UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Luiz Henrique De Lorenzi Cancellier

An Asymmetric Multi-Layer Visual Signal Compression Approach Combining Handcrafted and Learned Solutions

Florianópolis
2025

Luiz Henrique De Lorenzi Cancellier

An Asymmetric Multi-Layer Visual Signal Compression Approach Combining Handcrafted and Learned Solutions

Tese submetida ao Programa de Pós-Graduação em Ciência da Computação para a obtenção do título de Doutor em Ciência da Computação.
Orientador: Prof. José Luís Almada Güntzel, Dr.
Coorientadores: Prof. Luís Alberto da Silva Cruz, Dr. e Prof. Mateus Grellert da Silva, Dr.

Florianópolis
2025

Ficha catalográfica gerada por meio de sistema automatizado gerenciado pela BU/UFSC.
Dados inseridos pelo próprio autor.

Cancellier, Luiz Henrique De Lorenzi
An Asymmetric Multi-Layer Visual Signal Compression Approach Combining Handcrafted and Learned Solutions / Luiz Henrique De Lorenzi Cancellier ; orientador, José Luís Almada Güntzel, coorientador, Luís Alberto da Silva Cruz, coorientador, Mateus Grellert da Silva, 2025.
116 p.

Tese (doutorado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Ciência da Computação, Florianópolis, 2025.

Inclui referências.

1. Ciência da Computação. 2. Ciência da Computação. 3. Video coding. 4. Learned compression. 5. Scalable compression. I. Güntzel, José Luís Almada. II. Cruz, Luís Alberto da Silva. III. Silva, Mateus Grellert da IV. Universidade Federal de Santa Catarina. Programa de Pós Graduação em Ciência da Computação. V. Título.

Luiz Henrique De Lorenzi Cancellier

An Asymmetric Multi-Layer Visual Signal Compression Approach Combining Handcrafted and Learned Solutions

O presente trabalho em nível de doutorado foi avaliado e aprovado por banca examinadora composta pelos seguintes membros:

Prof. Guilherme Ribeiro Corrêa, Dr.
Universidade Federal de Pelotas

Prof. Vanessa Testoni, Dr[a].
Universidade Estadual de Campinas

Prof. Mauro Roisenberg, Dr.
Universidade Federal de Santa Catarina

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi julgado adequado para obtenção do título de Doutor em Ciência da Computação.

ICP-Edu
Documento assinado digitalmente
Marcio Bastos Castro
Data: 04/10/2025 16:36:04-0300
CPF: ***.707.720-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Márcio Bastos Castro , Dr.
Coordenador do Programa

ICP-Edu
Documento assinado digitalmente
Jose Luis Almada Guntzel
Data: 04/10/2025 14:05:54-0300
CPF: ***.780.670-**
Verifique as assinaturas em https://v.ufsc.br

Prof. José Luís Almada Güntzel, Dr.
Orientador

Florianópolis, 2025.

Este trabalho é dedicado a todos que me recebem e são recebidos como família.

ACKNOWLEDGEMENTS

Pouco é mencionado no documento da tese, mas a execução deste trabalho consistiu muito mais em decisões ruins e fracassos do que em acertos e sucessos. Ao longo de toda essa jornada, eu felizmente sempre pude contar com o apoio de muitas pessoas. A começar pela minha família. Menciono aqui meus pais Enio e Eunice, e meu irmão Gabriel, mas extendo os meus agradecimentos a todos os avós, tios e primos. Tive a felicidade de nascer em uma família muito unida e toda a minha jornada acadêmica sempre recebeu o suporte dela.

Agradeço ao meu orientador José Luís Güntzel, não só por toda a paciência e dedicação ao longo deste trabalho, mas também por ter aceitado e apoiado minhas escolhas de explorar temas tão desafiadores. Além disso, se hoje eu prezo tanto pela qualidade em meus trabalhos, é porque esse valor sempre esteve presente em sua orientação. Também agradeço aos meus coorientadores Mateus Grellert e Luís Cruz, por terem me dado tanto apoio durante a experiência de intercâmbio, que tanto marcou a minha vida e mudou minha visão de mundo, e aos membros da banca, pela participação em uma defesa muito construtiva.

Acredito que é arrogante pensar que uma pesquisa se faz sozinho. Quando utilizo a primeira pessoa do plural nessa tese, me refiro não só aos orientadores mas também incluo a presença dos meus colegas de laboratório Ismael Seidel, André Bräscher e a Gabriela Silveira. Nossa equipe sempre foi pequena, mas sempre muito colaborativa na troca de ideias e no aprofundamento de conhecimentos. Muitos dos avanços dessa tese foram frutos das nossas interações em workshops internos.

Em todo lugar onde estive ao longo deste trabalho, sempre pude contar com o apoio de amigos muito queridos. Menciono aqui o Tiago Fontana, a Cecília Contreras, a Rafaela Ludwinsky, o Gabriel Arthur Gerber, o Bruno Zavarise, o Soma Hatházi, a Maria Ester e o Kaan Akalp. Aos colegas de laboratório, que são também grandes amigos Marleson Graf, Sheiny Fabre, Renan Netto, Bernardo Ferrari, Alexandre Santana, Bruno Bonotto e Bruno Miranda. A Andréia Kmita, o Daniel Floriani, o Danish Khan, o Jorge Villaroel, a Laura Sousa, o Makhles, a Mainara Tesser, o Saman e o Wellington Oliveira. Como existe uma alta probabilidade de eu ter esquecido o nome de alguém, fique à vontade para cobrar a rodada para brindar quando nos encontrarmos novamente.

Também registro aqui os meus agradecimentos a todos os professores com os quais tive a oportunidade de ter aulas. Admiro aqueles que se dedicam a educar com qualidade.

Por fim, mas não menos importante, deixo meus agradecimentos aos técnicos que fizeram o acompanhamento do processo de intercâmbio da bolsa PRINT/CAPES. Ir para terras estrangeiras é um grande desafio e agradeço por todo o apoio e esclarecimentos prestados por vocês, que facilitaram este processo. O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001.

“Eu vivo sempre no mundo da lua
Porque sou um cientista
O meu papo é futurista
E lunático”
(Guilherme Arantes – Lindo Balão Azul)

RESUMO

As soluções utilizando Redes Neurais (RN) têm ganhado popularidade em diversos domínios científicos e aplicações nos últimos anos. Os primeiros trabalhos de compressão de vídeo por aprendizado concentraram-se na substituição de algoritmos utilizados em etapas específicas da codificação híbrida por implementações baseadas em NN. Mais recentemente, modelos de compressão totalmente baseados em aprendizagem foram desenvolvidos, alcançando resultados de eficiência de compressão semelhante ou superiores àqueles alcançados por padrões convencionais, os quais têm sido otimizados por décadas. Entretanto, até o presente momento, poucos estudos exploraram a combinação de soluções baseadas em aprendizagem e em algoritmos convencionais em uma estrutura de compressão em múltiplas camadas, abordagem esta investigada neste trabalho.

A metodologia proposta aproveita as vantagens de ambos os métodos ao aplicá-los em camadas apropriadas: o *bitstream* da camada base, gerado por um codec convencional, mantém conformidade com um padrão consolidado, enquanto que a camada de aprimoramento utiliza os espaços latentes para melhorar a compressão e reconstrução. Além disso, o modelo proposto, nomeado Compressor Multi-Camadas Assimétrico – *Asymmetric Multi-layer compressor* (AMLC), desacopla as camadas base e de aprimoramento na etapa de codificação, permitindo simplificações da codificação na camada de aprimoramento, que é aprendida de ponta-a-ponta.

Experimentos com uma implementação parcial do *framework*, incorporando diferentes soluções de *upscaling* espacial e temporal, revelaram que o uso exclusivo de upscaling espacial apresenta resultados mais próximos aos da compressão convencional em camada única. Consequentemente, optou-se pela implementação do *framework* para codificação quadro-a-quadro.

O AMLC codifica uma imagem 1,28 vezes mais rapidamente do que um codec multi-camadas totalmente aprendido, apresentando eficiência de codificação semelhante, mesmo utilizando uma camada base convencional que é menos eficiente em termos de compressão. O codificador assimétrico aprendido, em conjunto com a camada base convencional, reduz a complexidade geral, enquanto a camada de aprimoramento aprendida melhora a eficiência de codificação.

Os resultados experimentais demonstram que o AMLC supera a eficiência de codificação do software de referência SHVC, um codec convencional multi-camadas. Em uma segunda análise, observou-se que a inclusão das dependências entre camadas na etapa de codificação produziu resultados semelhantes ao AMLC, confirmando que a adoção das simplificações adotadas pelo AMLC não causa perdas significativas de eficiência de codificação.

Palavras-chave: Otimização ponta-a-ponta, compressão de sinais visuais, compressão aprendida, compressão multi-camadas, escalabilidade.

RESUMO ESTENDIDO

Introdução

O processo de codificação de vídeos é fundamental para que estes sejam armazenados e transmitidos por usuários nos mais diferentes dispositivos. A crescente demanda por *streaming* e o aumento das resoluções são um incentivo para a criação de novos padrões de codificação, que buscam melhorar cada vez mais a eficiência de codificação, fornecendo vídeos com melhor relação entre qualidade e taxas de bits.

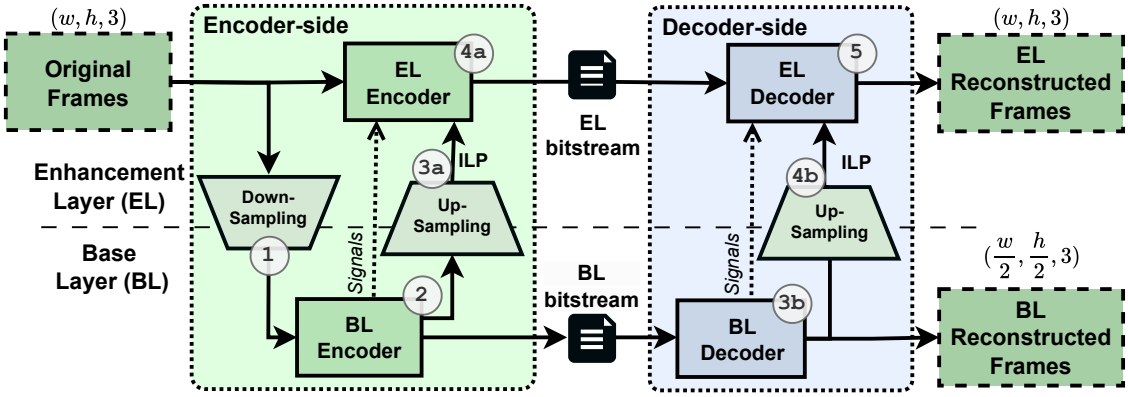
Enquanto os padrões convencionais continuam evoluindo com ferramentas projetadas manualmente para melhorar a eficiência de codificação, os avanços em redes neurais e em plataformas de hardware com elevados graus de paralelismo têm impulsionado o desenvolvimento de soluções baseadas em redes neurais para processamento de imagem e vídeo. Inicialmente aplicadas em tarefas como super-resolução espacial e interpolação de quadros, essas soluções passaram a ganhar destaque também na compressão de vídeo. Apesar de serem mais recentes, os métodos neurais de compressão já apresentam eficiência comparável aos padrões convencionais mais modernos, com maior flexibilidade e capacidade de adaptação a diferentes métricas de qualidade e aplicações específicas.

As soluções de compressão, tanto aprendidas quanto convencionais, geralmente seguem o modelo de camada única, apresentado na figura 1, no qual um codificador comprime o sinal original em um bitstream padronizado e um decodificador reconstrói o sinal a partir desse bitstream. Trabalhos mais recentes passaram a explorar também o modelo de compressão em múltiplas camadas, apresentado na figura 2. Nesse modelo, a mídia é inicialmente reduzida em resolução e comprimida por uma camada base. A camada de aprimoramento então utiliza informações da camada base para melhorar a reconstrução

Figura 1 – Modelo de compressão em uma única camada. O codificador mantém a resolução da entrada em sua saída.

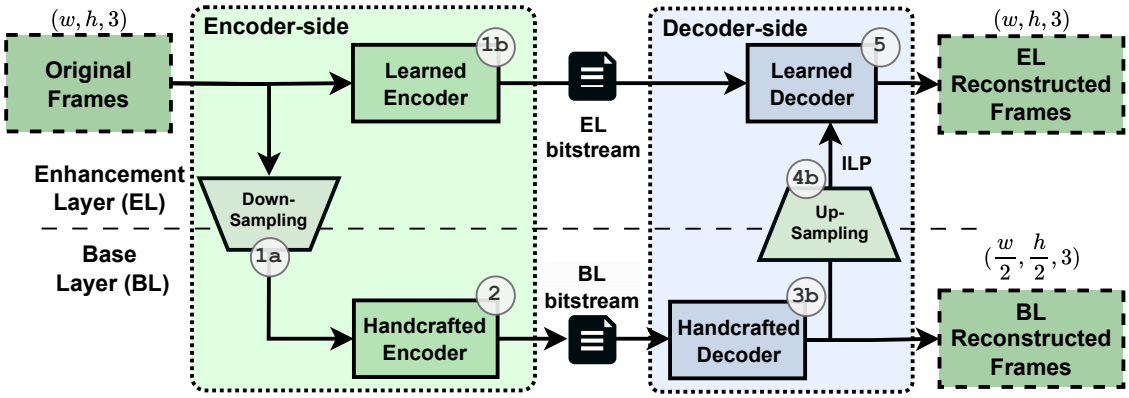
Fonte: the author (2025).

Figura 2 – Modelo de compressão com múltiplas camadas. Além da versão de mesma resolução que a original, o codificador também é capaz de reproduzir a mídia em outras resoluções mais baixas.



Fonte: the author (2025).

Figura 3 – Modelo proposto. Foi adotado um codificador convencional como camada base e um codificador aprendido como camada de aprimoramento, removendo as dependências entre camadas no processo de codificação.



da mídia em resolução original.

Apesar dos avanços nas redes neurais, a adoção de modelos totalmente aprendidos em dispositivos com restrições energéticas ainda é limitada devido à falta de suporte de hardware. Como alternativa, este trabalho propõe uma solução híbrida: a camada base utiliza um codec convencional, garantindo compatibilidade com padrões existentes, enquanto que a camada de aprimoramento aplica técnicas aprendidas para melhorar a reconstrução final. Além disso, a proposta permite que a codificação da camada base e de aprimoramento ocorra em paralelo, como ilustrado na figura 3, eliminando dependências entre as camadas no processo de codificação. No lado da decodificação, a estrutura segue a lógica tradicional de múltiplas camadas, garantindo a eficiência da reconstrução.

Objetivos

O objetivo principal deste trabalho é projetar e avaliar uma estrutura de compressão em múltiplas camadas que combine componentes convencionais e aprendidos, equilibrando desempenho e complexidade. Para alcançar esse objetivo sem depender de elementos sintáticos de padrões existentes, isolamos o padrão convencional em um codec de camada base e introduzimos modelos de redes neurais para o codec de camada de aprimoramento. Os objetivos específicos deste trabalho são:

- Avaliar o impacto na qualidade do processo de reamostragem espacial em um cenário de compressão de vídeo com perdas;
- Propor e analisar um modelo para o problema de reamostragem temporal em um cenário de compressão de vídeo com perdas;
- Explorar diferentes combinações de soluções de reamostragem espacial e temporal para melhorar as taxas de compressão de um padrão tradicional de codificação de vídeo híbrido;
- Propor e avaliar combinações de camada base convencional com camada de aprimoramento aprendida, analisando sua eficiência de codificação e complexidade em comparação com codecs multi-camadas totalmente convencionais e totalmente aprendidos.

