IA 323: Computer Vision **(20/07/2020)**

Projet #2 : Uncertainty in Deep Learning

Enseignant: Gianni Franchi *Élève:* Julien Guégan

Lorsque les classifieurs d'apprentissage automatique sont utilisés dans des tâches du monde réel, ils ont tendance à échouer lorsque les distributions d'apprentissage et de test diffèrent. Pire, ces classificateurs fournissent des prédictions de confiance élevé tout en étant complètement incorrectes. Ce type de classifieurs qui n'indiquent pas un score de confiance pertinent peuvent provoquer de graves accidents selon l'application qu'on en fait. Par exemple, un modèle de diagnostic médical pourrait classifier avec une grande confiance qu'une intervention chirurgicale est facile à effectuer par un être humain alors qu'en vérité il doit signaler l'intervention comme difficile à opérer, ou encore une voiture autonome pourrait mal identifier un piéton sur la route et accélérer en étant sûr de sa décision.

Ces prédictions à confiance élevée sont la plupart du temps produites par la fonction *softmax* du réseau de neurone :

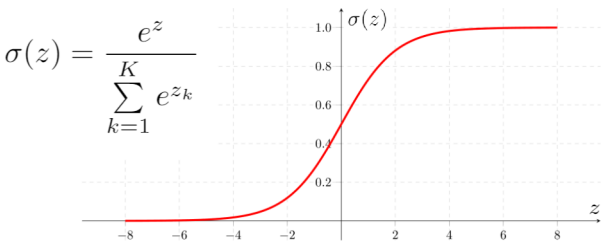

Figure 1: Fonction softmax.

En effet, les probabilités calculées par le modèle sortent du softmax qui utilise la fonction exponentielle. D'une part, des ajouts mineurs en entrées du softmax (les logits), peuvent entraîner des changements substantiels dans la distribution des sorties. Et d'autre part, étant donné que la fonction softmax est une approximation de la fonction d'indicatrice, il est quasi-impossible d'avoir une distribution uniforme produite pour des exemples hors distribution. C'est-à-dire que même en utilisant un bruit gaussien en entrée d'un classificateur d'images MNIST, on obtiendrait des confiances de prédiction de plus de 90%.

1 Calibration de modèle

On appelle ce type de problème sur la confiance la calibration des réseaux de neurone. Par exemple, si on a 80% de confiance pour 100 prédictions, alors si le modèle est calibré on s'attend à ce que 80% des prédictions soient correctes. Les diagrammes de fiabilité sont une représentation visuelle de la calibration de modèle. Ils affichent la précision en fonction de la confiance. Si le modèle est parfaitement calibré, il doit affiché la fonction identité. Dans la figure ci-dessous, le modèle de gauche est bien calibré alors que celui à droite ne l'est pas.

Les diagrammes de fiabilité sont une représentation visuelle de la calibration de modèle. Ils affichent la précision en fonction de la confiance. Si le modèle est parfaitement calibré, il doit affiché la fonction identité. Et on note dans la figure 2 ci-dessous que les architectures les plus modernes et les plus performantes sont souvent les moins bien calibrés avec ci-dessous le modèle LeNet (1998) à gauche est bien calibré alors que ResNet(2016) à droite ne l'est pas.

Pour mesurer cette erreur de calibration, on utilise l'erreur attendue de calibration (ECE) qui est la différence en espérance entre la confiance et la précision, i.e. :

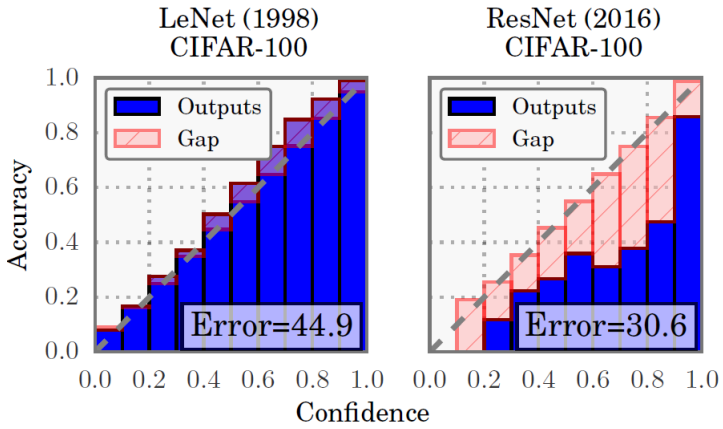

Figure 2: Diagrammes de fiabilité de LeNet(1998) et ResNet(2016) [1].

$$\text{ECE}^{1} = \sum_{m=1}^{M} \frac{|B_m|}{n} \big|\text{accuracy}(B_m) - \text{confidence}(B_m)\big|$$

Comme pour les diagrammes de fiabilité, ECE groupe les prédictions en M intervalles également espacés et calcule une moyenne pondéré des différences précision/confiance. Avec B_m l'ensemble des données dont la confiance tombe dans l'intervalle $I_m = (\frac{m-1}{M}, \frac{m}{M}]$ et n le nombre d'échantillon.

