Optimisation du décodage par liste de vidéos corrompues basée sur une architecture CNN

Yujing Zhang, Stéphane Coulombe, François-Xavier Coudoux, Anthony Trioux, Patrick Corlay

Optimisation du décodage par liste de vidéos corrompues basée sur une architecture CNN

Y. Zhang[1,2] S. Coulombe[1] F-X. Coudoux[2] A. Trioux[2] P. Corlay[2]

[1] Dept. of Software and IT Engineering, École de technologie supérieure, Montreal, Canada

[2] UMR 8520 - IEMN, DOAE, Univ. Polytechnique Hauts-de-France, CNRS, Univ. Lille, YNCREA, Centrale Lille, France

{yujing.zhang.1@ens.etsmtl.ca, stephane.coulombe@etsmtl.ca}

{francois-xavier.coudoux, anthony.trioux, patrick.corlay}@uphf.fr

Résumé

Cet article présente une solution de décodage par liste optimisée pour des vidéos corrompues par des erreurs de transmission. Elle est basée sur l'évaluation de la qualité des images sans référence utilisant un réseau de neurones convolutif (CNN) qui gère efficacement les distorsions non uniformes. À l'issue d'un processus de décodage par liste, nous évaluons la qualité de chaque image candidate générée (sans référence) afin de sélectionner la meilleure. Lorsque l'erreur de transmission se produit dans une image intra, notre architecture a une précision de décision de plus de 98% contre 46% pour l'architecture CNN originale pré-entraînée. Pour les erreurs dans une image inter, c'est 79% contre 33%.

Mots clefs

Transmission vidéo, distorsions non uniformes, évaluation de la qualité des images, réseau de neurones convolutif.

1 Introduction

Nous assistons à un développement très rapide des applications impliquant la transmission de contenus vidéos. Cependant, les erreurs de transmission sur des réseaux sans-fil compromettent gravement la qualité visuelle des contenus vidéos reçus, ce qui se traduit par une mauvaise qualité d'expérience pour l'utilisateur final. Différentes approches existent dans la littérature pour réparer les paquets vidéos erronés reçus [1, 2, 3, 4, 5, 6]. Parmi celles-ci, nous nous intéressons aux approches de décodage par liste qui exploitent les paquets reçus corrompus. À partir de chaque paquet corrompu, la méthode génère plusieurs paquets *candidats*. Ces candidats représentent diverses tentatives de correction du paquet erroné. Le défi consiste à estimer sans référence au récepteur la qualité de chacun de ces candidats pour ensuite choisir le meilleur. Ce dernier correspondra idéalement à la version intacte originalement transmise.

Dans cet article, nous proposons donc un cadre d'optimisation du décodage par liste où une évaluation sans référence de la qualité visuelle permet d'identifier le meilleur candidat parmi une liste de plusieurs. Notre approche est basée sur l'usage d'un réseau de neurones convolutif (CNN) modifié afin de permettre la prise en compte de distorsions non-uniformes dues aux erreurs de transmission. Nos principales contributions sont :

1. Une nouvelle méthode d'évaluation de la qualité basée sur le CNN présenté dans [7], mais améliorée à plusieurs égards, dont une normalisation et une mesure de qualité locales, opérant par patch, pour supporter des distorsions non-uniformes dans les images.
2. Une nouvelle base de données constituée de vidéos encodées avec la norme High Efficiency Video Coding (HEVC) [8] et auxquelles nous avons injecté des erreurs de transmission. Cela mène à des images possédant des artéfacts non-uniformément distribués spatialement sur lesquelles notre système peut s'entrainer.
3. Un nouveau cadre d'optimisation du décodage par liste capable de sélectionner la vidéo ayant la meilleure qualité visuelle parmi plusieurs candidats.

À la section 2, nous présentons la méthode proposée. À la section 3, nous présentons nos résultats expérimentaux. Enfin, nous présentons nos conclusions à la section 4.