Metodologia

O primeiro passo consistiu na execução de uma revisão sistemática da literatura, que resultou na elaboração de uma taxonomia para a classificação dos trabalhos na área de

codificação de vídeos com modelos de redes neurais. Através de refinamentos na revisão da literatura, foi possível identificar trabalhos correlatos, além de outros que auxiliaram posteriormente na definição dos componentes da arquitetura do modelo AMLC.
Na sequência, diferentes componentes para reamostragem espacial e temporal foram testados em um cenário de compressão de vídeo com perdas, sem a utilização da camada de aprimoramentos. Para a reamostragem temporal, em particular, foi proposto um modelo de interpolação de quadros utilizando métodos de predição de fluxo óptico. Das análises desta versão parcial do modelo, foi tomada a decisão de continuar a pesquisa com modelos de compressão quadro-a-quadro, utilizando somente a reamostragem espacial. Os detalhes do método para elaboração, treinamento e avaliação do modelo proposto serão detalhados na sequência.
Modelos de referência Como codec convencional de múltiplas camadas, foi utilizado o software de referência do Scalable HEVC (SHVC) (SHM12.4) na configuração All-Intra. Para comparação com um codec totalmente aprendido (*fully-learned*, ou FL-codec), foi adotado o modelo Cheng2020 (Cheng et al., 2020) em modo *simulcast*, conforme descrito por Mei et al. (2022). As curvas taxa-distorção (RD) foram construídas configurando as camadas base e de aprimoramento com os mesmos níveis de qualidade. Para o SHVC, foram utilizados os valores de parâmetro de quantização (QP) 27, 32, 37 e 42, enquanto para o FL-codec utilizaram-se os níveis 1, 3, 4 e 6.
Datasets O dataset Vimeo90k (Xue et al., 2019) foi utilizado para treinamento e avaliação. Cada amostra foi convertida de RGB para YUV420 e, posteriormente, reconvertida para RGB utilizando o FFmpeg, sem interpolação dos canais de crominância. Esse processo garante duas propriedades fundamentais: 1) Evita o acúmulo de erro nas conversões entre YUV420 e RGB; 2) Impede que o modelo se beneficie da representação de cores em RGB que não poderiam ser reproduzidas de forma equivalente em YUV420. Além do Vimeo90k, o modelo proposto foi avaliado utilizando os conjuntos de validação dos datasets CLIC-Professional e CLIC-Mobile (Toderici et al., 2020). Foi realizada a operação de *cropping* para garantir que todas as amostras tivessem dimensões múltiplas de 64, evitando a necessidade de preenchimento (*padding*) nos codecs.
Camada Base. As imagens dos datasets foram convertidas e utilizadas para treinar a compressão da camada de aprimoramento. No entanto, foi necessário pareá-las com reconstruções da camada base. Para isso, os quadros YUV420 foram redimensionados para 50% do tamanho original com interpolação bicúbica (via FFmpeg), e posteriormente comprimidos com o High Efficiency Video Coding (HEVC) Test Model (HM-16.4) na configuração All-Intra, utilizando os valores de QP 22, 27, 32 e 37. O treinamento foi realizado separadamente para cada QP, gerando quatro conjuntos de dados distintos, o que possibilitou avaliar como a camada de aprimoramento lida com diferentes alvos de taxa-distorção na camada base.
Instâncias do Modelo. Foram utilizados os pesos pré-treinados disponibilizados pelo CompressAI para o modelo de Cheng2020. Embora esse modelo forneça seis níveis de compressão, quatro foram selecionados para o estudo: dois dentre os de mais alta taxa de compressão (níveis 1 e 3) e dois dentre os de mais alta qualidade (níveis 4 e 6) Os níveis 1 e 3 utilizam 128 kernels, enquanto os níveis 4 e 6 utilizam 192 kernels. Todos os modelos foram configurados com $N = 8$ no bloco Merge.
Parâmetros de treinamento. O treinamento foi conduzido com recortes de resolução 256×256, com 12 por batch e o otimizador Adam. A taxa de aprendizado inicial foi de 1e-4, sendo reduzida para 1e-5 na segunda etapa de treinamento. Como métrica de distorção, foi utilizado o erro quadrático médio (MSE).
Métricas de avaliação. As comparações foram realizadas em termos de eficiência de

codificação, medida tanto em função da taxa de bits quanto da qualidade. A taxa de bits, medida em bits por pixel, calculada considerando a soma dos tamanhos dos arquivos das bitstreams de cada camada. Para a avaliação da qualidade das imagens reconstruídas, foram utilizadas seis métricas objetivas distintas: Peak signal-to-noise ratio (PSNR)-YUV (sendo o canal Y ponderado por 6, enquanto os canais U e V são ponderados por 1 cada), PSNR-Y, PSNR-UV, PSNR-RGB, PSNR-HVS e Multiscale Structural Similarity Index Measure (MS-SSIM).

Resultados e Discussão

A Tabela 1 apresenta os resultados de eficiência de codificação da solução Asymmetric Multi-layer Compressor (AMLC), em comparação com o codec SHVC. A solução AMLC apresenta ganhos significativos de eficiência de codificação em relação ao SHVC, demonstrando os benefícios de se utilizar uma camada de aprimoramento baseada em modelos aprendidos. É notável que os maiores ganhos, em todos os conjuntos de dados, ocorrem nos componentes UV. Ainda assim, os ganhos em relação ao SHVC não se limitam apenas às métricas PSNR-UV, PSNR-RGB e MS-SSIM, mas também se estendem para PSNR-Y e PSNR-HVS.

O estudo apresentado por Mei et al. (2022) demonstrou que treinar e avaliar um modelo aprendido diretamente em YUV420 não é tão promissor quanto realizar a pesquisa no domínio RGB: a proposta deles apresentou resultados ligeiramente inferiores ao codec SHVC, especialmente para a métrica PSNR-Y. Neste trabalho, o domínio YUV420 é relevante pois codecs de vídeo convencionais normalmente operam e são otimizados nesse formato. Embora não tenham sido feitos treinamento diretamente em YUV420 como sugerido por Mei et al. (2022), foi garantida a conversão sem perdas entre YUV420 e RGB, e vice-versa. Dessa forma, o treinamento é realizado no domínio RGB, mas sem perder dados devido à subamostragem de croma do YUV420. Os ganhos observados tanto nos componentes de croma (UV) quanto de luminância (Y) em relação ao SHVC indicam que essa abordagem de treinamento é uma alternativa viável ao treinamento direto em YUV420.

Os padrões HEVC e SHVC introduziram ferramentas específicas para melhorar a eficiência de codificação de amostras com resoluções mais altas do que aquelas presentes no Vimeo90k (Sullivan et al., 2012). O modelo AMLC, por outro lado, foi treinado apenas com quadros de baixa resolução do conjunto Vimeo90k. Por esse motivo, o SHVC tende a apresentar melhores resultados nos datasets CLIC. Ainda assim, a solução AMLC supera o SHVC em todos os casos, com uma melhoria de até -6.47% na taxa Bjøntegaard Delta (BD)-rate, mesmo no pior cenário.

A camada base convencional apresenta menor eficiência de codificação em relação a camada base aprendido, com diferença de até 33,68% e -1,5dB no pior caso. No entanto, como apresentado na Tabela 2, observa-se que a solução AMLC entrega resultados de eficiência de codificação similares ou ligeiramente superiores ao FL-codec, considerando diferentes métricas de qualidade e combinações de conjuntos de dados. Tomando como exemplo a avaliação PSNR-RGB no conjunto CLIC-Professional, a solução proposta perde apenas 0,6% e 0,03dB em relação ao FL-codec. Esse é um resultado notável, considerando a desvantagem do Base Layer (BL) convencional em termos de eficiência de codificação.

Em relação a complexidade, os experimentos mostraram que o codec convencional da camada base, adotado pelo AMLC, é significativamente mais rápido do que os modelos aprendidos. Ele é pelo menos 2× mais rápido na codificação e 257× mais rápido na decodificação em comparação ao modelo totalmente aprendido (Cheng2020). Ao considerar a solução completa de compressão multi-camada, o codec SHVC (totalmente convencional)

Tabela 1 – Resultados médios de BD do codificador AMLC em relação ao SHVC.

Quality Metrics	Vimeo90k		CLIC-Mobile		CLIC-Professional	
	BD-Quality (dB)	BD-Rate (%)	BD-Quality (dB)	BD-Rate (%)	BD-Quality (dB)	BD-Rate (%)
PSNR-YUV	1.69	-27.86	0.75	-18.06	0.85	-20.10
PSNR-Y	1.14	-17.05	0.27	-6.47	0.41	-8.06
PSNR-UV	3.66	-62.41	2.40	-56.71	2.55	-62.42
PSNR-RGB	1.99	-34.19	0.89	-20.60	1.04	-25.67
PSNR-HVS	1.14	-16.96	0.39	-7.76	0.41	-8.20
$\text{MS-SSIM}_{\text{dB}}$	1.98	-26.57	1.26	-22.56	1.35	-23.91

Tabela 2 – Resultados médios de BD do codificador AMLC em relação ao FL-codec.

Quality Metrics	Vimeo90k		CLIC-Mobile		CLIC-Professional	
	BD-Quality (dB)	BD-Rate (%)	BD-Quality (dB)	BD-Rate (%)	BD-Quality (dB)	BD-Rate (%)
PSNR-YUV	0.16	-3.61	0.14	-3.61	0.12	-2.90
PSNR-Y	0.07	-0.80	0.08	-1.40	0.09	-1.15
PSNR-UV	0.73	-13.04	0.56	-14.60	0.51	-12.76
PSNR-RGB	0.09	-2.73	0.01	0.31	-0.03	0.57
PSNR-HVS	0.04	-0.36	0.14	-2.47	0.08	-0.96
MS-SSIM	0.58	-8.08	0.68	-12.95	0.62	-11.27

supera tanto o AMLC quanto a solução totalmente aprendida (FL-codec) em termos de complexidade. No pior cenário, o SHVC foi 1,13× mais rápido na codificação e 205,6× mais rápido na decodificação em vídeos do Vimeo90k, chegando a ser 328× mais rápido para decodificar em 1080p.

Por um lado, a perda era esperada em relação ao SHVC, considerando que o modelo aprendido adotado como camada de aprimoramento no AMLC prioriza pela eficiência de codificação e não pela complexidade. Por outro, a abordagem proposta, que combina a camada base convencional e a camada de aprimoramento aprendida, mostra ganhos de desempenho em relação a solução totalmente aprendida: ela é pelo menos 1,28× mais rápida para codificar e 1,27× mais rápida para decodificar, destacando a vantagem no uso do camada base convencional.

Considerações Finais

O modelo proposto supera o software de referência SHVC em termos de eficiência de codificação, com um ganho de BD-Rate de -62,42%. Mesmo utilizando uma camada base convencional e sem dependências entre camadas durante a codificação, o modelo alcança uma eficiência semelhante à de soluções totalmente aprendidas com múltiplas camadas e conexões residuais. Isso destaca a vantagem de usar uma compressão assimétrica baseada em aprendizado na camada de aprimoramento. Os ganhos foram confirmados por meio de diferentes métricas de qualidade e conjuntos de dados.

Palavras-chave: Otimização ponta-a-ponta, compressão de sinais visuais, compressão aprendida, compressão multi-camadas, escalabilidade.

ABSTRACT

Neural Network (NN) solutions have gained popularity in many scientific domains and applications in recent years. Early works on learned solutions for video compression focused on replacing specific handcrafted algorithms with NN-based implementations. More recently, fully learned compression models have been developed, achieving similar or better compression efficiency than handcrafted standards, which have been optimized for decades. However, few studies have explored combining learned and handcrafted solutions in a multi-layer compression framework, the way we explored in this research. The approach proposed in this thesis leverages the advantages of both methods by employing them on appropriate layers: the Base Layer (BL) bitstream, generated by a handcrafted codec, remains compliant with a well-established standard, while the Enhancement Layer (EL) utilizes learned latent spaces to improve the compression and reconstruction. Additionally, the proposed AMLC model decouples the base and enhancement layers on the encoding stage, enabling an end-to-end learned enhancement layer that simplifies the coding process. Experiments conducted with a partial implementation of the proposed framework, incorporating different spatial and temporal upscaling solutions, revealed that using only spatial upscaling yields results closest to those from handcrafted single-layer compression. Consequently, the framework was implemented for single-frame encoding. The experimental results showed that AMLC encodes an image 1.28 times faster than a fully learned multi-layer codec while maintaining similar coding efficiency, despite relying on a less coding-efficient handcrafted base layer. The asymmetric learned encoder and handcrafted BL reduce the overall complexity, while the learned enhancement layer improves coding efficiency. The results also show that AMLC outperforms the coding efficiency of SHVC reference software, a handcrafted multi-layer codec. Furthermore, an alternative implementation of the proposed model, incorporating inter-layer dependency during encoding, produces results that are similar to AMLC, confirming its potential to simplify multi-layer compression without significant losses in coding efficiency.

Keywords: End-to-end optimization, visual signal compression, learned compression, multi-layer compression, scalability.

LIST OF FIGURES

Figure 1 – Single-layer compression. 23
Figure 2 – Multi-layer compression. 24
Figure 3 – Simplified version of AMLC 25
Figure 4 – Simplified hybrid model for video compression. 28
Figure 5 – A frame representation. 29
Figure 6 – Sequence of frames with POC, references and types identified. 29
Figure 7 – Possible PU partitioning in HEVC. 30
Figure 8 – HEVC partitioning examples and its quadtree structure. 30
Figure 9 – Possible block partitions in VVC 31
Figure 10 – VVC partitioning example and its quadtree multi-type tree structure. . 32
Figure 11 – Block diagram of Prediction step. 33
Figure 12 – Intra reference samples on HEVC standard. 34
Figure 13 – HEVC intra prediction modes example. 34
Figure 14 – Integer Motion Estimation example. 36
Figure 15 – Fractional sample domain. 36
Figure 16 – CABAC model. 39
Figure 17 – Example of RD curves. 40
Figure 18 – Artificial neuron. 42
Figure 19 – Multilayer perceptron neural network. 42
Figure 20 – Convolution neural network structure. 43
Figure 21 – Convolution operation. 43
Figure 22 – Convolution neural network feature dimensions. 44
Figure 23 – Pooling operation. 44
Figure 24 – Residual learning. 45
Figure 25 – Autoencoder model. 46
Figure 26 – Proposed taxonomy for the literature works. 49
Figure 27 – Learned image codec. 72
Figure 28 – Implementation of the AMLC model without the EL. 75
Figure 29 – The proposed Frame Interpolation Module. 77
Figure 30 – The Residual Block Module used in the Frame Interpolation Module. . 78
Figure 31 – Quality results distributions when using FlowNet2 and FlowNet2S. . . 79
Figure 32 – RD curves of *FourPeople* video sequence. 82
Figure 33 – Y MS-SSIM distributions of each experiment and QP 82
Figure 34 – Bitrate reduction relative to the *x265* experiment. 83
Figure 35 – Proposed framework (AMLC). 84
Figure 36 – Proposed framework decoder 85
Figure 37 – Merge block 86
Figure 38 – Merge block components 86

Figure 39 – Proposed framework RD results 90
Figure 40 – Results on Vimeo90k samples 95
Figure 41 – Results on CLIC-Professional image patches 96
Figure 42 – Rate-Distortion Curves A. 102
Figure 43 – Rate-Distortion Curves B. 103
Figure 44 – Rate-Distortion Curves C. 104
Figure 45 – Rate-Distortion Curves D. 105
Figure 46 – Rate-Distortion Curves E. 106

LIST OF TABLES

Table 1 – Video sizes. 22
Table 2 – Block partitioning related work results. 52
Table 3 – Intra mode decision related work results. 55
Table 4 – Static resolution scaling related work results. 57
Table 5 – Quality enhancement related work results. 59
Table 6 – Intra sample prediction related work results. 61
Table 7 – Inter sample prediction related work results. 62
Table 8 – Fractional sample prediction related work results. 64
Table 9 – Loop operations related work results. 66
Table 10 – Dynamic resolution scaling related work results. 70
Table 11 – Main Characteristics of Reviewed Works on Multi-layer compression and Ours. 74
Table 12 – Description of symbols used in Chapter 5 75
Table 13 – Experiments labels and configurations. 80
Table 14 – CTC video sequences used for RD-Analysis. 81
Table 15 – Description of symbols used in Chapter 6 84
Table 16 – λ Training Parameters for Each Combination of BL QP and Enhancement Layer (EL) Level. 89
Table 17 – Average BD results comparing the AMLC against the SHVC. 91
Table 18 – Average Handcrafted BL Coding Efficiency Loss Compared to the Learned One. 92
Table 19 – Average BD Results Comparing the AMLC Against the FL-codec. . . . 92
Table 20 – Average AMLC Coding Efficiency Results on Vimeo90k Compared to an Equivalent Implementation Using Residual Connections at the Encoding-side. 93
Table 21 – Average Encoding and Decoding Times (seconds) on Vimeo90k. 93
Table 22 – Average Encoding and Decoding Times on 1920×1080 resolution frames in Seconds. 94

LIST OF SYMBOLS

$\mathbf{B}^{\text{can}}$	Candidate block.
$\mathbf{B}^{\text{ori}}$	Original block.
d	Distortion..
j_{cost}	The lagrangian rate-distortion cost of selecting a given candidate as reference.
λ	The Lagrange multiplier..
$\mathbf{F}^{\text{ori}}$	Original frames.
$\mathbf{F}^{\text{rec}}$	Reconstructed frames.
$\mathbf{F}^{\text{u}}$	Up-sampled frames.
r	Rate..
$\mathbf{S}^{\text{ori}}$	Spatial frames.
$\mathbf{S}^{\text{rec}}$	Reconstructed spatial frames.
$\mathbf{S}^{\text{d}}$	Spatial sub-sampled frames.
$\mathbf{S}^{\text{d'}}$	Decoded Spatial Sub-sampled frames.
$\mathbf{S}^{\text{u}}$	Spatial Up-sampled frames.
$\mathbf{T}^{\text{ori}}$	Temporal frames.
$\mathbf{T}^{\text{rec}}$	Reconstructed temporal frames.
$\mathbf{T}^{\text{u}}$	Interpolated temporal frames.

