Accélération du partitionnement des blocs intra dans l'encodeur VVC via régression sur les coûts débit-distorsion

M.E.A. Kherchouche[1,2], F. Galpin[1], T. Dumas[1], D. Menard[2], L. Zhang[2]
[1] InterDigital, R&I, 845a Avenue des Champs Blancs, 35510 Cesson-Sévigné
[2]Univ Rennes, INSA Rennes, CNRS, IETR - UMR 6164, F-35000 Rennes, France

Résumé

VVC, dernière norme MPEG de compression vidéo 2D, réduit le débit binaire de 50% par rapport à son prédécesseur HEVC à qualité visuelle équivalente. Cependant, cette amélioration s'accompagne d'une complexité d'encodage accrue, notamment à cause du nombre élevé de partitionnements de blocs évaluées du côté encodeur. Cet article propose une méthode d'accélération du partitionnement des blocs intra dans l'encodeur VVC en utilisant un modèle de régression multi-sorties qui prédit les coûts débit-distortion des partitionnements possibles pour des unités de codage de taille 32×32*. Les expériences montrent que cette approche améliore la flexibilité des compromis entre complexité d'encodage et performance de compression tout en introduisant de nouveaux points de compromis par rapport à l'encodeur d'origine.*

Mots clefs

Compression vidéo, apprentissage profond, régression multi-sorties.

1 Introduction

La prolifération des contenus vidéo en ligne et en streaming augmente le trafic vidéo, nécessitant de nouvelles techniques de codage. Le Joint Video Experts Team (JVET) a développé le codec VVC [1] et son logiciel de référence VTM pour répondre à ce besoin et tester les outils de codage de la norme VVC. Le processus de codage peut être considéré comme un problème d'optimisation combinatoire et comporte plusieurs étapes : partitionnement des images, prédiction intra/inter, transformation, quantification et codage entropique. Ce travail se concentre sur l'étape de partitionnement dans les slices codées en intra. Dans cet article, nous proposons une méthode de réduction de la complexité d'encodage VVC en utilisant la régression sur les coûts débit-distorsion. Cette technique repose sur un réseau de neurones convolutionnel (CNN) qui prend une unité de codage (CU) de taille 32×32 de composante luma et le paramètre de quantification (QP) pour produire un vecteur de 6 coûts débit-distorsion, un pour chaque mode de partitionnement possible. Ensuite, $n \leq 6$ modes de partitionnement peuvent être testés sur cette CU via l'optimisation débit-distorsion (RDO), n étant déterminé par des seuils prédéfinis.

2 Contexte

2.1 Complexité d'encodage

Dans un codec hybride basé blocs, chaque image d'une séquence vidéo donnée est divisée récursivement en blocs. Dans HEVC [2], chaque image est divisée en unités d'arbre de codage (CTU). Une CTU/CU donnée peut être divisé en 4 sous-CUs carrées non superposées de même taille, appelées division en arbre à quatre branches (QT). Par exemple, dans HEVC, pour une slice intra, une optimisation causale complète débit-distorsion (RDO) sur une CTU de taille 64×64 teste au maximum 341 blocs. Cela correspond à la reconstruction de 20480 pixels par mode testé. En plus de QT et de l'absence de division, VVC permet le partitionnement d'une CU en sous-CUs rectangulaires via Multi-Type Tree (MTT), comprenant Binary Tree (BT) et Ternary Tree (TT). Dans BT, la CU est divisée en 2 sous-CUs de même taille, avec des divisions horizontales (BTH) et verticales (BTV). Dans TT, la CU est divisée en 3 sous-CUs avec un ratio de taille $1:2:1$, avec des divisions horizontales (TTH) et verticales (TTV). Pour une slice intra en VVC, une optimisation complète débit-distorsion sur une CTU de taille 64×64 peut tester jusqu'à $721k$ blocs, correspondant à la reconstruction de $19M$ pixels par mode testé. Comparé à l'encodeur HEVC, environ 2000 fois plus de blocs et 1000 fois plus de pixels doivent être traités dans le pire des cas.

2.2 Travaux connexes

Les chercheurs ont abordé la complexité élevée de l'optimisation débit-distorsion (RDO) avec des solutions de classification, divisées en deux groupes : l'apprentissage supervisé avec des CNNs et les autres méthodes (heuristiques ou apprentissage par renforcement). Notre proposition appartient au premier groupe.