2 La méthode proposée

2.1 Cadre d'optimisation de décodage

À la Figure 1, nous proposons un cadre pour améliorer le processus de décodage par liste constitué de : 1) la génération d'une base de données d'images avec distorsions non uniformes, 2) l'apprentissage pour l'évaluation de la qualité (entraînement), et 3) la sélection du meilleur candidat. Le processus de génération de la base de données comprend différentes étapes : l'encodage vidéo par la norme HEVC, la génération d'erreurs de transmission et le décodage vidéo par liste, sans dissimulation d'erreur, pour obtenir les N candidats représentant des tentatives, pour la plupart infructueuses, de correction de la vidéo. Le processus d'entraînement à évaluer la qualité comprend la conversion d'images (du format YUV au format utilisé pendant l'apprentissage), la génération de patchs et l'apprentissage du réseau de neurones de manière supervisée en utilisant une métrique de qualité avec référence complète. Le choix du meilleur candidat est réalisé en identifiant le candidat avec la qualité la plus élevée.

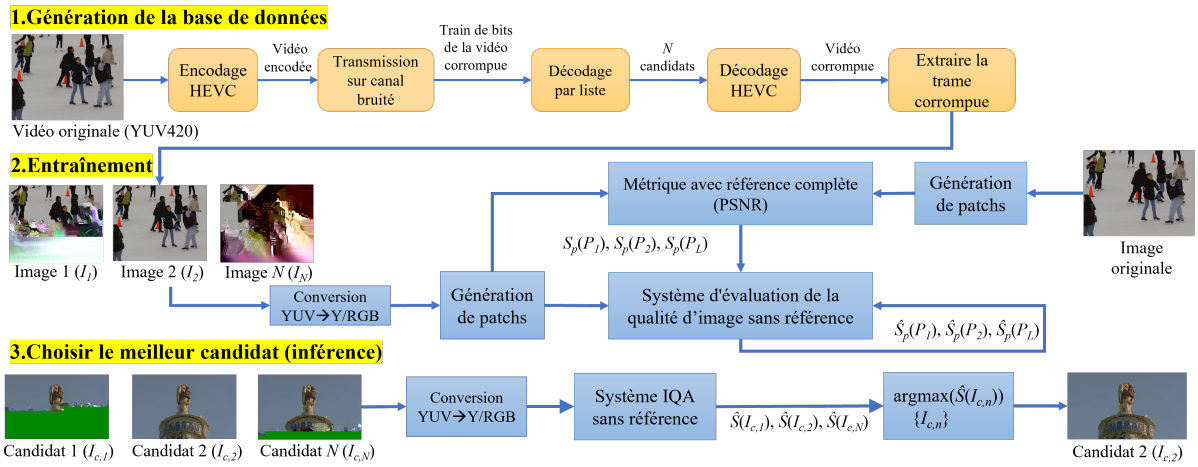

FIGURE 1 – *Cadre proposé pour optimiser le décodage par liste de vidéos corrompues lors de la transmission*

2.2 Méthode d'évaluation de la qualité visuelle améliorée basée sur le CNN

Plusieurs métriques basées sur les CNN [7, 9] séparent une image en plusieurs patchs de taille réduite et extraient les caractéristiques de chaque patch pour évaluer leur qualité. Les modèles basés sur les patchs attribuent souvent à tous les patchs de l'image le même niveau de qualité que l'image complète lors de l'apprentissage [10], ce qui donne de bons résultats pour les distorsions uniformes, mais n'est pas une approche souhaitable lorsque l'on considère des distorsions non uniformes. Nous proposons donc d'utiliser des scores locaux afin que chaque patch ait un score de qualité qui lui est propre. Cela peut aider le réseau de neurones à apprendre plus efficacement les distorsions locales.