LIST OF ABBREVIATIONS AND ACRONYMS

AMLC Asymmetric Multi-layer Compressor ... 11, 12, 14, 16, 24–26, 48, 74, 75, 84–86, 88–94, 96–98
AMVP Advanced Motion Vector Prediction 36
AVC Advanced Video Coding 22, 38

BD Bjøntegaard Delta 11, 40, 89, 91, 92
BD-Rate Bjøntegaard Delta Bitrate 40, 51, 52, 54–64, 66, 69–71
BL Base Layer 11, 16, 23–26, 73, 75, 79, 80, 85, 87–94, 97

CABAC Context-Adaptive Binary Arithmetic Coding 38, 39, 69
CNN Convolutional Neural Network 41, 43, 45, 50–53, 55–71
CPU Central Processing Unit 67
CTC Common Test Conditions 50, 58, 80
CTU Coding Tree Unit 29–32, 37, 53, 67
CU Coding Unit 29–32, 38, 39, 51, 53, 54

DCT Discrete Cosine Transform 37, 65
DST Discrete Sine Transform 37

EL Enhancement Layer 16, 23–26, 73, 81, 84–91, 94, 97, 98

FME Fractional Motion Estimation 35, 37, 65
fps frames per second 51
FSRCNN Fast Super-Resolution Convolutional Neural Network 80

GOP Group of Pictures 28, 60, 67, 68
GPU Graphics Processing Unit 67

HDR High Dynamic Range 22
HEVC High Efficiency Video Coding .. 10, 11, 22, 27–31, 34–39, 52, 53, 55, 57, 59, 62, 64, 67–69, 71, 73, 79, 80, 85, 91, 93
HM HEVC Test Model 39, 51, 54, 64
HVS Human Visual System 39, 46, 89

ILP Interlayer Prediction 23–25, 85
IME Integer Motion Estimation 35, 36, 61

JEM Joint Exploration Model 51

KLT Karhunen-Loève Transform 37

LCEVC Low Complexity Enhancement Video Coding 73
LSTM Long Short-Term Memory 60

ME Motion Estimation 35–37, 60
MLP Multilayer Perceptron 42, 52, 53, 55, 56, 61, 62, 65, 68, 71
MS-SSIM Multiscale Structural Similarity Index Measure 11, 39, 40, 46, 78–80, 86, 89, 91–93
MSE Mean Squared Error 39, 46, 86, 87, 89, 92
MV Motion Vector 35–37, 61, 69

NN Neural Network . 23–26, 41, 42, 45, 46, 48–51, 54, 56, 58, 60, 61, 63, 65, 66, 69–72, 97, 99

POC Picture Order Count 28
PSNR Peak signal-to-noise ratio 11, 39, 40, 68, 89, 91, 92
PU Prediction Unit 30–33, 35, 36, 38, 39, 52

QP Quantization Parameter..... 37, 39, 40, 52, 56, 58, 61, 67, 69–71, 79, 81, 82, 88, 90
QTBT Quadtree Binary-Tree..... 31, 53, 54
QTMT Quadtree Multi-type Tree..... 31

RD Rate-Distortion..... 73, 87–90
RDO Rate-Distortion Optimization..... 32, 35, 49–52, 54–56, 60, 61, 63, 68

SHVC Scalable HEVC 10–12, 16, 23, 73, 87, 88, 90, 91, 93, 94
SR Super-Resolution 23, 85
SVM Support-Vector Machine 48, 99

TU Transform Unit 31

UVG Ultra Video Group..... 80

VFI Video Frame Interpolation..... 23, 85
VTM VVC Test Model 39, 51
VVC Versatile Video Coding..... 22, 23, 27, 28, 31, 32, 35, 37, 39, 53, 63, 73, 80, 85

CONTENTS

1	**INTRODUCTION**	**22**
1.1	PROBLEM IDENTIFICATION	24
1.2	HYPOTHESIS AND GOALS	25
1.2.1	**Specific Goals**	**25**
1.3	CONTRIBUTIONS	26
1.4	DOCUMENT ORGANIZATION	26
2	**VIDEO COMPRESSION CONCEPTS**	**27**
2.1	A SEQUENCE OF IMAGES	27
2.2	BLOCK PARTITIONING	28
2.2.1	**Quadtree Partitioning in HEVC**	**30**
2.2.2	**Quadtree Multi-type Tree Partitioning in VVC**	**31**
2.3	RATE DISTORTION OPTIMIZATION MODEL	32
2.4	THE PREDICTION	33
2.4.1	**Intra Mode**	**33**
2.4.2	**Inter Mode**	**35**
2.5	TRANSFORM, QUANTIZATION AND RECONSTRUCTION	37
2.6	ENTROPY ENCODING	38
2.7	CODING EFFICIENCY	38
3	**NEURAL NETWORK CONCEPTS**	**41**
3.1	NEURAL NETWORK	41
3.2	CONVOLUTION NEURAL NETWORK	43
3.2.1	**Other NN structures**	**45**
3.3	NEURAL NETWORK TRAINING	45
4	**SYSTEMATIC LITERATURE REVIEW**	**48**
4.1	METHOD	48
4.2	PROPOSED TAXONOMY	49
4.2.1	**Complexity Reduction Works**	**51**
4.2.2	**Coding Efficiency Works**	**56**
4.2.2.1	*Non-normative Approach*	*56*
4.2.2.2	*Normative Approach*	*61*
4.2.3	**Learning-based Coding Models**	**71**
4.3	SINGLE- AND MULTI-LAYER COMPRESSION	71
4.3.1	**Multi-Layer Compression**	**73**

5 BASE-LAYER COMPRESSION AND SPATIAL-TEMPORAL UPSAMPLING 75
5.1 SPATIAL UPSAMPLING 76
5.2 TEMPORAL UPSAMPLING 77
5.3 EXPERIMENTAL SETUP 79
5.4 RESULTS AND DISCUSSIONS 81

6 ASYMMETRIC MULTI-LAYER CODEC 84
6.1 AMLC MODEL 85
6.1.1 Two Stage Training and Loss Functions 86
6.2 EXPERIMENTAL SETUP 87
6.2.1 Experimental Configuration 87
6.2.2 Evaluation metrics 89
6.3 EXPERIMENTAL RESULTS 90
6.3.1 RD Evaluation 90
6.3.2 Coding Efficiency Comparisons 91
6.3.3 Complexity 93
6.3.4 Visual Comparisons 94

7 CONCLUSIONS 97
7.1 FUTURE WORKS 97

APPENDIX A – SEARCH STRINGS 99
A.1 VIDEO AND IMAGE COMPRESSION USING NN SOLUTIONS . . 99
A.2 VIDEO COMPRESSION USING NN SOLUTIONS 100

APPENDIX B – RATE-DISTORTION CURVES 102

BIBLIOGRAPHY 107

1 INTRODUCTION

The evolution of mobile technology allows us to watch videos on devices that fit in our pockets. At the same time that videos can be watched everywhere, they also expose us to a broad range of content available worldwide through the internet. While 73% of all internet traffic was video in 2016 (Cisco, 2017), Cisco (2019) reports that such a share rose to 75% in 2017 and was estimated to reach 82% by 2022. According to Ericsson (2022), video will account for approximately 80% of global mobile data traffic by 2028.

In addition to the increasing demand for video streaming, people have become used to technological improvements in digital video. Nowadays, even smartphones can reproduce 4k (3840x2160) resolution and High Dynamic Range (HDR) videos. Storing or transmitting video data in raw form is unfeasible. This is presented in Table 1, showing that a full HD video at 30 FPS requires a 1.49 Gbps connection to be transmitted and a 5-minute video needs 55.99 GB for storage.

Table 1 – Video resolutions and their respective bitrates and file sizes. We consider videos with 30 frames per second, 3 bytes per pixel, and 5 minutes long, in raw format.

Resolution (pixels)	Bitrate (Gbps)	Size (GB)
HD (1280×720@30fps)	0.66	24.88
Full HD (1920×1080@30fps)	1.49	55.99
Ultra HD (3840×2160@30fps)	5.97	223.95
Full Ultra HD (7680×4320@30fps)	23.88	895.79

To enable significant data reduction for video representation, hybrid compression standards and handcrafted algorithms have been developed over the years. The compressed video is a bitstream that must follow a standard so that different devices compatible with that standard can later decompress and reproduce the video. The Advanced Video Coding (AVC) (ITU-T, 2017) standard was proposed in 2003 to improve HD video compression. Compared to H.263, the AVC coder reduces the bitrate by about 50% while keeping the same perceptive visual quality (Wiegand et al., 2003). It is important to notice that a video coding standard restricts the set of compression tools, which may not achieve the desired compression rates for higher video resolutions. The advent of Ultra and Full Ultra HD formats motivated the development of a new video coding standard called High Efficiency Video Coding (HEVC) to achieve higher compression rates. Once again, the new standard coder allows for reducing the bitrate by 50% while maintaining the same perceptive visual quality as the AVC reference software (Sullivan et al., 2012). Versatile Video Coding (VVC) is the current state-of-the-art standard from ISO/ITU-T, and it also achieved the mark of reducing 50% bitrate under similar perceptive visual quality relative to its predecessor, the HEVC (Bross et al., 2021a). Each new standard extends the computing cost of the coding tools to improve coding efficiency over previous standards. Consequently, the incremental improvements in coding efficiency have come

at the cost of significantly higher complexity, raising concerns about the sustainability of this approach. Nevertheless, the development of a new video coding standard is already under consideration, with ongoing MPEG activities investigating solutions beyond VVC (MPEG, 2025).

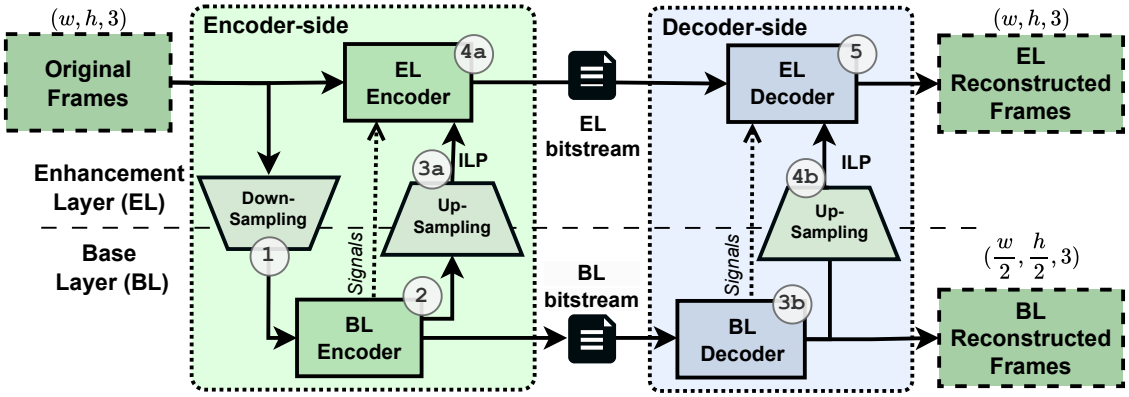
Meanwhile, the advancements in Neural Network (NN) technology and the hardware improvements that made it possible to train larger models have promoted an increasing research effort on learned solutions for image and video processing tasks. At first, spatial Super-Resolution (SR) (Dong et al., 2014; Liang et al., 2022; Chen et al., 2023) and Video Frame Interpolation (VFI) (Niklaus; Liu, 2018; Niklaus; Liu, 2020; Huang et al., 2022) tasks explored NN technology to improve the state-of-the-art results. After Ballé, Laparra and Simoncelli (2016a) presented a solution to the non-differentiable quantization problem, further research has been conducted to improve the learned image and video compression with more sophisticated models (Ballé; Laparra; Simoncelli, 2016b; Minnen; Ballé; Toderici, 2018; Lee; Cho; Beack, 2019; Cheng et al., 2020). Despite the shorter development history of learned compression solutions, which only gained popularity after 2016 (Ballé; Laparra; Simoncelli, 2016a), they are already on par with recent handcrafted standards in coding efficiency (Bégaint et al., 2020). NNs offer flexibility, such as the choice of loss function, making it easy to adapt to different quality metrics. Moreover, they are well-suited for specific content applications, like security camera footage, and opens possibilities for other applications, such as compression tailored for machine-to-machine communication.

The aforementioned learned and handcrafted compression works follow the single-layer model from Figure 1: an encoder (`A`) receives the original signal and produces a standardized compressed bitstream, then a decoder (`B`) parses that bitstream and runs the necessary steps to reconstruct the signal. Recent works expand the learned solutions to employ multi-layer compression (Su et al., 2020; Mei et al., 2022) (Figure 2) using concepts and techniques inherited from scalable, handcrafted codecs like the Scalable HEVC (SHVC) (Boyce et al., 2016). As shown in figure 2, two or more layers perform the compression in this approach. The original media is first downsampled (`1`) and then compressed by a Base Layer (BL) (`2`), which is usually a single-layer codec. An inter-layer processing step transforms the BL reconstructed media into an Interlayer Prediction (ILP) (`3a`) to be used by the Enhancement Layer (EL) (`4a`). In the model shown in figure 2, an upsampling method produces the ILP. If allowed by the standard, additional BL signals

Figure 1 – Single-layer image compression model. Both the input and output images have three channels, width w and height h.

Source: the author.

Figure 2 – Multi-layer image compression model. In addition to the EL output, the BL is capable of reproducing the image at half its original resolution.



Source: the author.

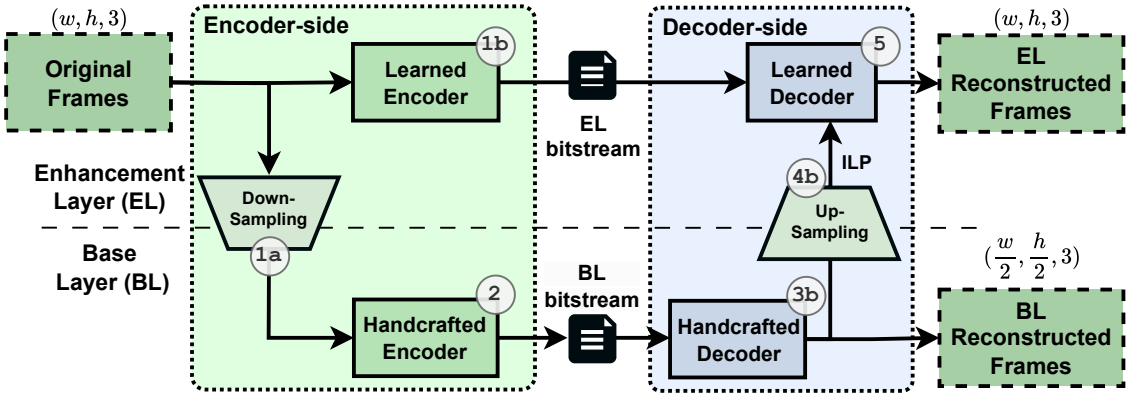
(e.g., motion vectors) may be passed along to improve compression. On the decoding side, the BL decoder reconstructs the media in a lower resolution (3b). Reflecting the structure of the encoder side, the upsampling produces the ILP (4b), which the EL decoder uses to reconstruct the media at its original resolution (5).

1.1 PROBLEM IDENTIFICATION

As mentioned before, the NN advancements are promoting the rise of new and different applications. In a short time span, learned compression achieved state-of-the-art coding efficiency results compared to handcrafted codecs, which have been optimized for decades. Multi-layer codecs are designed to provide functionalities that are not provided by single-layer codecs (Mei et al., 2022). The most straightforward one is the capability of efficiently generating bitstreams for different resolutions and/or quality levels, so that the reconstructed media achieves a level that satisfies the transmission or device requirements. Although it is possible to find fully learned multi-layer solutions in the literature (Su et al., 2020; Mei et al., 2022), they might bring a prohibitive overhead in pipelines that employ handcrafted codecs, since there is limited hardware support for this technology on user devices such as smartphones.

Instead of a fully learned or a handcrafted multi-layer solution, the technique proposed in this thesis, named Asymmetric Multi-layer Compressor (AMLC), combines the benefits of both solutions by employing them to appropriate layers: 1) the BL bitstream produced by a handcrafted codec, which is compliant with a video coding standard, while 2) on top of that, the EL can be used for exploring learned latent spaces to enhance the media reconstruction. Furthermore, we take advantage of the end-to-end training to remove dependencies between the EL and the ILP extracted from BL at the encoding side, as depicted in figure 3. Therefore, the steps necessary for BL encoding (1a and 2) and EL encoding (1b) can be executed in parallel. In this thesis, we adopt the term asymmet-

Figure 3 – Proposed AMLC model. We adopt a handcrafted-based BL and a learned-based EL, removing the ILP dependency between those layers at the encoding side.



ric to refer to the fact that the decoder's inter-layer dependencies are not 'mirrored' on the encoder side, which contrasts with the approach typically adopted by scalable coding standards (Boyce et al., 2016; Battista et al., 2022). In the asymmetric approach, the EL codec is trained as an end-to-end model, optimizing the encoder to transmit relevant information for the final image reconstruction. Although we removed the ILP at the EL encoding side, that prediction is still available at the decoding side, where all the steps follow the same structure as the ones depicted in figure 2.

1.2 HYPOTHESIS AND GOALS

This thesis work has the following hypothesis: **Combining handcrafted and learned compression models in an asymmetric multi-layer codec can improve coding efficiency compared to a fully-handcrafted multi-layer codec, and can also reduce complexity compared to a fully-learned approach while maintaining compatibility with existing handcrafted standards at the base layer.** In this thesis, we test the hypothesis using HEVC and SHEVC as handcrafted codecs, while describing the model generically as a framework intended to support future experiments with different standards.

Therefore, the primary goal of this work is to design and evaluate a multi-layer compression model combining handcrafted and learned components to balance performance and complexity. To achieve that, we isolate the handcrafted standard in aBL codec and introduce the NN models for the EL codec.

1.2.1 Specific Goals

1. Evaluate the quality impact of spatial upsampling processes in a lossy video compression scenario.
2. Propose and assess a model for the temporal upsampling problem in a lossy video compression scenario.

3. Exploit different combinations of spatial and temporal upsampling solutions to improve the compression rates of a traditional hybrid video coding standard.
4. Propose and evaluate the combinations of handcrafted BL and learned EL, assessing their coding efficiency and complexity compared to relevant fully-handcrafted and fully-learned multi-layer codecs.

