ENSTA

Rapport

Rapport projet 2 computer vision Semi Supervised Learning

Élèves :
Nathan Gaubil

Encadrants :
Gianni Franchi (ENSTA)

1 Introduction

Le but du projet est : d'entraîner un CNN sur CIFAR 10 en utilisant 250 images annotées. On peut bien entendue utiliser les autres images sans les annotations. Il s'agit de Semi Supervised learning. Il faut entraîner un Wide ResNet-28-2 qui n'est pas pré-entraîné. L'objectif est de s'assurer que le CNN a de bonnes performances sur l'ensemble de tests de CIFAR 10 et qu'il ne sur-apprenne pas trop.

Le manque de données étiquetées constitue un obstacle majeur dans l'application de l'apprentissage en profondeur à des domaines pratiques tels que l'imagerie médicale. Cela peut entraîner des performances médiocres des modèles. Pour surmonter ce défi, il est possible de tirer parti des images non étiquetées en utilisant une approche d'apprentissage semi-supervisé appelée FixMatch. FixMatch est une méthode plus simple que les approches précédentes telles que UDA et ReMixMatch. L'article explique également comment FixMatch a amélioré l'état de l'art de l'apprentissage semi-supervisé avec une précision médiane de 78 pourcent et une précision maximale de 84 pourcent sur CIFAR-10 en utilisant seulement 10 images étiquetées.

La *consistency regularization* est un élément important des algorithmes semi-supervisés de pointe. Elle utilise les données non étiquetées en supposant que le modèle devrait produire des prédictions similaires lorsqu'il est alimenté avec des versions perturbées de la même image. L'idée a été popularisée par plusieurs travaux de recherche et est implémentée à travers une fonction de perte qui combine une perte de classification supervisée et une perte sur les données non étiquetées. La méthode de pseudo-étiquetage consiste à utiliser le modèle lui-même pour obtenir des étiquettes artificielles pour les données non étiquetées. Elle utilise des étiquettes dures et ne conserve que celles dont la plus grande probabilité de classe est supérieure à un seuil prédéfini. Ces deux méthodes sont souvent utilisées ensemble pour améliorer les performances des modèles semi-supervisés.

2 Pipeline

Nous allons étudier l'augmentation des données

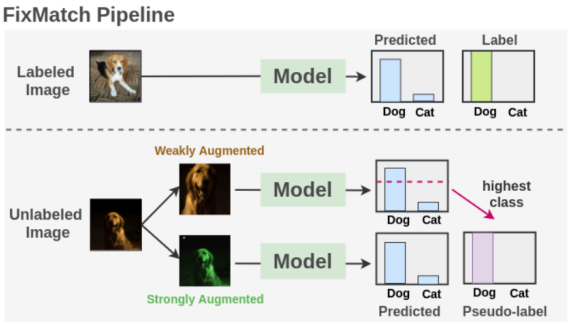

FIGURE 1 – Vision globale de FixMatch

2.1 Augmentation des données

FixMatch utilise deux types d'augmentations : "faibles" et "fortes". Les augmentations faibles consistent en une stratégie standard de retournement et de décalage aléatoire des images. Pour les augmentations fortes, deux méthodes basées sur RandAugment sont expérimentées, suivies de Cutout. RandAugment est une technique de régularisation pour l'apprentissage automatique qui augmente les données d'entraînement en appliquant des transformations aléatoires aux images. Cette méthode consiste à sélectionner un ensemble de transformations d'image aléatoires parmi une bibliothèque de transformations prédéfinies, telles que la rotation, le zoom ou la distorsion, et à appliquer ces transformations à l'image d'origine de manière aléatoire avec une intensité variable. Plus concrétement dans le code, on choisis n augmentation parmis les 14. La valeur de magnitude de M est un scalaire compris entre 0 et une valeur maximale prédéfinie, et elle détermine dans quelle mesure la transformation est appliquée à l'image originale. Une valeur plus petite de M donne une transformation plus douce, tandis qu'une valeur plus grande de M donne une transformation plus significative.