Nous choisissons l'architecture CNN proposée dans [7] comme architecture de base et l'améliorons afin de l'adapter à notre objectif. L'architecture originale est un réseau neuronal à 5 couches, dont 1 couche convolutive, 2 couches de pooling et 2 couches entièrement connectées. Comme illustré à la Figure 2, nous utilisons, dans cet article, la structure du réseau suivante : $64\times64\times3-58\times58\times50-2\times50-800-800-1$. Au lieu d'effectuer la simulation uniquement sur le canal de luminance comme dans [7], nous étendons nos expériences à trois canaux R, G, B. L'architecture de base [7] utilise une méthode de normalisation du contraste local. Supposons que la valeur d'intensité d'un pixel à l'emplacement (i,j) soit $v(i,j)$, les auteurs calculent alors sa valeur normalisée $v_n(i,j)$ comme suit :

$$v_n(i,j) = \frac{v(i,j)-\mu(i,j)}{\sigma(i,j)+C}, \text{ avec}$$

$$\mu(i,j) = \frac{1}{(2W+1)^2}\sum_{p=-W}^{p=W}\sum_{q=-W}^{q=W} v(i+p,j+q)$$

$$\sigma(i,j) = \sqrt{\frac{1}{(2W+1)^2}\sum_{p=-W}^{p=W}\sum_{q=-W}^{q=W}[v(i+p,j+q)-\mu(i,j)]^2}$$

où C est une constante positive qui empêche la division par zéro. La taille de la fenêtre de normalisation est de $(2W+1)\times(2W+1)$ pixels avec $W=3$. Cependant, cette méthode pose un problème lorsqu'elle est appliquée à des patchs uniformes. Le problème est que le décodeur initialise chaque patch au format YUV à (0,0,0) lorsqu'il démarre le décodage d'une image. Considérant un seul canal, par exemple Y, nous ne pouvons pas faire la distinction entre un patch uniforme bien reçu dont la valeur est normalisée à 0 et un patch erroné qui est uniforme parce qu'il a été initialisé à 0 par le décodeur. Cette situation est problématique quand elle survient dans la base de données d'entraînement, car le réseau neuronal devient confus pendant l'apprentissage. En effet, à l'issue de la normalisation, un patch uniforme et un patch en erreur deviennent identiques et entrent dans les couches du CNN avec des scores de référence différents pour l'apprentissage. Pour éviter ce problème, nous améliorons la normalisation locale en séparant ces deux situations. Lorsque nous détectons $\sigma(i,j)=0$ dans les patchs d'entrée, nous calculons $\mu(i,j)$. Si $\mu(i,j)\neq 0$, nous forçons $v_n(i,j)$ à être égal à $\epsilon\neq 0$ après normalisation. Nous utilisons l'Eq.(1) sur chaque canal d'une image en format RGB où nous forçons la valeur à (0,0,0) lorsque le décodeur récupère YUV à (0,0,0) suite à une erreur.

$$v_n(i,j) = \begin{cases} 0, & \text{si } \sigma(i,j)=0 \text{ et } \mu(i,j)=0 \\ \epsilon, & \text{si } \sigma(i,j)=0 \text{ et } \mu(i,j)\neq 0 \end{cases} \quad (1)$$

Notre réseau neuronal est tout d'abord entraîné sur des patchs non chevauchants de 64×64 pixels, correspondant à un coding tree unit (CTU) en HEVC [8], provenant d'images haute définition. Pour l'entraînement, nous attribuons à chaque patch un score de qualité Peak signal-to-noise ratio (PSNR), calculé entre le patch corrompu et le patch correspondant dans l'image originale avant encodage (métrique avec référence). Pour les tests, nous utilisons la moyenne des scores de patchs prédits pour chaque image afin d'obtenir le score $\hat{S}(I)$ de qualité au niveau image :