Concernant le premier groupe, dans HEVC, Xu et al. [3] créent un réseau neuronal convolutif hiérarchique à terminaison précoce qui fournit 21 prédictions cartographiant les divisions CTU/CU. Dans VP9 [4], Mukherjee *et al.* [5] conçoivent un réseau de neurones convolutionnel entièrement hiérarchique prédisant les arbres de partitionnement des macro-blocs via une approche ascendante. Dans HEVC avec QTBT (HEVC avec partitionnement amélioré, base du processus de standardisation de VVC), Galpin *et al.* [6] introduisent un CNN analysant la texture d'un bloc, luma ou chroma, pour prédire les divisions possibles des sous-

blocs. Dans VVC, Li *et al.* [7] proposent un CNN à sorties multiples avec un mécanisme de sortie précoce pour déterminer le partitionnement d'une CU. Tissier *et al.* [8] s'appuient sur [6], en remplaçant les heuristiques manuelles par un arbre de décision. Feng *et al.* [9] proposent un CNN Down-Up pour prédire une carte de partitionnement d'un bloc d'entrée de taille 64×64 en configuration intra VVC. Contrairement à [3, 5, 6, 7, 8, 9], notre approche repose sur un entraînement basé sur la régression.
Concernant le second groupe, dans VVC, Qiang *et al.* [10] utilisent un algorithme pour QTMTT basé sur le gradient dérivé de l'opérateur de Scharr pour la description de la texture et la différence des bords des sous-blocs pour l'information structurelle. Dans HEVC, Na *et al.* [11] ont développé un classificateur de terminaison précoce entraîné à l'aide des trajectoires des décisions CU à différentes profondeurs, grâce à un algorithme RL acteur-critique de bout en bout, rendant le classificateur indépendant de la profondeur. Dans [12], l'apprentissage par renforcement profond (DRL) pour optimiser le processus de décision des CUs de taille 32×32 dans VVC est introduit.

3 Approche proposée

3.1 Motivations

Dans VVC, pour une slice intra, la plus grande CU est 128×128, initialement divisée par QT en 4 CUs 64×64. Les décisions de partitionnement sur les CUs de taille 32×32 sont les plus cruciales en terme de compromis entre la vitesse d'encodage et les performance de compression. Pour les CUs relativement grandes, le partitionnement est souvent QT, tandis que pour les CUs relativement petites, le nombre de combinaisons à tester est inférieur à celui des CUs de taille 32×32. Cette étude se concentre donc sur la prédiction des partitionnements des CUs 32×32. Contrairement aux travaux précédents, notre approche prédit une variable calculée avant la décision de partitonnement, en l'occurrence les coûts RD normalisés. Cela offre une flexibilité accrue pour décider la stratégie de partitionnement. Par exemple, les différences de coûts RD des différents partitionnements peuvent indiquer quels partitionnements explorer. L'apprentissage par régression sur les coûts RD reflète mieux la continuité des évolutions des coûts RD selon les variations de texture et des paramètres de codage, par rapport à une tâche de classification. La Fig. 1 illustre la continuité des évolutions des coûts RD en fonction du QP. Elle montre l'évolution du coût RD en fonction du QP pour chaque mode de partitionnement d'un bloc de taille 32×32 sélectionné aléatoirement. À faible QP, le meilleur mode est QT. À QP élevé, BTV dépasse QT. Enfin, à QP très élevé, le mode No Split (NS) devient optimal.

3.2 Conception et entraînement

La méthode proposée utilise les CNNs pour accélérer la RDO dans les codecs vidéo. Un modèle d'apprentissage profond est appliqué à des CUs de taille 32×32 pour prédire 6 valeurs de coût RD, correspondant aux modes

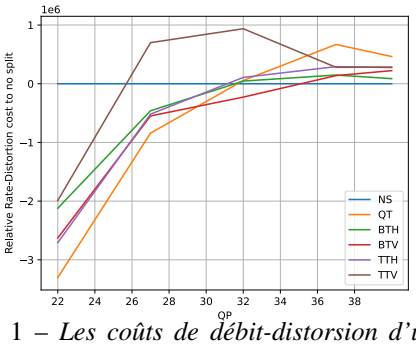

FIGURE 1 – *Les coûts de débit-distorsion d'une CU de taille* 32×32 *pour chaque mode de partitionnement à différents paramètres de quantification. Le coût RD du mode NS est soustrait à tous les coûts RD.*

de partitionnement {NS, QT, BTH, BTV, TTH, TTV}. Ce modèle est intégré du côté encodeur pour éviter l'évaluation des partitionnements non pertinents, accélérant ainsi le processus de RDO. La figure 2 illustre le schéma de cette approche, basée sur une architecture CNN inspirée de ResNet [13], avec des couches convolutionnelles utilisant ReLU comme fonction d'activation, et des maxpoolings. Le bloc de convolution (*ConvBlock*) comprend 4 couches avec une connexion shortcut, comme montré en Fig. 2(b). Après la dernière couche de maxpooling, le modèle concatène la valeur QP avec le vecteur résultant, produisant 6 coûts RD normalisés pour chaque mode.