1.3 CONTRIBUTIONS

This thesis brings the following contributions and novelties:

1. **It establishes a taxonomy for NN-based video compression methods.** By providing a structured view of the field, this taxonomy helps researchers understand the design space of learned compression models and situates video compression.
2. **It explores the impact of spatial-temporal upsampling solutions for video compression.** Usually, both spatial and temporal upsampling solutions are focused on the upsampling process by itself, working with near-lossless compressed media. In this work, we not only evaluate different combinations of spatial-temporal solutions, but also consider a broad range of compression rates and quality results in our experiments.
3. **It proposes a new multi-layer compression framework.** The asymmetric solution takes advantage of the learned compression solution to remove inter-layer dependencies at the encoding side without a significant impact on coding efficiency compared to a symmetric solution. Such an approach was not found in previous related proposals.
4. **It proposes a scalable compression solution leveraging a handcrafted BL.** By integrating a carefully designed BL, the framework not only achieves scalability across different compression rates but also takes advantage of well-established platforms and existing hardware implementations, facilitating practical deployment.

1.4 DOCUMENT ORGANIZATION

Chapter 2 covers the basic video coding concepts. Chapter 3 covers the basic neural network concepts. Chapter 4 presents a systematic review on NN-based solutions for video compression, proposing a taxonomy to classify the works found in the literature. Chapter 5 covers the specific goals 1, 2, and 3, presenting an implementation of the proposed solution without the EL, exploring different spatial upsampling methods and proposing a temporal upsampling solution. Chapter 6 covers the specific goal 4, presenting the AMLC model details and comparisons with other codecs in terms of coding efficiency and complexity. Chapter 7 draws the conclusions and presents future work.

2 VIDEO COMPRESSION CONCEPTS

This chapter explains hybrid video encoding concepts from handcrafted codecs and the method for evaluating video coding efficiency. Video compression concepts rely on Richardson (2010), whereas High Efficiency Video Coding (HEVC) coder details are based on Sullivan et al. (2012) and Sze, Budagavi and Sullivan (2014). Versatile Video Coding (VVC) coder details, in turn, are based on ISO Central Secretary (2020) and ITU-T (2020).

Video coding standards adopt the **hybrid model** depicted in Figure 4. It is called hybrid because it mixes operations that work on the spatial domain – such as the prediction step – with others that work on the frequency domain – transform and quantization. The encoding process begins by parsing the video into frames and dividing each frame into blocks, referred to as original blocks. During the prediction step, the encoder uses previously encoded frame data to select a reference block for representing the current block. The difference (residual) between the original and the reference blocks is then transformed and quantized.

All information required for reconstruction is converted into a bitstream. This bitstream is further compressed using an entropy encoder, which removes statistical redundancies among symbols. For reconstruction, the quantized residual undergoes inverse quantization and inverse transformation, and is then added to the reference block. The reconstructed block is filtered to reduce encoding artifacts before being assembled into the reconstructed frame. Within a frame, reconstructed blocks can be used as references for the prediction of other blocks in the same frame. Once an entire frame is reconstructed, it may be stored in a buffer and fed back into the encoding loop, allowing the encoder to use it as reference data for the prediction of subsequent frames. In the following sections, we describe each step of the hybrid model.

2.1 A SEQUENCE OF IMAGES

A video is a sequence of **frames** (temporal samples), and each frame is an array of **pixels** (spatial samples), as depicted in Figure 5. A pixel's color can be defined by a single number (grayscale images) or a set of numbers (color images). Each number used to represent the color is called a **color component**, or just **sample**, and the frame representation in terms of samples is called **channel**. Each sample is represented using a fixed bit-length, referred to as **bit-depth**.

Video compression typically adopts the **YCbCr** color format, also known as YUV. YCbCr separates image brightness – represented by a single luma channel (Y) – from color information – carried by two chroma channels (Cb and Cr, or Y and V). YCbCr is convenient for compression because human vision is more sensitive to luminance information. Therefore, it is possible to **sub-sample** the chroma channels to reduce the

amount of data required to represent the video without significant visual quality loss.

The encoder indexes each parsed frame with a Picture Order Count (POC) number that identifies its position within the sequence. Moreover, the encoder assigns each frame a type: **Intra (I)**, **Predicted (P)**, or **Bi-Predicted (B)**, depending on the time domain prediction arrangement. Figure 6 shows configurations using the three frame types. An I frame is encoded using only its own information; hence, the first frame of a sequence is always of type I. Once an I frame is encoded, subsequent frames may be of type P or B. P frames are encoded using a single reference frame list, whereas B frames use two. The differences between the P and B frames will become clearer after the section on the prediction step.

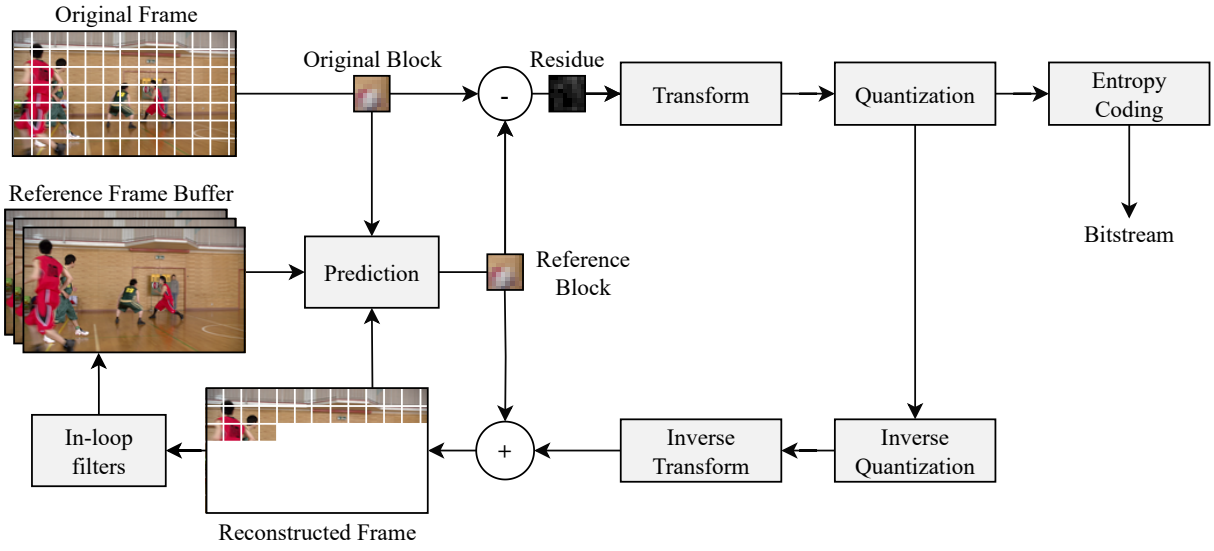
A set of frames that references only other frames within the same set is called **Group of Pictures (GOP)**. To ensure this property, the GOP must contain at least one I frame to serve as a reference for the others. The GOP concept is essential in video streaming. If information is lost during transmission, the affected frame will be reconstructed with errors. In such a scenario, other frames with references to the erroneous one would also be reconstructed with error, causing error propagation. A GOP limits error propagation to its scope, keeping the remaining sequence unaffected.

2.2 BLOCK PARTITIONING

A frame to be encoded is first split into small blocks of samples. Each new standard introduces more flexible block partitioning structures, allowing the encoder to find a partitioning scheme that improves encoding efficiency.

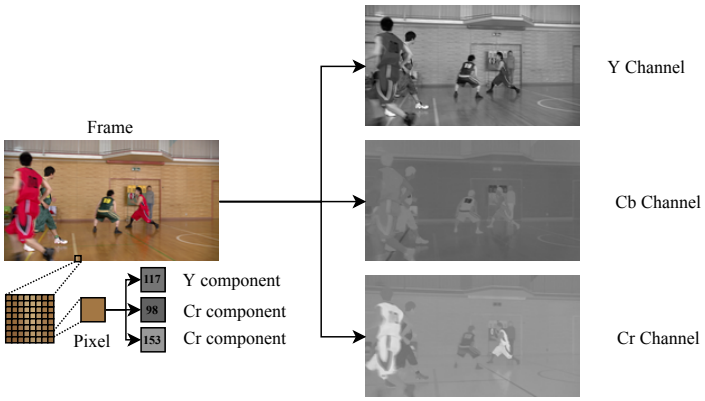
Although HEVC and VVC have different block partitioning structures, they share

Figure 4 – Simplified hybrid model for video compression.



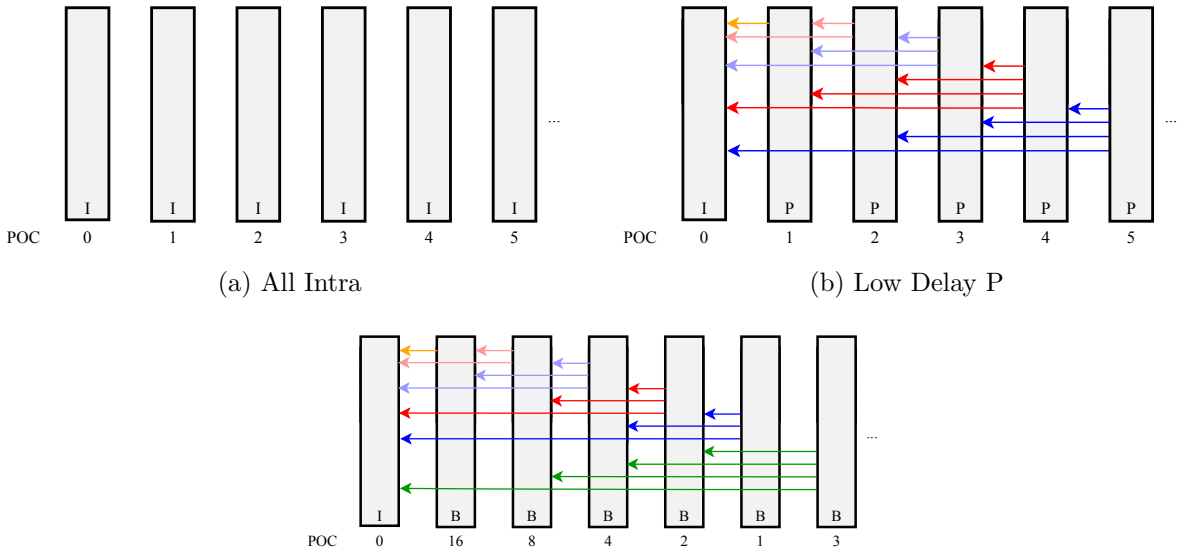
Source: adapted from (Richardson, 2010)

Figure 5 – A frame representation and its three channels.



Source: the author.

some concepts. The encoder first divides a frame into non-overlapping blocks called **Coding Tree Units (CTUs)**. The CTU partitioning is represented as a tree structure, and nodes of that tree are called **Coding Units (CUs)**. CTU, CU are respectively associated with Coding Tree Block (CTB) and Coding Block (CB). Each **Unit** encapsulates three blocks –one for each YCbCr component – and carries split signaling data. For simplicity, we will use the term associated with the Unit when referring to the standard and **block** when referring to a matrix of samples. We explain the peculiarities of each standard in the following subsections.



(a) All Intra

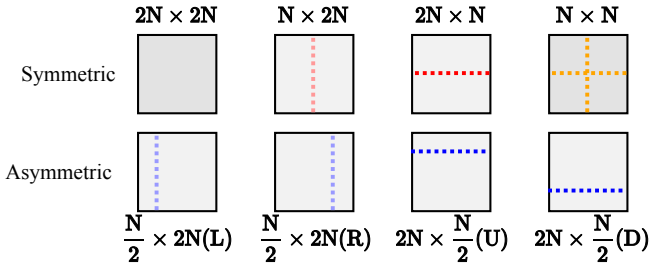
(b) Low Delay P

(c) Random Access

Figure 6 – Sequences of frames based on HEVC reference software under the configurations *All Intra* (a), *Low Delay P* (b) and *Random Access* (c). Each frame is identified by its POC and type. An arrow from frame X to Y indicates that X can use Y as a reference.

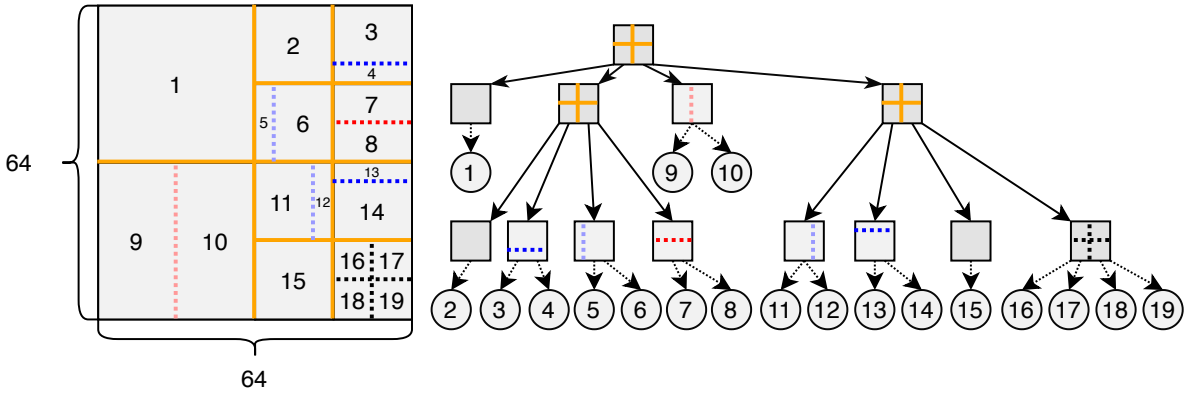
Source: the author.

Figure 7 – Possible PU partitions of a 2N×2N sized CU, where N ∈ {8, 16, 32}. Only symmetric partitions are allowed for N equals 4, and 4×4 PU size is only allowed for intra prediction.



Source: adapted from (Sullivan et al., 2012).

Figure 8 – An example of HEVC CTU and its partitioning scheme. Solid lines represent CU partitions, while dashed lines indicate PU partitions. Only the square nodes are signaled in the tree. The numbers indicate the parsing order.



Source: the author.

2.2.1 Quadtree Partitioning in HEVC

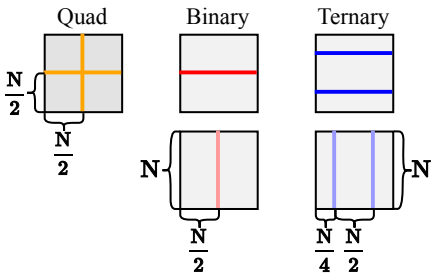
This subsection explains the block representation of HEVC, using Kim et al. (2012) as a reference. A CTU has a maximum size of 64×64 samples and roots a quadtree structure. In a quadtree, each CU node can be split to signal four new square CUs, and the smallest CU is limited to a size of 8×8.

Each CUs is further split into **Prediction Units (PUs)**. There are eight possible PUs, depicted in Figure 7, but some partitioning restrictions apply. Given a 2N×2N CU, the restrictions are as follows:

- Only symmetric partitions are allowed when N equals 4.
- A 4×4 block size is only allowed for intra partitions.
- Intra partitions can only use 2N×2N and N×N; hence, intra prediction is only defined for square blocks.

Figure 8 shows a possible block partitioning according to the HEVC standard.

Figure 9 – Possible partitions in VVC for a block with size N×N.



Source: the author.

2.2.2 Quadtree Multi-type Tree Partitioning in VVC

VVC simplifies some HEVC concepts: there no longer exist the PU or Transform Unit (TU) concepts. However, the nomenclature simplification does not result in simplified partitioning structures. Block partitioning became more flexible, introducing more partitioning options.

CTUs now has a maximum size of 128×128 and the minimum CU size is 4×4. The partitioning structure of VVC is called Quadtree Multi-type Tree (QTMT)[1]. As in HEVC, CTU partitioning in VVC starts with quadtree splits. The difference is that each block from quadtree partitioning may root a multi-type tree structure that allows a combination of binary or ternary splits. There are five partitioning types, depicted in Figure 9, but unlike the PUs, each partition can be further subdivided.