2 Temperature Scaling

Une méthode de calibration efficace et simple à mettre en oeuvre est la technique de *temperature scaling* qui consiste à remettre à l'échelle le vecteur de logits en utilisant un hyper-paramètre **T**. La fonction softmax devient alors :

$$\sigma(z/T) = \frac{e^{z/T}}{\sum_{k=1}^{K} e^{z/T}}$$

Cette transformation "adouci" la fonction softmax et augmente l'entropie de sortie (avec $T > 1$). On entraîne ce paramètre T avec une loss de cross-entropy sur un ensemble de validation et les poids du réseau fixés. Puisque T effectue une transformation monotone sur les logits, la précision du modèle n'est pas affecté, seulement la confiance.

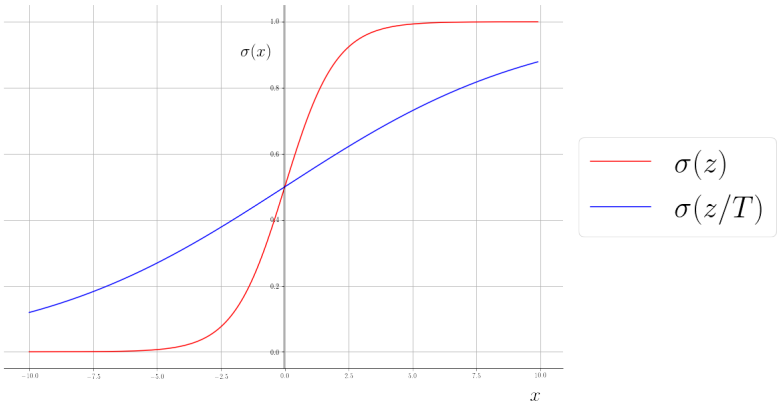

Figure 3: Fonctions Softmax 1D avec et sans temperature scaling.

[1] approximation empirique de l'espérance

3 Expérience

D'abord, on entraîne un réseau de neurone **ENet** avec le jeu de données *CamVid* auquel on a redéfini les classes 'car', 'pedestrian', et 'bicyclist' en 'unlabeled'. La figure 4 montre que lorsque le modèle voit un objet hors de la distribution sur laquelle il s'est entraîné (ici une voiture), il se trompe tout en étant confiant à plus de 90% (voir métrique msp).

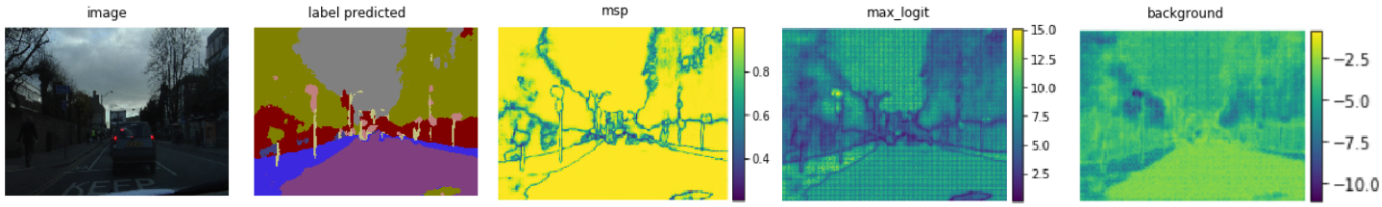

Figure 4: Valeurs de différentes métriques [2]

En appliquant la technique du *temperature scaling*, on espère voir la confiance diminuée aux endroits de l'image où des objets inconnus apparaissent. En optimisant le paramètre T, j'obtiens une température optimale $T_{opt} = 2.239$ et une mesure ECE qui diminue de 0.086 (avant température scaling) à 0.060. Cependant, en regardant à nouveau les valeurs de la confiance sur les pixels inconnus, je n'observe pas de différence malgré une légère augmentation moyenne des valeurs de max_logit et de background ... Il est possible que l'optimisation du paramètre T ne soit pas optimale ou bien qu'il y ait une erreur dans le code ou dans la gestion des données (ECE $= 10^{-2}$ paraît assez faible). J'ai également lu les articles ci-dessous avec des techniques intéressantes comme les Deep Ensembles, le MC dropout, l'Outlier Exposure mais sans avoir eu le temps de les intégrer dans l'implémentation ENet.

Le code peut être trouvé ici et provient en grande partie de là. Les ajouts principaux sont l'ajout d'un fichier main.ipynb qui peut tourner sur Google Colab , de temperature_scaling.py et quelques modifications mineurs dans les fichiers pré-existants. Quelques modèles ont été sauvegardés dans le répertoire save mais sans réels résultats.

[2] msp : max softmax probabilité d'une image de test

References

[1] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks, 2017.

[2] Dan Hendrycks, Steven Basart, Mantas Mazeika, Mohammadreza Mostajabi, Jacob Steinhardt, and Dawn Song. A benchmark for anomaly segmentation, 2019.

[3] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks, 2016.

[4] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure, 2018.

[5] Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep ensembles: A loss landscape perspective, 2019.

[6] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning, 2015.

[7] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation, 2016.

[8] Terrance DeVries and Graham W. Taylor. Learning confidence for out-of-distribution detection in neural networks, 2018.

[9] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks, 2015.

[10] Gianni Franchi, Andrei Bursuc, Emanuel Aldea, Severine Dubuisson, and Isabelle Bloch. Tradi: Tracking deep neural network weight distributions, 2019.

[11] Azadeh Sadat Mozafari, Hugo Siqueira Gomes, Wilson Leão, Steeven Janny, and Christian Gagné. Attended temperature scaling: A practical approach for calibrating deep neural networks, 2018.