La base de données d'apprentissage est générée en encodant les séquences BVI-DVC [14] avec VTM 18.0 selon deux configurations : la RDO par défaut et la RDO avec les heuristiques de partitionnement désactivées pour les blocs 32×32 (testant tous les modes de partitionnement). Chaque encodage utilise quatre QPs : 22, 27, 32 et 37. Avec 800 séquences, cela génère une base de données équilibrée comprenant plus de 6 millions de patches (1 million par mode de partitionnement). Chaque patch 36×36 x_i^0, englobant le bloc lui-même et une bordure causale de 4 pixels de large. De plus, la QP, notée x_i^1, de la CU d'indice i est extraite. Le vecteur de vérité terrain y_i correspond à la sélection des coûts RD déterminée par le processus RDO appliqué à la CU 32×32 d'indice i. y_i stocke 6 valeurs, un coût RD pour chaque mode de partitionnement testé.

La normalisation des valeurs de vérité terrain dans les tâches de régression continue apparaît comme une étape cruciale. Ainsi, dans notre travail, pour l'exemple de l'indice i, la vérité terrain $y'_{i,j}$ est un vecteur de 6 coûts RD normalisés par le coût RD NS $y'_{i,0}$, voir Eq. (1).

$$y'_{i,j} = \frac{y_{i,j}}{y_{i,0}}, j \in [|0, 5]. \quad (1)$$

j indexe les 6 modes de partitionnement. Pour l'exemple de l'indice i, $y_{i,j}$ désigne le coût RD original. La fonction f_θ, paramétrée par θ, définit le CNN. f_θ prend x_i^0 et x_i^1 pour calculer une prédiction $\tilde{y}_i$ des 6 coûts RD normalisés,

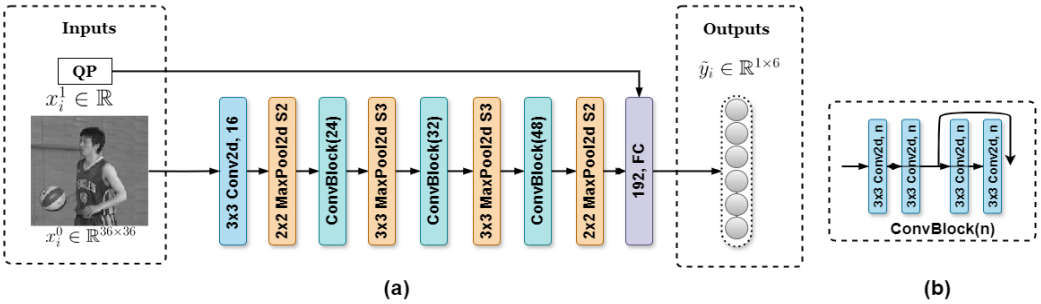

FIGURE 2 – *(a) Aperçu de l'architecture du modèle CNN. (b) Détails des couches ConvBlock*

voir Eq. (2).

$$\tilde{y_i} = f_\theta(x_i^0, x_i^1) \tag{2}$$

Pendant l'apprentissage, une fonction de perte $\mathcal{L}$ calcule l'erreur entre le vecteur prédit des coûts RD et le vecteur des coûts RD normalisés de vérité terrain, moyennée sur tous les exemples, plus un terme de régularisation de la norme L2 des poids W du CNN, $W \subset \theta$, voir Eq. (3).

$$\mathcal{L}\left(\mathcal{D};\theta\right) = \frac{1}{N}\sum_{i=0}^{N-1}(y_i' - f_\theta(x_i^0, x_i^1))^2 + \lambda\sum_{k=1}^{K}W_k^2 \tag{3}$$

N désigne le nombre total d'exemples dans l'ensemble d'apprentissage $\mathcal{D} = \{(x_0^0, x_0^1, y_0'), ..., (x_{N-1}^0, x_{N-1}^1, y_{N-1}')\}$. K désigne le nombre total de poids du CNN. Le CNN est entraîné dans le framework d'apprentissage profond PyTorch. L'optimiseur Adam est utilisé avec un taux d'apprentissage de 10^{-4}, un facteur de régularisation λ de 10^{-5}, et une taille de lot de 256. L'entraînement du modèle s'étend sur 10 itérations et se termine au bout de 4 heures, utilisant un seul GPU (NVIDIA Tesla V100 16GB).