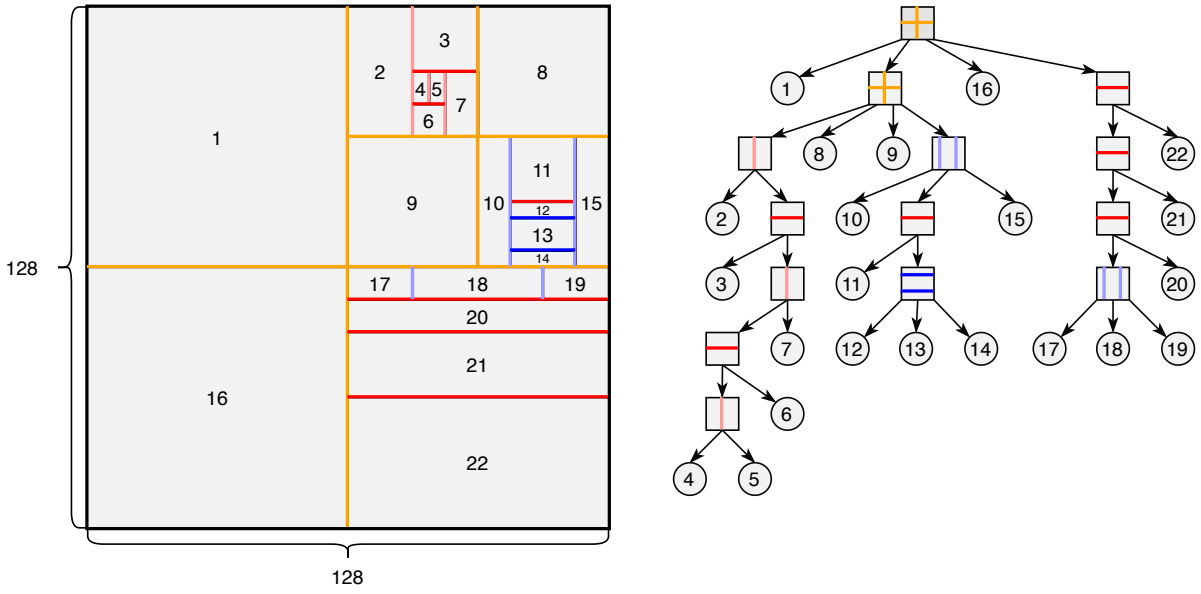
Although the structure is more flexible than the one adopted in HEVC, some restrictions need to be considered:

- When a multi-type split occurs in a branch, it is not allowed to perform new quadtree partitioning in that branch. The Multi-type tree is restricted to binary and ternary splits.
- It is not allowed to have a block with any edge size less than 4.
- A binary split with the same orientation as the previous one is not allowed on the middle block of a ternary split.
- Binary split is not allowed if the CU has one edge greater than 64 and the perpendicular edge is less than or equal to 64.
- Ternary splits are not allowed if any edge of the CU exceeds 64.

Figure 10 illustrates a possible partitioning on VVC standard.

[1] When the explorations for a new standard started, the initial proposal for block partitioning was a Quadtree Binary-Tree (QTBT) structure. It is possible to find related works using this structure, but we will not present details about it because it can be seen as a simplified version of the QTMT, where only quad and binary split are allowed.

Figure 10 – A VVC example of CTU and its partitioning scheme. Only the square nodes are signaled in the tree. The numbers identify the parsing order.



Source: the author.

There are additional restrictions to handle cases when a CTU is partially outside the frame. Another addition was the possibility of partitioning luma and chroma samples separately. Chroma partitioning has even more restrictions, which are out of this work's scope.

2.3 RATE DISTORTION OPTIMIZATION MODEL

In the prediction step, the encoder evaluates different partitioning schemes. The best partitioning scheme is chosen taking into account the prediction cost of all PUs (or leaf CUs in VVC). Therefore, we present the cost calculation in this section and the prediction in the subsequent one.

Video compression aims to reduce the number of bits to represent the video while maintaining similar perceptual quality. To achieve this, the encoder seeks to minimize a cost function that balances the final bitrate and the quality loss relative to the original video. Sullivan and Wiegand (1998) propose the Rate-Distortion Optimization (RDO) model to evaluate the cost. The RDO requires actual bitrate and quality resulting from the encoded video, making its exhaustive use infeasible due to the high computational complexity. To address this issue, encoders typically employ simplified RDO for most evaluations. The simplified RDO is formulated as follows:

$$j_{\text{cost}}(\mathbf{B}^{\text{ori}}, \mathbf{B}^{\text{can}}) = d(\mathbf{B}^{\text{ori}}, \mathbf{B}^{\text{can}}) + \lambda \times r(\mathbf{B}^{\text{ori}}, \mathbf{B}^{\text{can}}) \tag{2.1}$$

where the total cost (j_{cost}) of encoding the original block ($\mathbf{B}^{\text{ori}}$) with the candidate ($\mathbf{B}^{\text{can}}$) is the sum of **distortion** (d) and **rate** (r), weighted by a Lagrange multiplier (λ). The

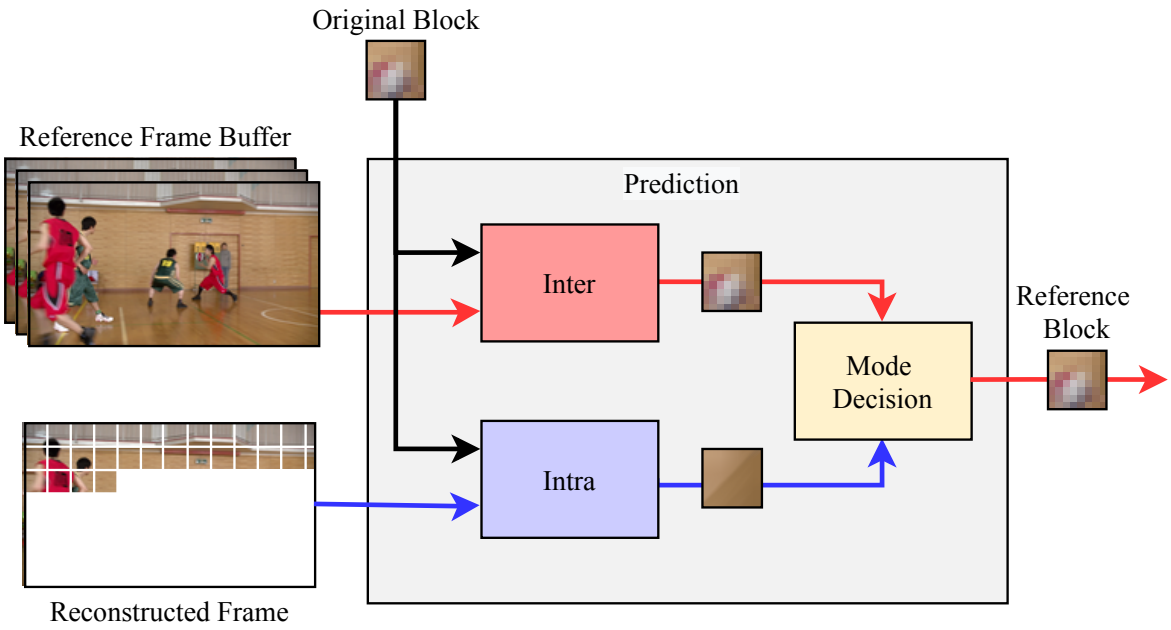
distortion measures the difference between two blocks. If the distortion is zero (the ideal case), the candidate is identical to the original. The rate estimates the number of bits required to signal the decision in the final bitstream. Henceforth, we assume that the optimal candidate in a given search set is the one that minimizes j_{cost}.

2.4 THE PREDICTION

A video contains a large amount of redundant information. Homogeneous textures within a frame, such as solid colors and motion blur, exemplify **spatial redundancies**. Because frames close in time often contain similar content, videos also exhibit **temporal redundancy**. The role of **prediction** is to explore spatial and temporal redundancies by using previously encoded data to represent the block being encoded (original block).

Figure 11 shows the prediction step block diagram. The encoder can predict a PU using either **Intra** or **Inter** modes. While the former generates a set of candidates based on previously encoded neighboring pixels within the same frame, the latter searches for candidates in already encoded frames.

Figure 11 – Block diagram of Prediction step.



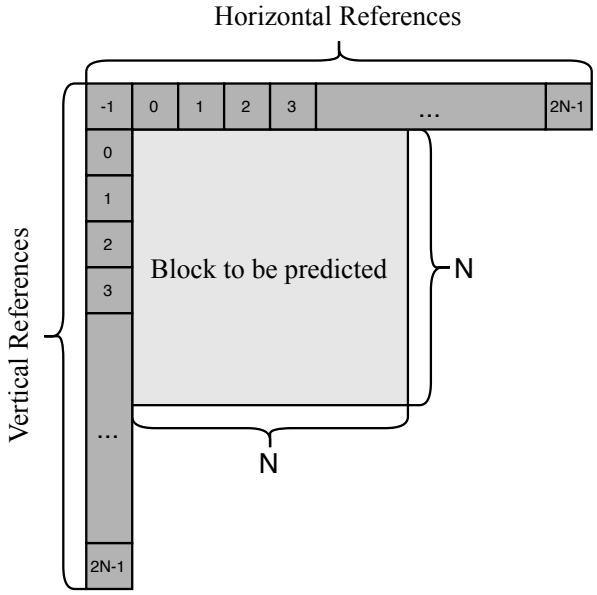
Source: the author.

2.4.1 Intra Mode

The Intra mode relies only on sample references within the frame being encoded to generate candidate blocks. The encoder generates candidates using reference samples from a line above (horizontal references) and a column on the left (vertical references)

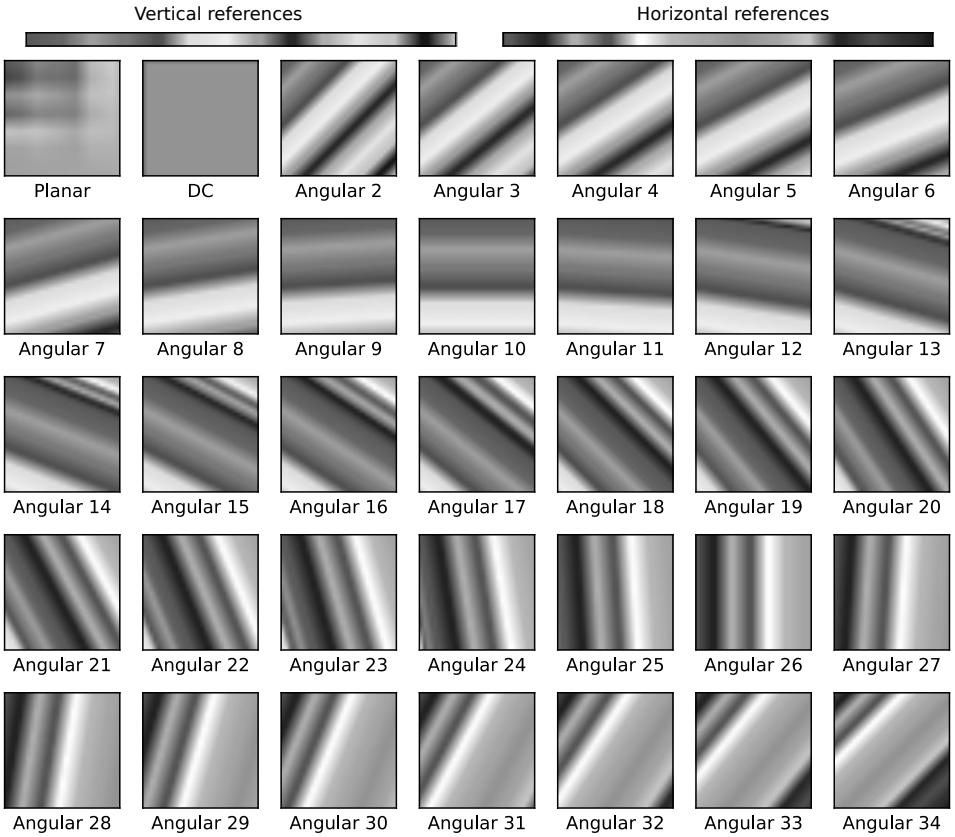
of the original block, as depicted in Figure 12. The reference samples belong to already coded, reconstructed blocks.

Figure 12 – Intra reference samples on HEVC standard.



Source: adapted from (Sze; Budagavi; Sullivan, 2014).

Figure 13 – An example of the 35 intra prediction modes available on HEVC.



Source: adapted from (Sze; Budagavi; Sullivan, 2014).

Figure 13 illustrates an example of 35 possible candidates and their respective indices on HEVC. The first candidate, called planar, is computed as the arithmetic mean of two linear predictions: one using horizontal references and the other using vertical references. The DC candidate, in turn, is generated using only the average value of all reference samples. The remaining 33 candidates are angular projections of the reference samples along different directions.

VVC introduces additional possibilities for intra mode prediction. There is a new angle for each pair of adjacent angles, resulting in 63 angular candidates. There is also the possibility of predicting non-square partitions – which are not allowed in HEVC – and of using multiple lines and columns of reference samples. We do not detail the VVC intra mode, as the candidate generation process presented for HEVC is sufficient to understand the intra-related works discussed in the next chapter.

Intra mode selects the best candidate using the RDO model, which evaluates the actual bitrate and resulting distortion. However, this approach increases the encoder's complexity, as the block must be processed through the entire encoding pipeline. Although the RDO model is used to select the reference block, it is only applied for a subset of pre-selected candidates evaluated with the simplified RDO model.

2.4.2 Inter Mode

Inter mode prediction performs a process called **Motion Estimation (ME)** to capture temporal redundancies. The ME searches a block that minimizes j_{cost}, relying on **Integer Motion Estimation (IME)** and **Fractional Motion Estimation (FME)** to achieve this goal. Figure 14 shows the IME process. It is termed 'integer' because candidates consist of samples positioned at integer locations. IME searches among candidates within a defined search area, and after selecting the best block, FME begins.

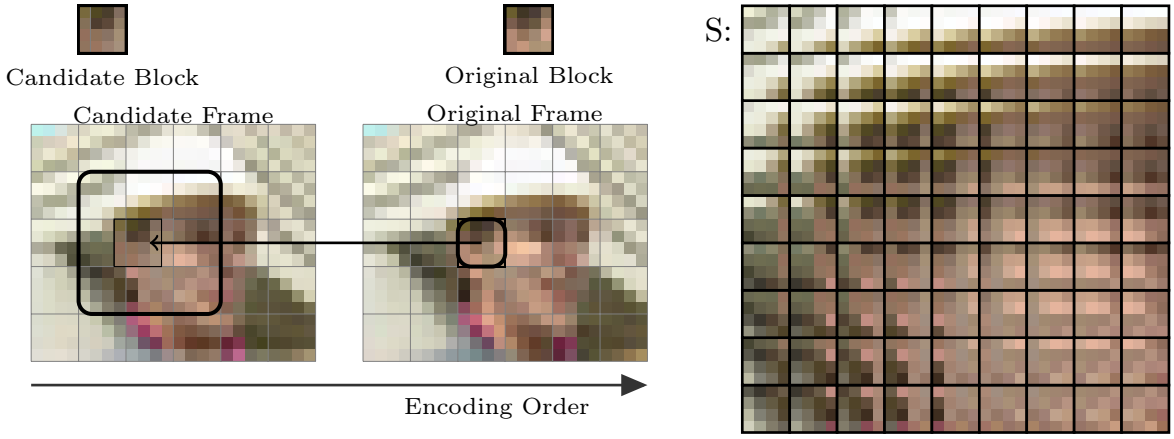
FME searches within synthetic samples located in fractional positions that are generated by applying an **interpolation filter** to the integer samples surrounding the block selected by IME. An interpolation filter expression is given by:

$$\sum_{i=1}^{n} X_i \times c_i \tag{2.2}$$

, where n is the total number of samples X_i to interpolate the new sample, and c_i is a filter coefficient defined by the standard. Figure 15 illustrates the fractional space with integer, half, and quarter sample precision. The output of the ME process is a **Motion Vector (MV)**, which indicates the motion of the original block relative to the reference block position with quarter-pixel precision.

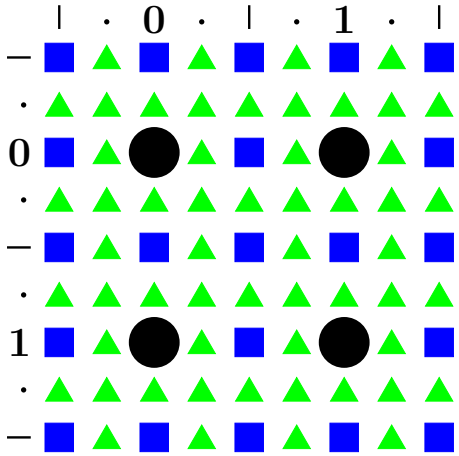
Similarly to the frame classification, there are three types of PUs. Furthermore, the frame type determines which PU types it can contain. An I frame has only I-type PUs. A P frame can have both I- and P-type PUs, and a B frame can have B-type

Figure 14 – IME compares the Original block with candidate blocks at integer positions. In this example, S is the set of all possible candidates within the search area delimited by the rectangle in the candidate frame.



Source: the author.

Figure 15 – Representation of fractional sample space. Circles, squares, and triangles represent samples at integer, half, and quarter positions, respectively.



Source: the author.

PUs in addition to the other two types. To encode a P-type PU in a P frame, the encoder processes ME for each frame in a list of reference frames. In addition to motion information, the reference frame containing the selected predictor must be identified. The execution flow is similar to encoding a P-type PU in a B frame. However, since there are two lists of reference frames, the encoder must also indicate which list contains the selected frame. A B-type PU can be seen as a combination of two P-type PUs, each predicted from a different list of reference frames associated with the B frame. The B-type predicted block is obtained by computing the element-wise average of two P-type PUs.

Besides the ME, HEVC employs two additional processes in Inter Mode. The first one is **Advanced Motion Vector Prediction (AMVP)**, which predicts the MV of a block based on data from previously encoded blocks The second one is called **Merge**

mode, reducing signaling when there is repeated motion information in neighboring blocks.

HEVC and VVC exhibit minimal differences regarding the ME process. MV signaling and FME filters differ for each standard, but the key concepts are the same. One notable addition in VVC is the **Affine Motion Compensation**, which extends redundancy exploitation beyond translation (block position shifts in the reference frame) to include scaling, rotation, shape transformations, and shearing.