2.2 Modèle

WideResNet 28 2 est un modèle de réseau de neurones convolutifs (CNN) largement utilisé pour la classification d'images. Il s'agit d'une version améliorée du modèle ResNet, qui utilise des couches plus larges pour améliorer la capacité de représentation du réseau. WideResNet 28 2 a 28 couches de neurones et un facteur d'élargissement de 2, ce qui signifie que chaque couche a deux fois plus de filtres que dans un ResNet standard. Cette augmentation de la largeur du réseau améliore considérablement sa précision de classification, en particulier pour les ensembles de données complexes et les images de haute résolution.

2.3 Entraînement du modèle

Dans FixMatch, les mini-batchs sont constitués d'un mélange de données étiquetées et non étiquetées. Les données étiquetées sont les images pour lesquelles les étiquettes sont connues, tandis que les données non étiquetées sont celles pour lesquelles les étiquettes ne sont pas connues. Les données non étiquetées sont multipliées par un facteur de MU(par exemple, MU = 7 pour CIFAR-10) pour augmenter le nombre d'échantillons non étiquetés dans chaque mini-batch. C'est un hyperparamètre important.

La partie supervisée de FixMatch utilise une perte de cross-entropy H() pour la tâche de classification sur les images étiquetées. La perte totale pour un mini-batch est définie par Ls et est calculée en prenant la moyenne de la perte de cross-entropy pour chaque image dans le mini-batch. La formule pour la perte supervisée est la suivante :

$$l_s = \frac{1}{B} \sum_{b=1}^{B} H(p_b, p_m(y|\alpha(x_b))) \tag{1}$$

où B est la taille du mini-batch, p_b est la prédiction de la classe pour l'image b, $p_m(y|\alpha(x_b))$ est la prédiction du modèle pour l'image perturbée, et α est la fonction de perturbation pour les images non étiquetées.

Pour les images non étiquetées, des augmentations "faibles" sont d'abord appliquées pour obtenir une version perturbée de l'image, puis la classe la plus probable est déterminée en appliquant la fonction argmax. Cette classe prédite est considérée comme l'étiquette artificielle (ou pseudo-étiquette) associée à cette image, et est comparée aux prédictions du modèle pour une version fortement perturbée de l'image non étiquetée. La formule pour la prédiction du modèle sur une image perturbée est :

$$q_b = p_m(y|\alpha(u_b)) \tag{2}$$

et la formule pour la pseudo-étiquette est :

$$\hat{q}_b = argmax(q_b) \tag{3}$$

Nous combinons finalement ces deux pertes pour obtenir une perte totale que nous optimisons pour améliorer notre modèle. λ_u est un hyperparamètre scalaire fixe qui décide de la contribution relative des pertes d'images non étiquetées par rapport aux pertes d'images étiquetées.

La formule de la perte totale est :

$$loss = l_s + \lambda_u l_u \tag{4}$$

Un résultat intéressant découle de λ_u. Des travaux antérieurs ont montré que l'augmentation du poids pendant l'entraînement est bénéfique. Mais dans FixMatch, cela est intégré dans l'algorithme lui-même. Au début de l'entraînement, le modèle n'a pas confiance en les données étiquetées, donc ses prédictions de sortie sur les données non étiquetées seront en dessous du seuil. Ainsi, le modèle ne sera entraîné qu'avec les données étiquetées. Mais au fur et à mesure de l'entraînement, le modèle devient plus confiant dans les données étiquetées et les prédictions sur les données non étiquetées commencent également à dépasser le seuil. Par conséquent, la perte commencera bientôt à intégrer les prédictions sur les images non étiquetées.

2.4 Généralisation

Pour permettre la généralisation de notre modèle et éviter le sur-apprentissage certaine techniques sont mises en place. Tout d'abord, la régularisation est particulièrement importante. Dans l'implémentation officiel, ils utilisent une régularisation de dégradation de poids simple. L'utilisation de SGD standard avec momentum est plus performante. Ils n'ont pas trouvé de différence substantielle entre le momentum standard et le momentum de Nesterov. Pour un scheduler du taux d'apprentissage, j'utilise une décroissance du taux d'apprentissage en cosinus qui définit le taux d'apprentissage à $\eta cos(7\pi k/16K)$, où η est le taux d'apprentissage initial, k est l'étape d'entraînement actuelle et K est le nombre total d'étapes d'entraînement. Enfin, nous rapportons les performances finales en utilisant une moyenne mobile exponentielle des paramètres du modèle (EMA).