$$\hat{S}(I) = \frac{1}{L}\sum_{l=0}^{L-1}\hat{S}_p(P_l)$$

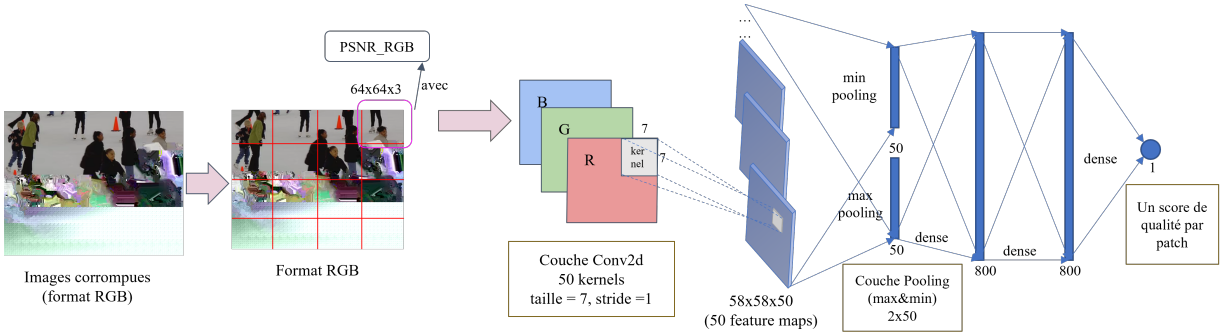

FIGURE 2 – *Architecture CNN proposée pour évaluer la qualité des images avec distorsions non-uniformes*

où $\hat{S}_p(P_l)$ indique le score de qualité prédit pour le patch P_l par le CNN (métrique sans référence), et L est le nombre total de patchs dans l'image. Le fait d'utiliser des petits patchs en entrée élargit considérablement l'échantillon d'apprentissage pour le CNN et évite le problème de manque de données rencontré lors de l'utilisation d'un ensemble d'images complètes. Nous utilisons la fonction de perte de [7], la descente de gradient stochastique et la rétropropagation sont utilisées pendant l'entraînement. Nous utilisons un ensemble de validation pour éviter un ajustement excessif et conservons les paramètres du modèle qui génèrent la valeur de *Spearman Rank Order Correlation Coefficient* la plus élevée sur l'ensemble de validation.

3 Résultats expérimentaux

3.1 Base de données vidéo utilisée

Toutes les séquences originales utilisées dans nos expériences proviennent de [11, 12]. Les vidéos collectées sont au format YUV avec une résolution de 1920×1024. Nous extrayons les 10 premières images de chaque vidéo pour les encoder avec la norme HEVC. Parmi les différentes valeurs du pas de quantification (QP) possibles, nous avons choisi 37 qui correspond à une valeur fréquemment utilisée. Nous supposons que chaque image encodée est contenue dans un seul paquet vidéo. La première image de la vidéo encodée est une image intra (I), et les 9 images suivantes sont des images inter (P). Pour simuler la combinaison d'une erreur de transmission suivie d'un décodage par liste où les bits sont inversés à différents endroits, des positions de bits inversés sont choisies en fonction de l'équation $p = \alpha \times M$, où $\alpha = \{0.1, 0.2, ..., 0.9, 0.99\}$ et M est la taille de chaque paquet. Ainsi, pour chaque séquence, nous avons 11 candidats à chaque fois (dont 1 est le candidat sans erreur). Nous obtenons 990 images corrompues à partir de 90 images de référence et finalement 475 200 patchs pour l'apprentissage, avec une taille de patch de 64×64 pixels. Nous avons séparé notre base de données en ensembles d'entraînement et de test, avec une répartition de 60% et 40%, respectivement. Chaque patch est associé à un score PSNR dans l'intervalle [0, 50] dB, qui est normalisé dans l'intervalle [0, 1] pendant l'apprentissage. Sur la base de résultats empiriques de simulation, nous fixons $\epsilon = -0.013$ dans l'Eq.(1).