2.5 TRANSFORM, QUANTIZATION AND RECONSTRUCTION

After the prediction step, the encoder has all the information about decisions and residuals to encode a CTU. So far, the encoder represents frames and block's data in a 2D space called **spatial domain**. However, the **Transform** step changes the samples from the spatial domain to the **frequency domain**. In the frequency domain, the block data is represented by the weights of different frequency components.

There are two advantages to changing from the spatial to the frequency domain. First, a block in the frequency domain concentrates energy in low-frequency components. Energy compaction can be achieved through various transforms, among which the Karhunen-Loève Transform (KLT) and the Discrete Cosine Transform (DCT) are particularly notable. The KLT, in particular, is known for its optimal energy compaction properties, but its data-dependent nature requires the transmission of transform coefficients, limiting its suitability for standardized compression systems. Usually, a transform based on DCT – proposed by Ahmed, Natarajan and Rao (1974) – is adopted with standardized coefficients. HEVC uses a single transform matrix for each block with a size up to 32×32. Although KLT was not considered for the VVC standard, it includes three different transforms: two DCT-based and one Discrete Sine Transform (DST)-based. With more than one possibility, the encoder can select the one offering the best energy compaction for a given block.

The second main advantage of the frequency domain is that human vision is less sensitive to high-frequency elements. If no chroma channel is sub-sampled, up to this point the encoding process could remain lossless[2] , but this changes after the **Quantization** step. Quantization may scale down the components in the frequency domain. Elements at higher frequencies undergo coarser quantization than those at lower frequencies. The scaling values are calculated based on a **Quantization Parameter (QP)**. A large QP will perform a more aggressive quantization, prioritizing a higher compression rate at the expense of reduced visual quality.

The encoder must reconstruct the block to feed back the prediction step. The first two steps consist of inverse quantization followed by inverse transform to reconstruct

[2] Usually, the standard specifies just the bitstream and how to decode it, but there are some exceptions. The standard has a constraint to the number of bits of any internal value in encoding. If some value exceeds this limit, it is clipped, and an error is introduced before the quantization.

the residual block in the spatial domain, which considers the error introduced by the encoder. The residual block must be added to the reference to generate the block that composes the reconstructed frame. The Advanced Video Coding (AVC) standard was the first to introduce **in-loop filters**. In-loop filters are applied to smooth edges and reduce block artifacts[3], improving the perceptive visual quality. The reconstructed blocks are then added to a buffer to be used as a reference for the following reconstructions.

2.6 ENTROPY ENCODING

The quantization result and standard signaling (syntax elements) suffice to represent the encoded video. However, significant symbol repetition occurs (e.g., long sequences of 0s resulting from the quantization of high-frequency components). Therefore, an **entropy encoder** reduces symbol redundancies before generating the final bitstream. Entropy encoding performs lossless compression, which is mandatory because the bitstream must be recovered on the decoder side to reconstruct the video.

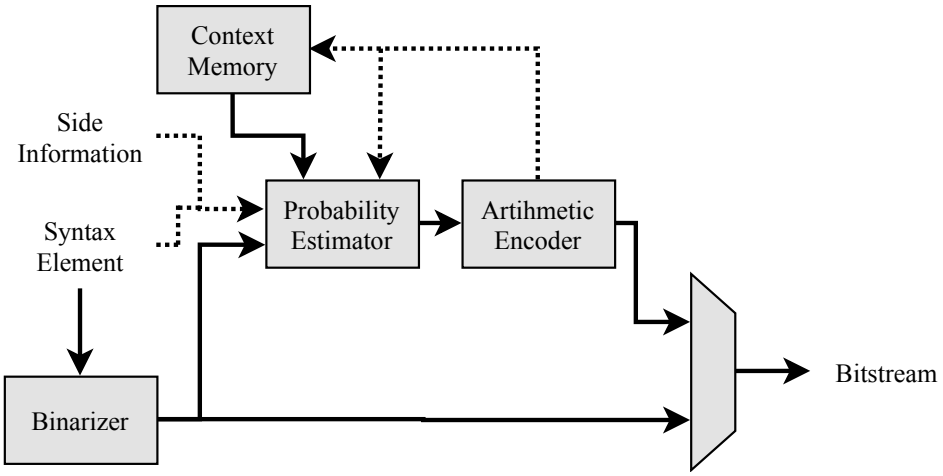
HEVC uses an entropy encoder called Context-Adaptive Binary Arithmetic Coding (CABAC) (Sze; Budagavi, 2013), which block diagram is shown in Figure 16. First, CABAC converts syntax elements to a sequence of bits through the binarization process. The compression itself is performed using **arithmetic coding**. Arithmetic coding encodes data based on the probability of each symbol's occurrence. If the probability model is sufficiently accurate, the output bitstream is close to the optimal compression rate. To better explore this characteristic, HEVC uses distinct probability models for different syntax elements. Moreover, the probability estimator uses side information (such as CU and PU sizes), syntax elements, and feedback from the arithmetic encoder to change its state and select the appropriate probability distribution used by the arithmetic encoder – which justifies the term **context-adaptive** in arithmetic coding.

2.7 CODING EFFICIENCY

When a new coding technique is proposed, its impact must be measured in terms of bitrate and quality. The bitrate is the number of bits per time unit (usually per second) or bits per pixel (bpp); the smaller the bitrate, the higher the compression. Quality can be evaluated using either subjective or objective methods. Subjective metrics require controlled environments and specific protocols for human evaluation of compressed frames, whereas objective metrics are computed directly from the original and compressed frames. Therefore, objective methods are commonly used in scientific works because of the lower costs and easier reproducibility.

[3] In video compression, artifacts are visual perceptive errors in the decoded video. A block artifact occurs when the block separation used by the encoder becomes visible in the decoded video.

Figure 16 – The CABAC model in HEVC. Based on the coding context, the probability estimator uses side information (such as CU and PU sizes), syntax elements, and feedback from the arithmetic encoder to change its state and select the appropriate probability distribution.



Source: adapted from Sze, Budagavi and Sullivan (2014) book.

Codecs such as the HEVC and VVC reference softwares – HEVC Test Model (HM) and VVC Test Model (VTM), respectively – report the compression quality with a metric called Peak signal-to-noise ratio (PSNR), measured in decibels (dB). The PSNR of a color channel can be calculated as follows:

$$\text{PSNR} = 10 \times log_{10}\left(\frac{\text{MAX}_b^2}{\text{MSE}(\mathbf{F}^{\text{ori}}, \mathbf{F}^{\text{rec}})}\right) \tag{2.3}$$

, where MAX_b is the maximum value for a sample with b bits and the Mean Squared Error (MSE) between an original frame ($\mathbf{F}^{\text{ori}}$) and a reconstructed frame ($\mathbf{F}^{\text{rec}}$) of $w \times h$ resolution is calculated as:

$$\text{MSE}(\mathbf{F}^{\text{ori}}, \mathbf{F}^{\text{rec}}) = \frac{1}{w \times h} \sum_{i=0}^{h-1} \sum_{j=0}^{w-1} \left(\mathbf{F}^{\text{ori}}{}_{i,j} - \mathbf{F}^{\text{rec}}{}_{i,j}\right)^2 \tag{2.4}$$

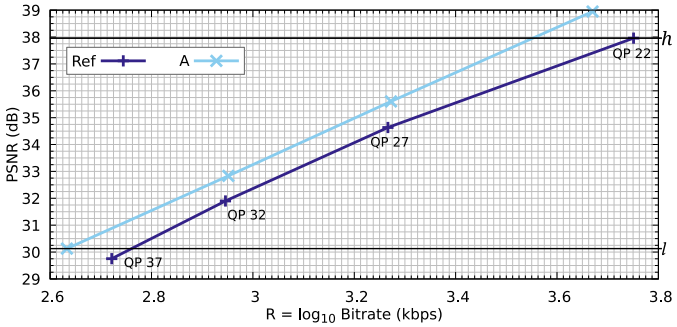
The PSNR is first calculated for each channel (Y, U, and V), and then the PSNR-YUV is calculated as the weighted average of the channel's results, with the luma channel weighted by six, and the chroma channels each weighted at one. Since learned compression models typically operate on the RGB color space, studies often report PSNR-RGB, calculated as the simple average of each color component.

There are other quality metrics that better correlate with the Human Visual System (HVS). For instance, the PSNR-HVS-M remains based on PSNR but takes into account the characteristics of the HVS (frequency and contrast). The Multiscale Structural Similarity Index Measure (MS-SSIM) metric also aims to improve the correlation, but instead of being based on PSNR, it considers the image structural information at multiple scales.

Encoding the same video using two configurations combined with four different QPs, two Rate-Distortion (RD) curves can be plotted (RD curves) as shown in Figure

17. A point in those curves represents the bitrate and quality results for a given QP. To compare those RD curves, Bjøntegaard (2001) created a metric called Bjøntegaard Delta Bitrate (BD-Rate) that we explain with the following example. In this example, we want to compare the encoding efficiency of curve A relative to *Ref* (the latter is the baseline configuration).

Figure 17 – Example of RD curves using QPs 22, 27, 32, and 37. l and h are, respectively, the lower and the higher values of PSNR shared by A and *Ref* RD curves.



Source: Adapted from: Bjøntegaard (2001).

Notice that the bitrate has to be scaled as follows:

$$R = log_{10}(bitrate) \tag{2.5}$$

Each RD curve is then approximated by a polynomial function of third-degree like the following one:

$$f(D) = R = a + b \times D + c \times D^2 + d \times D^3 \tag{2.6}$$

, where a, b, c, and d are fitting constants used to describe the scaled bitrate (R) as a function of PSNR (D).

Let f_A and f_{Ref} be the functions that approximate A and *Ref* RD curves, the BD-Rate is now calculated as follows:

$$\text{BD-Rate} = (10^{\frac{1}{h-l}\int_l^h \left(f_{Ref}(D) - f_A(D)\right)dD} - 1) \times 100\% \tag{2.7}$$

, where l and h are, respectively, the lower and the higher values of PSNR shared by A and *Ref* RD curves. A positive BD-Rate indicates that A has worse encoding efficiency than *Ref* because, on average, A results in a higher bitrate to achieve the same quality – in terms of PSNR – than *Ref*. The opposite happens when the BD-Rate result is negative (as shown in the example in Figure 17), indicating that, on average, A requires a lower bitrate to achieve the same quality as *Ref*.

The Bjøntegaard Delta (BD) formulation can be applied to MS-SSIM after appropriate conversion, as described below (Herglotz et al., 2024):

$$\text{MS-SSIM}_{\text{dB}} = -10 \log_{10}(1 - \text{MS-SSIM}) \tag{2.8}$$

3 NEURAL NETWORK CONCEPTS

A **machine learning model** is a function that establishes a relationship between a set of input variables, referred to as **features**, and an output value. The model learns how to make accurate output predictions through a training process, which, in this work and related studies, is performed using **supervised learning** methods. Supervised learning demands a dataset with input features and the corresponding expected solution. The model is fed with input features and a backpropagation algorithm adjusts the model's parameters, aiming to reduce the error between the prediction and the expected outputs.

There are two problem categories that models trained with supervised learning can solve: classification and regression problems. In **classification problems**, the model evaluates the features and predicts a label from a predefined finite set of labels (e.g. predict if a block has homogeneous or complex texture). In contrast, **regression problems** predict a continuous value in a given range (e.g. predict a pixel value considering the surrounding ones).

In this chapter, we provide an overview on Neural Network (NN) basic concepts. We explain NN structures in the following sections as well as the Convolutional Neural Network (CNN), a special case of NN that uses convolution layers and is widely adopted in image and video compression research. Our explanation focuses primarily on model architecture. The reader should refer to Goodfellow, Bengio and Courville (2016) and Aggarwal (2018) for detailed explanations on training algorithms and challenges related to NN development.

3.1 NEURAL NETWORK

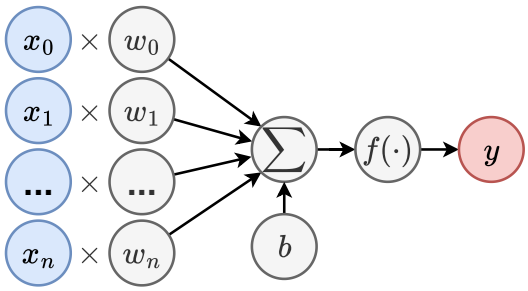
Before presenting the architecture of a neural network itself, it is necessary to understand its fundamental component: the artificial neuron (or perceptron). A **artificial neuron** (Figure 18) receives input features, each multiplied by a weight, which are then summed together with a bias. This operation can also be expressed with the following equation:

$$y = f\left(b + \sum_{i=1}^{n} w_i \times x_i\right) \tag{3.1}$$

, where x_i are the input features, w_i are the weights of the neuron, and b is the bias. Observe that a function $f(\cdot)$ is applied to the result of this weighted sum. This function, called the **activation function**, allows the neuron to produce a non-linear output. There are many possible activation functions. One example is the Rectified Linear Unit (ReLU) function, expressed as:

$$\mathrm{ReLU}(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases} \tag{3.2}$$

Figure 18 – Structure of an artificial neuron. It receives x input values, each one multiplied by a learned parameter w. The results are summed together with a bias value b, and then an activation function $f(\cdot)$ is applied, producing the output value y.



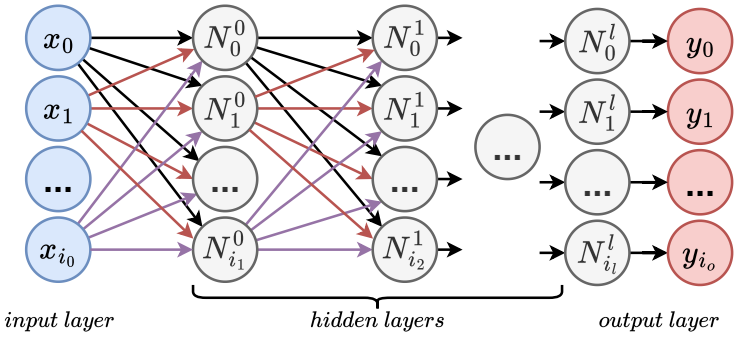
A single neuron can solve only simple problems and is therefore not very useful on its own. To increase representation power, neurons are combined into layers. The operation of a fully connected layer can be expressed as:

$$\mathbf{h}^{(l)} = f\left(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}\right) \tag{3.3}$$

, where $\mathbf{h}^{(l-1)}$ is the input vector from the previous layer, $\mathbf{W}^{(l)}$ is the weight matrix of layer l, $\mathbf{b}^{(l)}$ is the bias vector and $\mathbf{h}^{(l)}$ is the output of layer l.

A NN with a single layer can solve different problems. However, such a layer may become impractically large and lose its ability to generalize to unseen data. For this reason, NN models are usually composed of multiple layers, each containing a smaller number of neurons. Because each neuron in a layer connects to all neurons in the next layer, these structures are known as fully connected (or dense) layers. Figure 19 shows an example of a NN model with multiple layers, so called Multilayer Perceptron (MLP). A MLP typically consists of three types of layers: the input layer, which its only role is to receive the features; one or more hidden layers, each with its own number of neurons and activation functions; and the output layer, which produces the final prediction.

Figure 19 – A multilayer perceptron neural network with one input layer of x values, l hidden layers, and one output layer of y values. Each hidden layer l can be parameterized with i_l neurons N.



MLP is a powerful model, but it lacks an essential property for image- and video-related problems: it does not account for spatial correlations between input features. Indeed, in an image, each pixel is correlated with its neighboring pixels, and exploiting such correlations can help extract more relevant information for the NN model. To over-

come such limitation, CNN architectures were introduced, and in the next section we present the key aspects relevant to this work.

3.2 CONVOLUTION NEURAL NETWORK

A **CNN**, as the name suggests, is a neural network model that applies convolution kernels to the input features. This allows the CNN to exploit image structures derived from the spatial correlations of neighboring pixels. Figure 20 shows a typical structure of a CNN. The input is processed by a **convolution layer**, followed by the activation function. After this step, the network may include a **pooling layer**, which extracts relevant features and reduces the spatial resolution for the next layer.

Figure 20 – Example of a sequence of operations within a convolution neural network. The network can be built by concatenating multiple sequences of this structure.

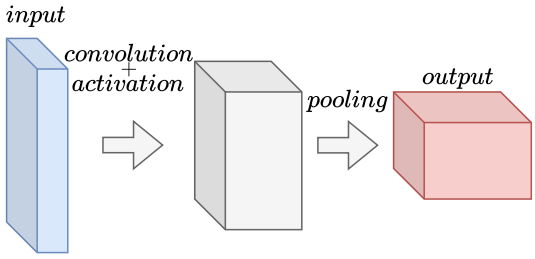
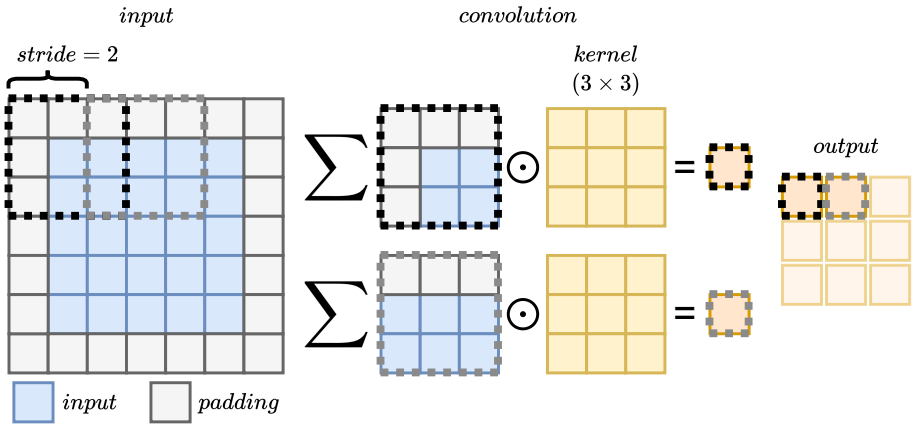


Figure 21 illustrates the convolution process. For a convolution layer, it is necessary to specify the kernel size, the number of kernels, and the stride, the latter specifies how the kernel is shifted across the image. It should be noted that even with a stride of one, convolution naturally reduces the spatial dimensions of the input, since the kernel cannot extend beyond the image boundaries. To preserve the original resolution, padding can be applied, extending the input image by adding nominal values (commonly zeros) around its borders.