L'EMA est utilisée pour lisser les paramètres du modèle en réduisant le bruit et en évitant le surajustement (overfitting). Elle permet également d'ajuster la vitesse à laquelle les anciennes données sont oubliées et les nouvelles données sont intégrées.

Le calcul de l'EMA consiste à donner un poids décroissant exponentiellement aux données précédentes et à donner un poids plus élevé aux données récentes. Plus précisément, l'EMA pour un instant t est calculée à partir de la valeur actuelle x_t et de la valeur précédente de l'EMA pour un instant précédent $t-1$. Le paramètre alpha est utilisé pour contrôler la vitesse à laquelle l'EMA réagit aux nouvelles données et est généralement fixé entre 0 et 1.

Les hyperparamètres sont dans les annexes.

3 Conclusion

Lors de sa présentation, FixMatch était un algorithme SSL plus simple qui avait atteint des résultats de pointe sur de nombreux ensembles de données. Pour l'entraînement, seules des pertes d'entropie croisée standard étaient utilisées sur les données étiquetées et non étiquetées, et l'objectif pouvait être écrit en quelques lignes de code seulement.

Il était important de prendre en compte la décroissance de poids et le choix de l'optimiseur pour obtenir les meilleurs résultats possibles avec FixMatch. En effet, ces facteurs jouent un rôle crucial dans l'efficacité de l'algorithme, même lorsque l'architecture du modèle est contrôlée.

Les résultats sont dans les annexes.

4 Annexe

4.1 Train

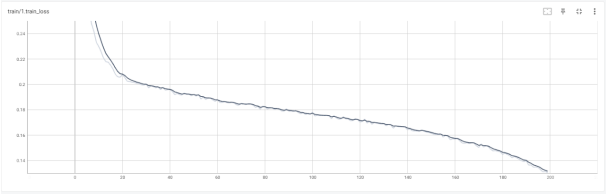

FIGURE 2 – Training loss

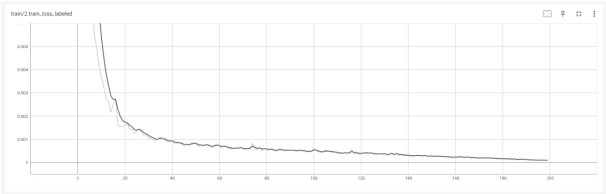

FIGURE 3 – Training loss label images

On observe que l'entrainement est plutot stable, les pertes décroit de manière uniforme. J'ai décié de n'utiliser que 200 epochs par rapport à 1024 du papier car l'entrainement est trop long et j'obtiens déja un score largement suffisant pour montrer que ça fonctionne.

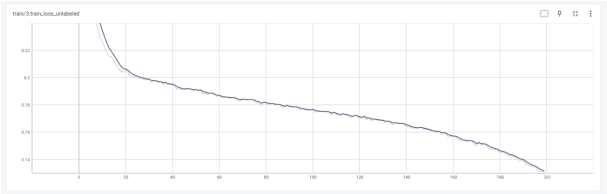

FIGURE 4 – Training loss unlabeled images

4.2 Test

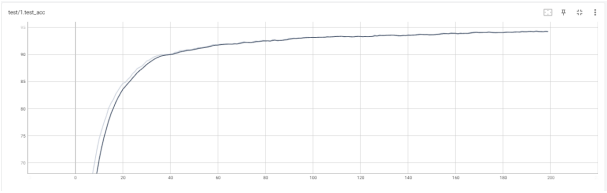

FIGURE 5 – Testing accuracy

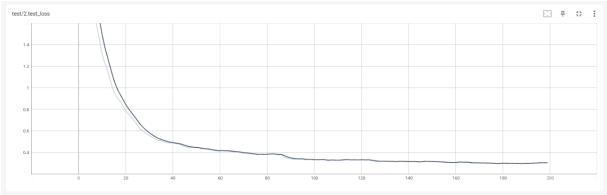

FIGURE 6 – Testing loss

4.3 Paramètres du modèle

Variable	Valeur
BATCH_SIZE	64
EVAL_STEPS	1024
LEARNING_RATE	0.03
NUM_EPOCHS	200
LAMBDA_U	1
WEIGHT_DECAY	5e-4
MU	7
TEMPERATURE	1
THRESHOLD	0.95

TABLE 1 – Valeurs des variables utilisées dans l'entraînement du modèle