3.2 Évaluation des performances

Nous entraînons et testons le modèle original CNN et notre version améliorée sur la nouvelle base de données dédiée. Afin de mieux évaluer les performances des modèles, nous définissons plusieurs métriques. Comme indiqué ci-dessous, $\overline{S}_{\text{intact}}$ indique le PSNR moyen, par rapport aux versions originales, de toutes les images intactes, qui sont compressées mais reçues sans erreur de transmission. Il est calculé sur YUV selon [8] et noté PSNR_{YUV}. $\overline{S}_{\text{système}}$ représente le PSNR moyen, par rapport aux versions originales, de toutes les images sélectionnées par une méthode donnée. $\overline{S}_{\text{diff}}$ est, pour une méthode, la différence absolue entre la qualité moyenne des images sélectionnées et celle des images intactes. N correspond au nombre de séquences originales considérées. K est le nombre de candidats, $I_{\text{c},i}$, parmi lesquels choisir pour chaque image corrompue.

$$\overline{S}_{\text{intact}} = \frac{1}{N}\sum_{n=0}^{N-1} \text{PSNR}_{\text{YUV}}(I_{\text{original},n}, I_{\text{intact},n}),$$

$$\overline{S}_{\text{système}} = \frac{1}{N}\sum_{n=0}^{N-1} \text{PSNR}_{\text{YUV}}(I_{\text{original},n}, I_{\text{système},n}),$$

$$\text{où } I_{\text{système}} = \underset{\{I_{\text{c},i},\ 0\le i<K\}}{\arg\max}\ \hat{S}(I_{\text{c},i}),\quad \overline{S}_{\text{diff}} = \left|\overline{S}_{\text{intact}} - \overline{S}_{\text{système}}\right|$$

Méthodes	Précision	$\overline{S}_{\text{intact}}$ (dB)	$\overline{S}_{\text{système}}$ (dB)	$\overline{S}_{\text{diff}}$ (dB)
CNN_Y_G pré-entraîné [7]	45.6%	39.18	28.69	10.49
CNN_Y proposé	93.0%		38.39	0.79
CNN_RGB proposé	96.5%		38.88	**0.30**
CNN_Y_NL proposé	**98.2%**		38.60	0.58
CNN_RGB_NL proposé	96.5%		38.88	**0.30**

TABLEAU 1 – *Performances sur les images codées en intra*

Méthodes	Précision	$\overline{S}_{\text{intact}}$ (dB)	$\overline{S}_{\text{système}}$ (dB)	$\overline{S}_{\text{diff}}$ (dB)
CNN_Y_G pré-entraîné [7]	33.3%	38.62	32.99	5.63
CNN_Y proposé	60.0%		30.19	8.43
CNN_RGB proposé	66.7%		36.49	2.13
CNN_Y_NL proposé	77.0%		36.55	2.07
CNN_RGB_NL proposé	**79.0%**		36.71	**1.91**

TABLEAU 2 – *Performances sur les images codées en inter*

Les Tableaux 1 et 2 présentent les résultats expérimentaux obtenus en utilisant des images codées respectivement en *intra* et en *inter*, en comparant la méthode CNN originale à différentes configurations de simulation. Pour les résultats

sur image *inter*, l'erreur a directement frappé l'image inter en question, et ne correspond pas à une propagation d'erreur survenue dans l'image intra précédente. Les meilleurs résultats sont indiqués en gras.

La première ligne de chaque tableau, CNN_Y_G pré-entraîné, utilise le modèle déjà entraîné de l'article [7], qui applique le même score pour chaque patch de la composante Y de l'image (score global), et teste avec notre base de données avec distorsions non uniformes. CNN_Y proposé indique la solution proposée où on utilise un score différent par patch de luminance et où nous ré-entraînons et testons sur notre base de données (comme toutes les méthodes proposées). Nous pouvons constater l'intérêt d'utiliser un score local par patch et de ré-entraîner sur la base de données proposée puisque la précision passe de 46% à 93% sur les images intra et de 33% à 60% sur les images inter. Nous croyons que ce score local permet de mieux apprendre les caractéristiques des distorsions provenant d'erreurs de transmission. CNN_RGB proposé indique que les images utilisées initialement au format YUV sont converties au format RGB pour l'entraînement et l'inférence. Une méthode avec le suffixe _NL indique une configuration qui applique notre méthode de normalisation locale améliorée (Eq.(1)). Pour les images codées en intra, l'utilisation de la normalisation locale permet d'obtenir une meilleure précision et une moins grande différence de qualité lorsque le CNN utilise le canal Y. Cependant, aucune amélioration n'est obtenue lorsque RGB est utilisé. Pour les images inter, l'utilisation de la normalisation locale permet d'améliorer significativement les performances tant pour Y que pour RGB avec une précision qui passe de 60% à 77% pour Y et de 68% à 79% pour RGB. Finalement, bien que les performances soient similaires pour Y et RGB en intra, l'usage de RGB performe mieux en inter. Nous croyons qu'il a l'avantage de pouvoir identifier les distorsions de couleurs. On note que la précision est beaucoup plus faible pour les images inter que pour les intra. En effet, les erreurs de transmission dans les images inter n'engendrent pas de pertes aussi importantes de qualité que sur les images intra, ce qui rend plus difficile l'apprentissage du modèle.