Figure 21 – Example of convolution layer operations. The main parameters include padding (to extend the input borders), stride (to space the convolution operations), and kernel size. The $\odot$ represents the element-wise multiplication.



Each convolution consists of an element-wise multiplication between the kernel weights and the input patch, followed by summation, producing a single output value of the output result. It is important to note that the kernel weights are learned during the training process. Although the example in Figure 21 illustrates the convolution in a 2D input, the 2D convolution is commonly applied to multiple input layers. In turn, Figure 22 presents an example with an input with depth size greater than one, also exemplifying how to calculate the output dimensions of a convolution layer.

Figure 22 – Example illustrating how to calculate the output sizes, given an input with dimensions (h_i, w_i, d_i), k kernels of size (h_k, w_k), and stride s.

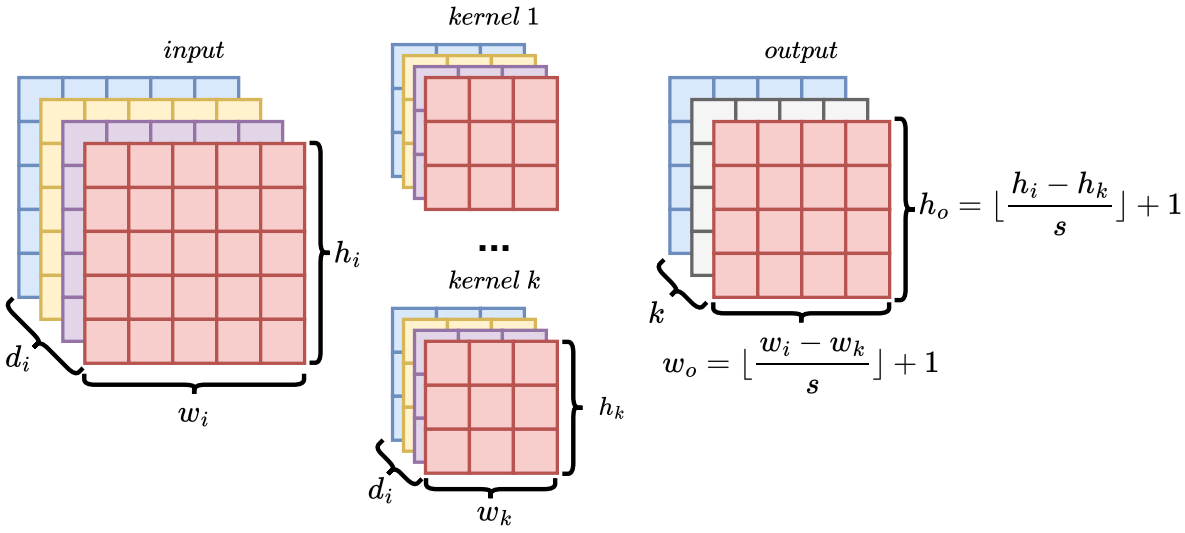
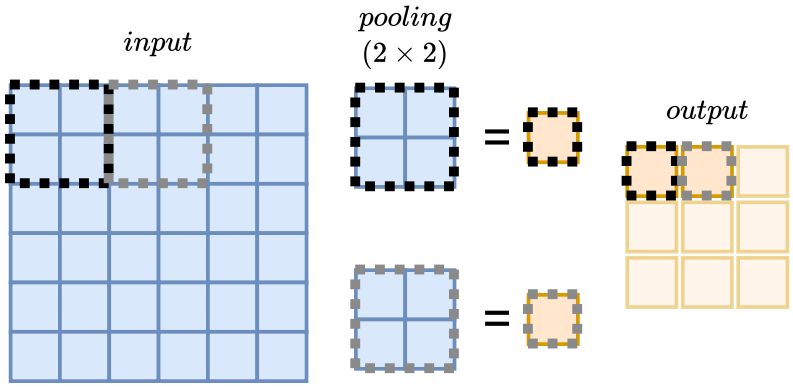


Figure 23 illustrates the pooling operation. The most common parameters for the pooling layer are the window size, stride and the pooling method. Usually the pooling operation consists in calculating either the average or the maximum value of each window.

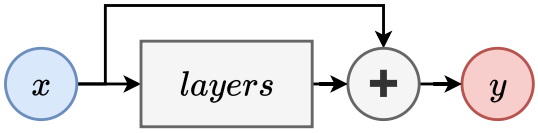
Figure 23 – Example of a pooling operation with size 2×2. The most common pooling operations are max pooling and average pooling. Pooling can also be configured with a stride value.



3.2.1 Other NN structures

The first and most widely used CNN structure in video compression, illustrated in Figure 24, is designed to perform **residual learning**. Given an input image, instead of directly predicting the expected output image, the CNN learns the residuals between the input and the target image. The main motivation for residual learning is that it accelerates convergence during training. Importantly, this does not alter the problem definition, since adding the predicted residual to the input image yields the same result as predicting the target image directly.

Figure 24 – The structure used for residual learning.



Another common strategy to enrich feature representations is by **concatenating** them with additional data at certain stages of the CNN design. A related operation is the **merge**, which combines two input images into a single representation. This can be implemented, for example, through element-wise summation, averaging, or concatenation. In this work, the proposed model has a module called "merge block". The merge block is a structure inspired by the U-Net architecture (Ronneberger; Fischer; Brox, 2015), and it is responsible for processing concatenated images to produce a single merged image.

Specifically for compression, there is a neural network architecture called autoencoder, depicted in Figure 25. In this structure, the feature dimensions are progressively reduced layer by layer until they reach a compact latent representation, known as the bottleneck. Subsequent layers expand the feature dimensions to reconstruct the input at the output. This structure can be interpreted as a form of compression, since the network must learn to encode the input into a compact latent representation. However, unlike traditional compression algorithms, the reconstruction is learned and may not be exact.

3.3 NEURAL NETWORK TRAINING

Training a NN consists of adjusting its parameters to minimize a loss function over a set of training data. There are three important parameters in NN training: number of epochs, batch size, and learning rate. An **epoch** is defined as one complete pass through all instances in the training dataset. The number of epochs limits the training process and should be large enough to ensure convergence.

The second parameter is the **batch size**. Updating the NN weights for each sample individually (stochastic gradient descent) can be computationally expensive and may lead to highly noisy updates. On the other hand, updating weights only after processing the entire dataset (batch gradient descent) produces smoother updates but slows

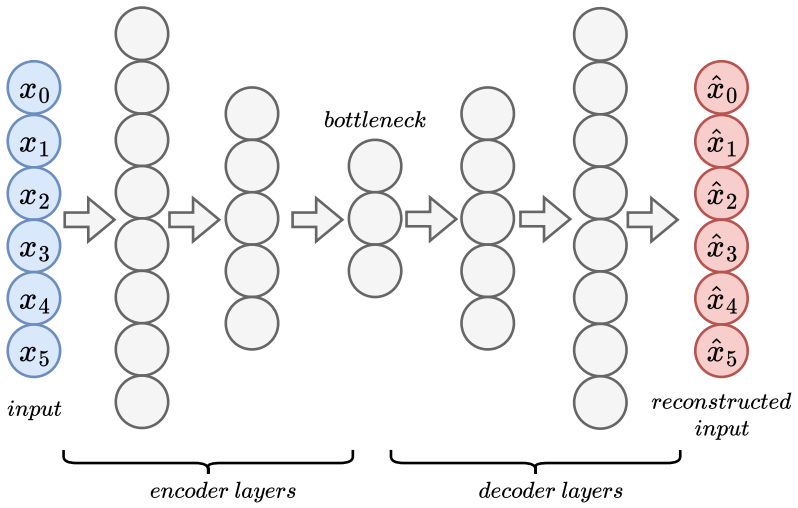
convergence due to fewer parameter updates. Processing the dataset in small batches balances these effects, accelerating convergence while keeping the noise in weight updates manageable.

The **learning rate** determines the step size used by the training algorithm to adjust the NN weights toward cost minimization. A small learning rate results in slow convergence, while a learning rate that is too large may cause the weights to diverge from the optimal solution. Typically, the optimal learning rate is determined empirically.

In addition to these training parameters, there are also key algorithmic choices and functions that guide the learning process. Training requires a definition of a **loss function** (or cost function), which quantifies the difference between the model's predictions and the expected outputs. This function provides the objective that the optimization algorithm seeks to minimize. The choice of loss function depends on the task: for example, regression problems commonly employ the MSE, while classification tasks often rely on cross-entropy loss. In specialized applications, such as image compression, alternative loss functions may be used to better align with human visual perception. For instance, perceptual metrics like MS-SSIM are often preferred because they correlate more strongly with the HVS than simple pixel-wise errors. Furthermore, in multi-objective scenarios, the loss function can incorporate weighted terms that balance different goals – for example, simultaneously minimizing distortion error while aiming for higher compression efficiency.

To perform the parameter updates, different **optimization algorithms** can be employed. The simplest one is the stochastic gradient descent (SGD), where parameters are updated in the opposite direction of the gradient. Several refinements of SGD are widely used in practice, such as Momentum, which accelerates convergence by smoothing updates; RMSProp, which adapts the learning rate of each parameter individually; and

Figure 25 – Example of an autoencoder neural network. The model is trained to reconstruct its input, but because it includes a bottleneck, the reconstruction is a learned approximation. The bottleneck can be seen as a compressed representation of the input.



Adam, which combines both momentum and adaptive learning rates and has become a default choice for many applications.

Finally, it is important to address the risk of **overfitting**, which occurs when the model fits the training data too closely and fails to generalize to unseen examples. To mitigate this problem, various regularization strategies are employed. Common approaches include weight decay (L2 regularization), which penalizes large parameter values; dropout, which randomly deactivates neurons during training to reduce feature co-adaptation; and early stopping, which halts training when validation performance ceases to improve. These techniques ensure that the trained model remains robust and generalizes effectively beyond the training dataset.

4 SYSTEMATIC LITERATURE REVIEW

In this chapter, we present the systematic literature review conducted to identify the related work. After applying the method presented in Section 4.1 and reading the selected papers, we identified different groups of works, which led us to propose the taxonomy presented in Section 4.2. In Section 4.3, we expand the review by presenting additional fully learned image and video compression solutions, as well as covering multi-layer compression solutions, which were not included at the time the systematic review was conducted. This section also highlights works that are most closely related to our approach, including the study by Cheng et al. (2020), which was incorporated as a component of the proposed Asymmetric Multi-layer Compressor (AMLC) solution.

4.1 METHOD

We planned the systematic review to identify works that use NN in proposed solutions to improve video compression, with complexity reduction or coding efficiency gains as target goals. We collected primary studies using *Scopus* as the search engine. Our search string (see Appendix A.1) was designed to filter works that propose machine learning solutions applied to video or image compression. Additionally, we excluded studies focused on medical or forensic applications.

A first search was conducted in July 2019 and restricted to publications after 2017. We found 369 papers (excluding three duplicated entries) with the specified search string. After reading the title and abstract, we selected only 104 papers. The broad scope of our search string may explain the low acceptance rate. We excluded works that propose solutions applied only to images, using machine learning techniques other than NN (e.g., Support-Vector Machine (SVM), or decision trees), and the ones addressing topics not related to our review, such as video-content classification, or visual quality assessment.

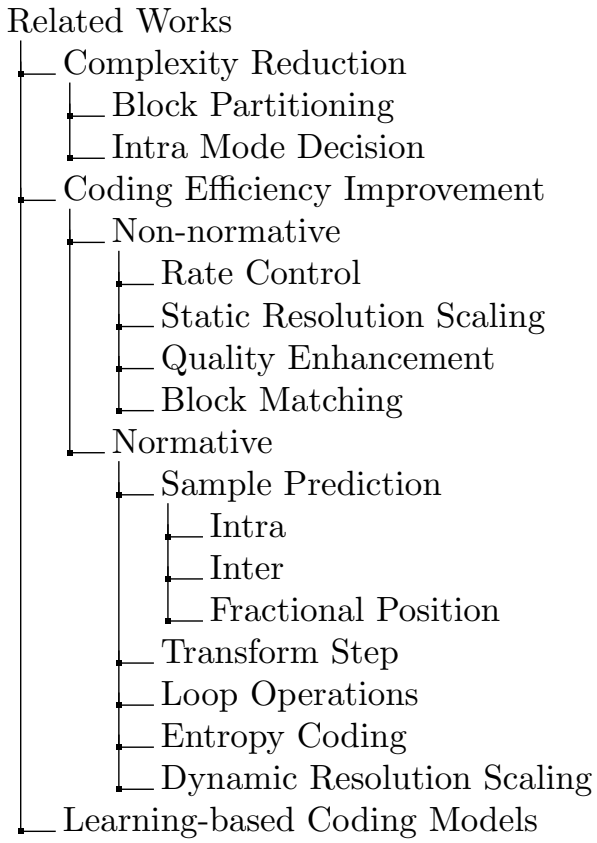
After reviewing the papers, we selected 43 for inclusion in our analysis. In March 2020, we updated our research using a search string restricted to the video compression topic (see Appendix A.2). We selected 16 additional works using the same method applied to the previous search.

In our research, we found a survey entitled "Machine Learning Based Video Coding Optimizations: A Survey" from Zhang, Kwong and Wang (2020). Their work presents a broad overview of machine learning techniques applied to video coding. Three of the 11 works presented by the authors, which also relate to our objective, were already present in our systematic review. After adding the remaining ones to our selection, **we covered 67 works in this review**. By focusing on NN-based video compression, we were able to conduct a more exhaustive review of this specific domain, including more works on the topic than previous, broader surveys.

4.2 PROPOSED TAXONOMY

After reading the works, we grouped them according to the taxonomy presented in Figure 26. First, we distinguished the works by their primary goals: Complexity reduction or coding efficiency improvement. If the goal is **complexity reduction**[1], a common approach involves pruning the number of encoding options. As a result, these solutions tend to reduce coding efficiency. In contrast, NN-based solutions that aim to **improve coding efficiency** introduce additional processing to the video encoding flow, which increases the overall complexity. A third category identified in this study comprises works that propose new encoding models entirely based on NNs. In this case, the encoding and decoding flows must be completely redesigned.

Figure 26 – Proposed taxonomy for the literature works.



Source: the author.

The **complexity reduction** works found in this study are all **non-normative**, i.e., they do not require any modifications on the decoder side and are thus considered as standard compliant. We found works that propose restricting the exploration of **block partitioning**, resulting in fewer tested modes to encode a block. Because **intra mode decision** is made using the full RDO method – a very time-consuming process – it is also possible to find works that propose narrowing the number of intra modes tested.

[1] In video coding works, complexity is commonly measured as total execution time.

We categorize the works aiming to improve **coding efficiency** into non-normative and normative. In the non-normative group, we identify the following categories: rate control, static resolution scaling, quality enhancement, and movement prediction. **Rate control** works aim to encode a video with a target bitrate, adjusting the encoding quality as necessary. **Static resolution scaling** works reduce the spatial resolution before encoding and perform super scaling after decoding, with an optional quality enhancement step. Resolution scaling can be applied to spatial and temporal dimensions, as well as to bit-depth scaling strategies. **Quality enhancement** works operate on the decoder side to generate better frame quality. Finally, **block matching** solutions replace the conventional RDO-based method with evaluations using CNN models.

In the normative group of works that improve coding efficiency, we list the following categories: sample prediction methods – further divided into intra, inter (integer position samples only), and fractional position sample predictions – transform step, loop operations, entropy coding, and dynamic resolution scaling. **Sample interpolation** works typically use CNN models to generate references unavailable in the conventional encoding flow. It is possible to find works that explore intra and/or inter-frame data to generate the samples. In inter frame prediction, some solutions directly predict the samples at integer positions. However, other works aim to replace conventional **fractional interpolation** methods with ones derived from NN models. Some works use NN models to learn how to perform the **transform step**. In the **reconstruction loop**, many works propose replacing handcrafted in-loop filters with a CNN model to improve reconstruction quality. There are works aiming to improve the **entropy encoding** by replacing hand-crafted probability models with the ones predicted by an NN. Similar to static resolution scaling, **dynamic resolution scaling** also applies pre- and post-scaling methods. Dynamically scaling the resolution requires an additional signal to define when a given scaling ratio is used, which makes those works fall in the normative group.

Lastly, there are works proposing **encoder flows entirely based on NN models**. Due to its nature, this category contains only normative works. However, we defined it as a separate category because they are not built on top of any existing video compression standard. They may share solutions with previous categories, such as intra- or inter-sample predictions, but not as replacements or extensions to standard handcrafted tools within existing codecs. In these works, the entire set of coding tools is implemented using learned models.