4 Résultats expérimentaux

La méthode est comparée avec VTM 18.0 en considérant deux configurations principales, avec et sans heuristiques, et plusieurs profondeurs de MTT, voir Fig. 3, qui correspondent à plusieurs points de complexité de l'encodeur. Les valeurs 1, 2, 3 et 4 correspondent à la profondeur maximale d'hiérarchie MTT des blocs luma. Dans la première configuration, nous utilisons l'ancrage de base de VTM avec les heuristiques par défaut activées pour les CUs de taille 32×32. Cela nous permet de construire notre base de données en utilisant les séquences BVI-DVC [14], comme expliqué dans la Section 3. De plus, un CNN est entraîné avec une profondeur MTT par défaut de 3. Cependant, en activant ces heuristiques, VTM peut parfois sauter certains modes de partitionnement, ce qui crée des valeurs de coût manquantes dans nos données d'apprentissage. Pour résoudre ce problème, les valeurs manquantes sont remplacées par la valeur maximale multipliée par une constante α (par exemple $\alpha = 2$). Ce mécanisme de pénalité incite le modèle à prédire des coûts plus élevés pour les partitionnements non explorés.

Dans la deuxième configuration, sans heuristiques pour les CUs de 32×32, VTM évalue tous les modes de partitionnement, assurant des données d'apprentissage complètes. Cela permet de comparer l'impact des heuristiques sur les prédictions de coût du modèle CNN. La même architecture

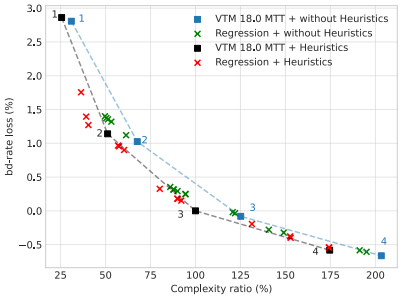

FIGURE 3 – *Comparaison du ratio BD-rate par rapport au gain en vitesse en complexité en All Intra (AI) avec trois configurations de profondeur MTT avec/ou sans activation des heuristiques pour les CUs de taille* 32×32.

illustrée dans la Fig. 2 est utilisée pour entraîner les modèles des deux configurations avec les ensembles de données appropriés. Le modèle avec heuristiques activées est intégré dans la première configuration, et celui sans heuristiques dans la deuxième, après conversion de torch à SADL [15]. L'implémentation SADL permet une intégration autonome dans le codec.

Lors de l'inférence, pour chaque CU 32×32, les modèles prédisent 6 coûts RD et les modes de partitionnement à vérifier sont sélectionnés. Les coûts prédit sont comparés à un seuil ajustable, ce qui donne une perte en BD-rate et un gain en vitesse par rapport à l'ancrage. Les séquences de test des Conditions de Test Communes de JVET [16] évaluent notre approche, incluant toutes les classes (A à F). La réduction de complexité est calculée comme la différence du nombre total de pixels traités avec et sans le modèle :

$$\Delta C = \frac{1}{4}\sum_{QP_i\in22,27,32,37}\frac{T_a(QP_i) - T_p(QP_i)}{T_a(QP_i)} \tag{4}$$

où $T_p(QP_i), T_a(QP_i)$ sont les complexités avec et sans le modèle intégré, respectivement.

Fig. 3 montre le ratio d'accélération par rapport à la perte en BD-rate des deux configurations avec et sans les mo-

dèles à trois configurations MTT. La ligne en pointillés noire représente la première configuration avec heuristiques activées, tandis que la ligne en pointillés bleue représente la deuxième configuration sans activation des heuristiques. Les croix rouges correspondent à notre approche avec heuristiques activées, et les croix vertes à notre méthode sans activation des heuristiques. Notre solution permet d'atteindre des compromis intermédiaires par rapport à une profondeur MTT de base.

5 Conclusion

Cet article propose une approche pour accélérer la recherche exhaustive de la RDO dans VTM-18.0 en utilisant la régression des coûts de débit-distorsion. Un CNN prédit les coûts de débit-distorsion pour tous les modes de partitionnement disponibles, facilitant ainsi le choix optimal du mode de partitionnement. Les expériences montrent que l'entraînement du CNN avec un ensemble de données complet, sans heuristiques (assurant la disponibilité de tous les coûts), améliore l'efficacité et permet d'atteindre des compromis supplémentaires en termes de complexité lors de l'inférence. En combinant ces modèles CNN avec diverses configurations, nous accélérons efficacement le processus de RDO à différentes profondeurs de MTT.