Nous pouvons voir l'intérêt du score local, du ré-entraînement sur notre base de données et la normalisation locale. Néanmoins, les performances ne sont pas optimales et plusieurs travaux sont envisagés pour les améliorer. Par exemple, on pourrait penser à modifier la taille des patchs pour pouvoir détecter les discontinuités aux frontières des CTUs HEVC, rendues visibles suite à la présence d'erreurs de transmission. Aussi, nous pourrions adapter notre système en opérant directement en YUV plutôt qu'en Y ou RGB pour détecter les distorsions de couleur tout en évitant des conversions supplémentaires. Les derniers résultats de ces améliorations seront présentés lors de la conférence.

4 Conclusion

Nous avons développé une architecture d'évaluation de la qualité d'image reposant sur un CNN existant, modifié pour devenir sensible à des distorsions non uniformes, rencontrées lors de transmission avec erreurs. Cette méthode peut être utilisée pour extraire la meilleure reconstruction d'une liste d'images candidates générées à l'issue d'un processus de correction d'erreurs. Sur cette base, nous présentons également un cadre de simulation pour simuler le processus de génération d'images candidates, d'évaluation de la qualité de l'image et de sélection de la meilleure image. Notre architecture possède une précision de décision de plus de 98% lorsque l'erreur de transmission est localisée dans une image intra et d'environ 80% en inter.

Références

[1] Yao Wang et Qin-Fan Zhu. Error control and concealment for video communication : A review. *Proceedings of the IEEE*, 86(5) :974–997, 1998.

[2] W-Y Kung et al. Spatial and temporal error concealment techniques for video transmission over noisy channels. *IEEE transactions on circuits and systems for video technology*, 16(7) :789–803, 2006.

[3] Xijin Liu et al. Exploiting error-correction-CRC for polar SCL decoding : A deep learning-based approach. *IEEE Transactions on Cognitive Communications and Networking*, 6(2) :817–828, 2020.

[4] Jinzhi Lin et al. Joint source-channel decoding of polar codes for HEVC-based video streaming. *ACM Trans. Multimedia Comput. Commun. Appl.*, 18(4), mar 2022.

[5] Vivien Boussard et al. Table-free multiple bit-error correction using the CRC syndrome. *IEEE Access*, 8 :102357–102372, 2020.

[6] Galina Sabeva et al. Robust decoding of H.264 encoded video transmitted over wireless channels. Dans *2006 IEEE Workshop on Multimedia Signal Processing*, pages 9–13, 2006.

[7] Le Kang et al. Convolutional neural networks for no-reference image quality assessment. Dans *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1733–1740, 2014.

[8] Vivienne Sze et al. *High Efficiency Video Coding (HEVC) : Algorithms and Architectures*. Springer Publishing, 2014.

[9] Simone Bianco et al. On the use of deep learning for blind image quality assessment. *Signal, Image and Video Processing*, 12 :355–362, 2018.

[10] Junyong You et Jari Korhonen. Transformer for image quality assessment. Dans *2021 IEEE international conference on image processing (ICIP)*, pages 1389–1393. IEEE, 2021.

[11] Xiph.org video test media [derf's collection].

[12] Margaret H. Pinson. The consumer digital video library [best of the web]. *IEEE Signal Processing Magazine*, 30(4) :172–174, 2013.