CR3 PROJET HARMONISATION DES COULEURS M2

Roland BERTIN-JOHANNET, Benjamin PRE

November 20, 2022

Travail réalisé

Depuis la dernière fois, nous avons :

1. Implémenté un'algorithme de down sampling et d'upsampling avec carte des différences par bloc.
2. Equilibré les classes de notre base d'images (certaines avaient 4000 images et d'autres 400) en relançant l'algorithme de classification de types d'harmonie sur les classes les plus grosses, avec des paramètres différents (par exemple, en baissant le seuil de refus de cluster pour diminuer la taille de la classe monochromatique).
3. Converti toute la base de données en LCH pour éviter d'avoir à le faire pendant l'entraînement du modèle.
4. Relancé l'expérimentation du second CR sur les scores d'harmonies, mais avec la bonne conversion cette fois-ci.
5. Revu notre architecture de CycleGAN pour qu'il prenne en entrée les composante L,C,H et aie en sortie les composantes C et H (L est récupérée de l'entrée et concaténée à l'output pour former l'image LCH finale. Le raisonnement est que l'on ne veut changer que les couleurs mais quand même donner l'information de Luminance au modèle.)
6. Implémenté la première couche MLE (maximum likelihood estimation) décrite dans l'article *A Color-Pair Based Approach for Accurate Color Harmony Estimation*[1] pour obtenir un score d'harmonie entre deux couleurs (l'article étant plus récent, c'était intéressant).
7. étendu ce nouveau score d'harmonie à des images comme nous le faisions pour l'autre, et lancé l'expérimentation consistant à calculer l'harmonie de 100 images de chaque classes et afficher leurs distributions dans un boxplot.
8. créé une classe "unknown" dans nptre base de données qui sert d'input à nos modèles : l'utilisateur donne une image au type d'harmonie unknown et le modèle doit savoir en faire une image du bon type d'harmonie.

Travail à réaliser

D'ici au prochain compte-rendu, nous espérons :

1. Entraîner un modèle par type d'harmonie, et faire un peu plus de *fine-tuning*.
2. Commencer à explorer l'idée d'une fonction de coût dérivable (donc rétro-propageable) sur le générateur qui indique combien on se rapproche de l'harmonie voulue.
3. Commencer à explorer l'idée de changer nos GANs en Wasserstein GANS, et d'entraîner le critique (le discriminateur) à donner un score d'harmonie (que nous savons calculer) en même temps qu'on l'entraîne en compétition avec le générateur.

[1] https://www.researchgate.net/publication/337247169_A_Color-Pair_Based_Approach_for_Accurate_Color_Harmony_Estimation

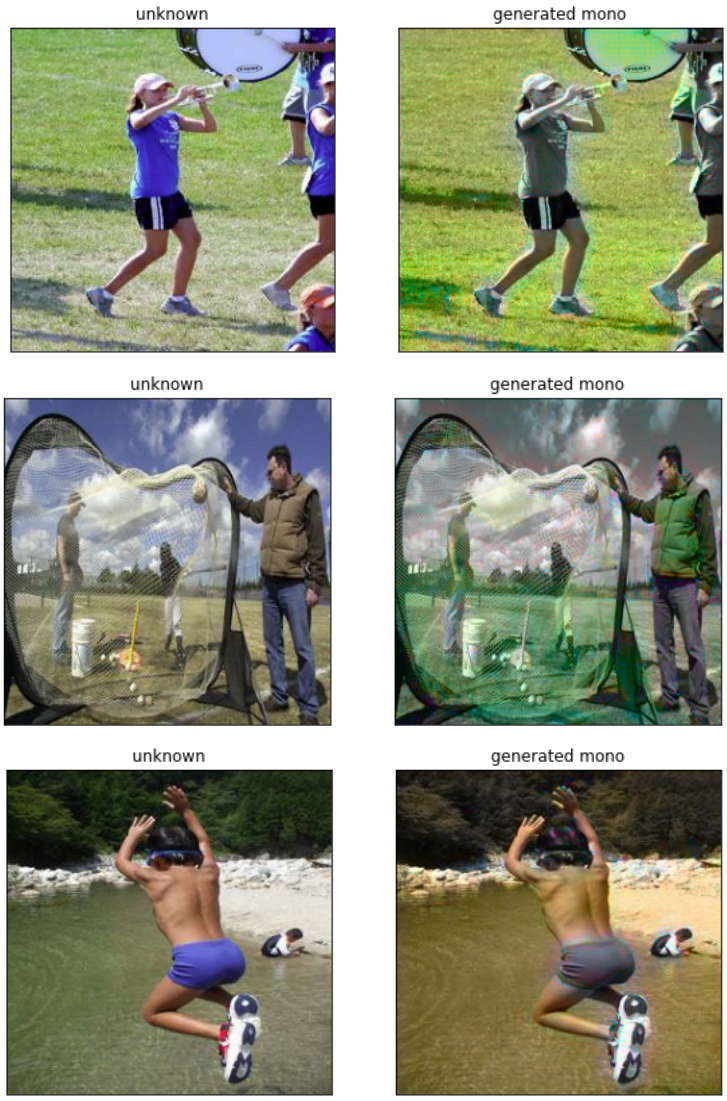

Figure 1: Exemples de résultats de notre CycleGAN partant de la classe "unknown" et générant des images de style monochrome.

4. Implémenter le reste de la méthode de Yang et al. et l'utiliser pour comparer, par rapport à une nouvelle vérité de terrain (la base Kuler décrite dans l'article), l'efficacité des deux scores.

5. Voir si une combinaison linéaire des deux scores ne prédit pas mieux l'harmonie que les scores individuellement (avec une régression linéaire).

6. Commencer l'interface graphique (notamment comment mélanger nos implémentations en python et c++).