Annexe

La méthode est publiée dans IEEE ICASSP 2024 [17].

Références

[1] Benjamin Bross, Ye-Kui Wang, Yan Ye, Shan Liu, Jianle Chen, Gary J. Sullivan, et Jens-Rainer Ohm. Overview of the versatile video coding (vvc) standard and its applications. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(10) :3736–3764, 2021.

[2] Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, et Thomas Wiegand. Overview of the high efficiency video coding (hevc) standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12) :1649–1668, 2012.

[3] Mai Xu, Tianyi Li, Zulin Wang, Xin Deng, Ren Yang, et Zhenyu Guan. Reducing complexity of hevc : A deep learning approach. *IEEE Transactions on Image Processing*, 27(10) :5044–5059, 2018.

[4] Debargha Mukherjee, Jingning Han, Jim Bankoski, Ronald Bultje, Adrian Grange, John Koleszar, Paul Wilkins, et Yaowu Xu. A technical overview of vp9 – the latest open-source video codec. Dans *SMPTE 2013 Annual Technical Conference & Exhibition*, pages 1–17, 2013.

[5] Somdyuti Paul, Andrey Norkin, et Alan C. Bovik. Speeding up vp9 intra encoder with hierarchical deep learning-based partition prediction. *IEEE Transactions on Image Processing*, 29 :8134–8148, 2020.

[6] Franck Galpin, Fabien Racapé, Sunil Jaiswal, Philippe Bordes, Fabrice Le Léannec, et Edouard François. Cnn-based driving of block partitioning for intra slices encoding. Dans *2019 Data Compression Conference (DCC)*, pages 162–171, 2019.

[7] Tianyi Li, Mai Xu, Runzhi Tang, Ying Chen, et Qunliang Xing. Deepqtmt : A deep learning approach for fast qtmt-based cu partition of intra-mode vvc. *IEEE Transactions on Image Processing*, 30 :5377–5390, 2021.

[8] Alexandre Tissier, Wassim Hamidouche, Souhaiel Belhadj Dit Mdalsi, Jarno Vanne, Franck Galpin, et Daniel Menard. Machine learning based efficient qtmtt partitioning scheme for vvc intra encoders. *IEEE Transactions on Circuits and Systems for Video Technology*, 33(8) :4279–4293, 2023.

[9] Aolin Feng, Kang Liu, Dong Liu, Li Li, et Feng Wu. Partition map prediction for fast block partitioning in vvc intra-frame coding. *IEEE Transactions on Image Processing*, 32 :2237–2251, 2023.

[10] Qiang, Hui Li, Ya Meng, et Li. Texture-based fast qtmt partition algorithm in vvc intra coding. *Signal, Image and Video Processing*, 17(4) :1581–1589, 2023.

[11] Na Li, Yun Zhang, Linwei Zhu, Wenhan Luo, et Sam Kwong. Reinforcement learning based coding unit early termination algorithm for high efficiency video coding. *Journal of Visual Communication and Image Representation*, 60 :276–286, 2019.

[12] Jinchao Zhao, Yihan Wang, Mingying Li, et Qiuwen Zhang. Fast coding unit size decision based on deep reinforcement learning for versatile video coding. *Multimedia Tools Appl.*, 81(12) :16371–16387, May 2022.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, et Jian Sun. Deep residual learning for image recognition, 2015.

[14] Di Ma, Fan Zhang, et David R. Bull. BVI-DVC : A training database for deep video compression. *IEEE Transactions on Multimedia*, 24 :3847–3858, 2022.

[15] Franck Galpin, Pavel Nikitin, Thierry Dumas, et Philippe Bordes. SADL small adhoc deep-learning library. Rapport technique JVET-W0181, InterDigital, Juillet 2021.

[16] Karczewicz Marta et Ye Yan. Common test conditions and evaluation procedures for enhanced compression tool testing. *WG 05 MPEG Joint Video Coding Team(s) with ITU-T SG 16,* 30^{th} *meeting, Antalya*, June, 2023.

[17] M. E. A. Kherchouche, F. Galpin, T. Dumas, D. Menard, et L. Zhang. Rd-cost regression speed up technique for vvc intra block partitioning. Dans *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3530–3534, 2024.