hinton, un des autres inventeurs du domaine), et il existe depuis quelques temps de nombreuses bibliothèques équivalentes à TensorFlow qui permettent de faire du machine learning, en particulier du deep learning, sur GPU, CPU, etc.

Les deux plus connues sont Theano [5] et Torch [6]. Si on commence par les facteurs différenciants, on peut dire que TensorFlow voit plus loin que le *machine learning* en offrant des primitives de travail sur les tenseurs indépendamment du domaine d'applications et également que TensorFlow a toutes les chances d'être plus robuste car pensé pour la mise en production. Ceci étant, il est encore un peu tôt pour savoir si la bibliothèque remportera l'adhésion de tous. Tout d'abord les perfomances ne semblent pas être meilleures (ni moins bonnes) que celles des autres bibliothèques comme par exemple Theano. Par ailleurs, il existe déjà de nombreux outils à coté de Theano et Torch pour simplifier leur utilisation, ce que l'on appelle un wrapper, et qui est une surcouche logicielle permettant de s'abstraire des difficultés techniques pour écrire simplement du code efficace. C'est par exemple le cas de Keras, qui a été récemment porté vers TensorFlow par ailleurs.

Au final, TensorFlow bénéficie d'un design bien pensé et de la force de frappe de Google, mais nous nous garderons bien de prédire qui gagnera la bataille des outils de *deep learning Open Source*.

Conclusion

Même si il est de bon ton de faire le snob quand Google propose du code, on doit bien dire que TensorFlow est une belle bibliothèque bien réalisée, qui a su montrer ses capacités sur des applications diverses. Rajoutons également qu'il y a fort à parier que Google utilise déjà en interne des évolutions substantielles à TensorFlow, et donc qu'il n'y avait pas grand risque à rendre tout cela public.

Maintenant, à vous de vous remonter vos manches, et à prendre en main la bête, notamment grâce aux liens ci-dessous, et bon courage !

Références

[1] https://www.tensorflow.org/

[2] http://research.google.com/pubs/pub40565.html

[3] https://en.wikipedia.org/wiki/Jeff_Dean_%28computer_scientist%29

[4] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781. http://arxiv.org/abs/1301.3781

[5] http://deeplearning.net/software/theano/

[6] http://torch.ch/

Guillaume Peyronnet *est gérant de Nalrem Médias.* **Sylvain Peyronnet** *est co-fondateur et responsable des ix-labs, un laboratoire de recherche privé. Ensemble, ils font des formations, pour en savoir plus : http://www.peyronnet.eu/blog/*