In the following subsections, we describe the details of the works belonging to each category, including the proposed solutions and summarized results. In cases where the original authors presented a broad exploration of NN designs and experimental results, we summarize the data in separate rows within each table to reflect these variations. Despite Common Test Conditions (CTC) guidelines, we found that encoding configurations, test sequences, and metrics vary across works, making direct comparisons unreliable. In addition, some works lacked essential methodological details to enable experiment and

results replication, while others used unconventional metrics. We strongly advise reading these papers for more details about the proposals, experimental setup, and conclusions.

4.2.1 Complexity Reduction Works

In this subsection, we present the groups of works proposing solutions to reduce encoding complexity.

Block Partitioning

The block partitioning problem involves evaluating the costs of various partitioning schemes to determine the best one. The increasing number of possible partitions – resulting from more flexible partitioning structures in recent video standards – combined with the large number of RDO computations to decide the best mode for each evaluated partition is a major contributor to encoding complexity. A solution that imposes restrictions on the block partitioning search reduces complexity by avoiding mode evaluations.

Table 2 summarizes the works reporting BD-Rate and time reduction reported results relative to their baseline encoder. Only three works use the Joint Exploration Model (JEM) encoder, which was built from the HM for exploration purposes but already contains a draft of features that were later incorporated into VTM.

We found three different methods for fast block partitioning decisions:

- Decide when the split is needed (binary classification);
- Limit the search depth based on different block categories (multi-label classification);
- Evaluate the probability of an edge being part of the final partitioning scheme (regression).

Liu et al. (2016), Li et al. (2019a), Zhou, Ye and Wang (2017), Kim and Ro (2019), and Chung, Peng and Hu (2017) present solutions using the first method. Liu et al. (2016) propose a CNN to decide if a CU needs to be split. The CNN receives only 8×8 blocks, and other block sizes are sub-sampled with local averaging to fit the model input. This was the only study in this review that proposed a hardware architecture for the CNN and reported detailed results in that context. The proposed hardware can encode 1080p videos in real-time at 55 frames per second (fps). They also propose a fast intra mode decision, but their strategy analyzes texture complexity with a hand-crafted algorithm. Li et al. (2019a) use a single CNN model to predict split decision of different block sizes. A distinctive aspect of their work was the restriction to predicting partitioning only in intra frames from screen content video, which limits its applicability. The CNN model is fed with a 32×32 block, while 64×64 is sub-sampled and 16×16 upsampled.

Zhou, Ye and Wang (2017) combined hand-crafted algorithms with a NN approach to guide the split decisions. They use a metric to analyze the block texture. If

Table 2 – Block partitioning related work results.

Work	Encoder	Configuration	BD-Rate (%)	Time Saving (%)
Liu et al. (2016)	HM-12.0	AI	2.66 4.79	63.00 73.30
Li et al. (2019a)	HM-16.5	LDB, GOP 4	0.85 2.56	34.34 73.30
Zhou, Ye and Wang (2017)	HM-15.0	AI	1.78	51.85
Kim and Ro (2019)	HM-15.0	LDP RA	3.75 3.91	61.62 61.77
Chung, Peng and Hu (2017)	HM-16.15	AI	2.50	-
Feng et al. (2018b)	HM-16.9	AI	0.99	27.54
Jin et al. (2018a)	JEM-3.1	AI	0.69 2.04	42.33 62.80
Xu et al. (2018)	HM-16.5	AI LDP LDB RA	2.247 1.49 1.72 1.48	61.84 54.20 50.44 54.60
Wang et al. (2018)	JEM-7.0	RA	0.55	35.00
Galpin et al. (2019)	JEM-7.0	AI	≈0 ≈1	54.00 75.00

the texture is classified as homogeneous, it is not further split. On the other hand, if the block's texture is too complex, it is split. An uncertainty region lies between these two extremes. If the block size is 32×32 or 16×16, the authors adopted the execution flow from HEVC reference encoder. If the block size is 64×64 or 8×8, a MLP model decides to split it or not. A specific model was designed for each block size, and they were trained to predict only split decisions of intra frames.

Kim and Ro (2019) trained a CNN to decide when to split a block for each depth level of the quadtree structure. In addition to the convolution layers output, the final fully connected layers also take in a vector with information about the mode decisions of the block under evaluation. For intra mode, the network is fed with the PU mode and luma and chroma angles. For inter mode, PU mode and motion vector information are provided.

Considering all selected works, Chung, Peng and Hu (2017) are the only ones who use reinforcement learning. Instead of training on a dataset of split/non-split decisions, they propose to make decisions based on the RDO model. They define a state of the model as the block of luma samples and the QP; the action as split the input block; the reward as the RDO cost reduction relative to the non-split decision (immediate reward). Unlike conventional reinforcement learning problems, this model generates a set of four

new states after a reward representing the quadtree split.

Feng et al. (2018b), Jin et al. (2018b), Xu et al. (2018), Soulef et al. (2019), and Wang et al. (2018) use the second method, proposing a single model to predict the partitioning depth. Feng et al. (2018b) propose a CNN model to predict five different sets of HEVC partitioning depths: {0}, {1}, {1,2}, {1,2,3}, and {2,3}. Afterward, they group the five predictions into two categories: simple texture – search restricted from depth 0 to 2 – and complex texture – search restricted from depth 1 to 3. The authors argue that the model can better explore CTU structures and achieve greater accuracy with five depth partitioning labels rather than directly predicting the two texture labels.

Jin et al. (2018b) propose a model to analyze 32×32 CUs and define the QTBT search depth in VVC encoders considering five depth descriptions: very shallow, shallow, medium, deep, and highly deep. Considering a 128×128 CTU, their algorithm has three scenarios: 1) If all CU are predicted as very shallow, only quadtree partitioning from 128×128 to 32×32 are evaluated; 2) If there is no deep or highly deep prediction, only quadtree partitionings from 64×64 to 32×32 are evaluated; 3) Otherwise, the min and max search depths of each 32×32 CU are predefined for each predicted class. The authors expanded their investigations in (Jin et al., 2018a), analyzing the depth description impact, using different loss metrics for training, and exploring additional experiment configurations.

Xu et al. (2018) present a solution for HEVC CTU partitioning using a single model to derive quadtree split decisions in all depth levels. Firstly, their solution evaluates split decisions for intra predictions. Considering a 64×64 CTU, they first perform the mean removal for the whole block – subtracting the average block sample from each sample – and then sub-sample it to a 16×16 block. The mean removal is repeated for 32×32 partition size, then the CTU is sub-sampled to 32×32. A third block is generated by applying the mean removal for partitions with size 16×16 without performing the sub-sampling step. The model has three convolutional branches, each processing a block from the previous step. The output concatenation of the convolutional branches feeds three MLP models, each one trained to predict the split decisions of a given depth level. A top-down approach is used to analyze the predictions: if the decision is to not split a CU in a given depth level, all predictions of the resulting CUs in the next level are ignored. For inter CU split prediction, additional MLP models are employed, which use the output of the intra CU splitting prediction of the current CTU together with previous split decisions. Soulef et al. (2019) adopted a similar solution, but all predictions are made by a single MLP model instead of separating them into different ones.

Wang et al. (2018) propose a solution for QTBT partitioning in VVC encoders. They propose to use a single CNN trained to predict the partitioning depth range for the current and the reference frame co-located CTU. The difference between the decision and prediction of the co-located CTU is used to relax the prediction of the CTU under evaluation. They try to control possible false prediction risks by considering the temporal

correlation between frames to adjust the predicted depth. Their algorithm evaluates the modified predicted depths of the four partitions with size 64×64 from a 128×128 block: if all partitions are predicted to have depth 0, all possible quad and binary partitions are restricted to a maximum depth of 2; if a single block has depth higher than 0, it will not be limited, whereas the others are early terminated; if more than one block is predicted with depth higher than 0, the 128×128 is split, and each 64×64 CU is evaluated within the respective predicted range.

The work of Galpin et al. (2019) is the only reviewed one that models block partitioning as a regression problem, checking the probability of a given block edge being part of the final QTBT partitioning for VVC encoders. An input block is divided into 4×4 non-overlapping blocks, and the splitting horizontal and vertical edges of those blocks – excluding the ones that are part of the input block edges – are associated with an element in a vector. Each element in that vector indicates the probability of the corresponding edge being part of the final partitioning. Considering a 64×64 block, there are 16 4×4 blocks and 15 vertical edges in a single row, totalizing 240 vertical edges in that block. The same reasoning can be applied to the horizontal edges, resulting in 480 edges to be predicted. A given split decision is only evaluated if the average probability, considering all edges associated with that split, surpasses a fixed threshold. The encoder can limit the search space by pre-determining adequate thresholds for different partitioning decisions. The authors also show results aiming to speed-up the encoding by restricting different combinations of depth levels and split decisions.

Although the block partitioning problem is independent of encoding mode, three works limit their scope to intra frame partitioning. Galpin et al. (2019), for example, trained their model only with intra frame data, but we did not find evidence that the same strategy cannot be applied to guide partitioning in inter frames. Huang et al. (2018) and Kuanar, Rao and Conly (2018) also propose intra-only termination strategies for the block partitioning problem. Because their main contribution is specific to the intra mode decision, we will present these two works in the following subsection.

Intra Mode Decision

The Intra candidate generation is designed to allow software- and hardware-friendly implementations (Sze; Budagavi; Sullivan, 2014). However, candidate selection may require executing the full RDO model, increasing the encoding complexity. In HM, an algorithm first evaluates all candidates using a simplified RDO model, and only a few are then selected for full RDO evaluation. Table 3 summarizes the BD-Rate and time reduction results from the works with relation to the baseline encoder.

Intra Mode Decision works use NN models to improve candidate selection, seeking to reduce encoding complexity with as little impact on coding efficiency as possible. The most frequently adopted approach to reduce the number of intra mode evaluations mixes

Table 3 – Intra mode decision related work results. All the works use AI configuration in their experiments.

Work	Encoder	BD-Rate (%)	Time Saving (%)
Huang et al. (2018)	SCM-8.3	1.36	49.33
Kuanar, Rao and Conly (2018)	HM-16.9	1.31	66.89
Song et al. (2018a)	HM-15.0	1.15	27.92
Laude and Ostermann (2017)	HM-16.6	0.52	-

CNN models with algorithmic solutions to classify a block, usually based on statistical analysis. Different works categorize blocks based on characteristics such as content or texture. Huang et al. (2018) proposes using a MLP to identify whether a block has screen content or natural camera-captured content. These two content types are so distinct that HEVC has an extension to optimize screen-content compression (Xu; Joshi; Cohen, 2016). One MLP model is trained to classify the block's content type at each depth level. The model is fed with hand-crafted features designed to analyze the block texture complexity. Less complexity indicates that the block is more likely to have screen content and present a homogeneous surface.

Kuanar, Rao and Conly (2018) propose a CNN to predict the block content category, separating them into static and dynamic with a large object, granular texture, or small object. The CNN model was designed to identify rectangular regions of interest containing objects. In addition to regions of interest, the fully connected layers receive texture features for the classification. As any intra mode decision work, the number of verified modes reduces to 4, 6, and 8 for the static, dynamic with a big object, and dynamic with a granular texture categories. As any block partitioning work, the depth search is terminated if the block is static.

Song et al. (2018a) solution only works with 4×4 and 8×8 input blocks, categorizing them before CNN processing. They quantify the vertical and horizontal variations of Intra reference samples and compare the result against fixed thresholds. The comparison result is a label that can be flat, weak, or strong. In the case of a flat block, only DC and planar modes are evaluated. For the other two, specific CNN models predict each Intra mode probability to be selected, and then only the eight most probable modes are chosen for RDO evaluation. The eight selected modes may be further reduced to 5 if they detect a corner shape inside the block. Unlike the aforementioned works, Laude and Ostermann (2017) do not perform any classification. Instead, a CNN model gives the probability of each mode to be selected, considering only the input block data. The approach is similar to Song et al. (2018a)'s, but they use a more complex CNN rather than pre-evaluating the original block characteristics using a hand-crafted algorithm. Even though Laude and Ostermann (2017) do not present complexity reduction results, they state that the

proposed method shows a slight loss in compression efficiency without the necessity of RDO evaluations.

4.2.2 Coding Efficiency Works

In this subsection, we present works that propose solutions to improve coding efficiency, which are divided into Non-normative (4.2.2.1) and Normative (4.2.2.2).

4.2.2.1 Non-normative Approach

Rate Control

While all other selected works in this systematic literature review use the QP for encoding under quality constraints, we found a single work using the NN model for encoding under bitrate constraints. The most naïve method for rate control is to encode each frame varying QPs until the target bitrate is achieved. A better approach relies on a model to predict the QP that achieves the target bitrate. Quality may be prioritized over the bitrate constraint if the model is too optimistic. If the model is too pessimistic, the bitrate constraint may be maintained but with unnecessary quality degradation.

Sun et al. (2018) propose a method using a MLP to select the QP for rate-constrained compression. The authors show that a linear model is inaccurate in defining the QP as a function of the target rate. They then present a second-order model to solve the problem. The parameters of that model are content-related and calculated by the proposed MLP, which takes 14 hand-crafted features extracted from encoded frames as input. The authors report that 90.9% and 72.3% of the total samples are within 20% and 10% of the bitrate error, respectively. Although the authors report improvements over the linear model, they do not specify the encoder or baseline configurations.

Static Resolution Scaling

Super-resolution is an application of CNN that takes a low-resolution image and increases its spatial resolution. Although initially applied to images, this approach has already been adapted for video content. All works in this group can generate a bitstream compatible with the target standard and do not require additional signaling. The solution predefines the scaling options, which do not change during the encoding process (they are static). Moreover, although the solutions may include additional operations before and after decoding, the bitstream is still non-normative. Table 4 summarizes the BD-Rate and complexity results from the works in comparison to their baseline encoder.

These solutions can be categorized by resolution domain (spatial, temporal, or bit-depth) or processing steps (with or without a sub-sampling process). The classification by resolution domain is self-explanatory, but it is important to note that a single

Table 4 – Static resolution scaling related work results.

Work	Encoder	Configuration	BD-Rate (%)	Encoding Complexity	Decoding Complexity
Feng et al. (2018a)	HM-16.17	LDP, QPs 30-44	-31.48	-	-
Zhang, Afonso and Bull (2019)	HM-16.20	RA, 10 bit	-6.40	1.02×	69.3×

work may address multiple domains because they are not mutually exclusive. Regarding processing steps, the sub-sampling and non-sub-sampling approaches have distinct goals. Without sub-sampling, post-decoding processing aims to upscale the original video resolution. When sub-sampling is applied pre-encoding, the decoding process aims to reconstruct the original scale. The latter approach naturally reduces bitrate and complexity and may yield better quality due to enhancement capabilities learned during training (see the Quality Enhancement category for further discussion).

Kappeler et al. (2016) propose a method using CNN to increase the spatial resolution of an frame. The CNN model takes multiple consecutive frames as input to explore spatial and temporal video characteristics. A separate branch of the model processes each frame, and the outputs are concatenated and refined through two additional convolutional layers. They also use optical flow prediction and compensation to align the input frame content with the target output frame. Because the authors evaluate their method using only one video sequence, this work was not included in Table 4.

Feng et al. (2018a) state that if spatial resolution is sub-sampled before encoding, then a single CNN model struggles to handle distortions introduced by both sub-sampling and encoding when performing the upsampling process. To address this, the authors propose a two-stage approach: one CNN to enhance the decoded frame, followed by a second one to restore the original resolution. After decoding a frame, eight variations are generated by rotating and flipping the decoded frame. The frames are then processed by a CNN, and the outputs are fused to generate the enhanced frame. Another set of eight augmented frames is generated from the enhanced frame and processed by a super-resolution CNN. Finally, the outputs are fused to produce the final high-resolution frame.

Kim et al. (2017) perform sub-sampling on spatial resolution. Even though the authors claim that their method is dynamic, the experimental results were presented in the HEVC context without demonstrating dynamic resizing. The authors show that the sub-sampling method is more effective when implemented with a CNN model than hand-crafted methods. The authors designed a loss function with three terms to improve the sub-sampling with the CNN model. The first term compares the sub-sampled frame to the original, checking for structural integrity preservation. The second term evaluates the variations within the generated frame because frames with low sample variations may improve compression efficiency by reducing high-frequency components. The third term reflects the reconstruction loss after upsampling, capturing the actual error introduced by the encoding process. Both networks are simultaneously trained because they are

complementary, and the error from one has implications for the other. Kim et al. (2017) reported a 51.7% improvement in quality based on the VMAF metric. The VMAF metric proposed by Netflix uses machine learning to approximate perceptual visual quality assessment. (Li et al., 2016). As this is the only selected work using VMAF, we chose not to report VMAF results in the comparative analysis.

Zhang, Afonso and Bull (2019) is the only work that performs sub-sampling and restoration in the bit-depth domain. Before encoding, each sample has its least significant bit removed. The encoding and decoding process operates with fewer bits, and a CNN model is used to restore the original bit depth representation. The proposed CNN model processes a frame in overlapping blocks of size 96×96 with 4-pixel overlaps in both rows and columns.

Feng et al. (2018a) use the QPs from 30 to 44 instead of the CTC, which recommends QPs from 22 to 37. Other works using NN solutions also report results for various QP ranges, with better results at higher QPs. Because higher QPs lead to poorer visual quality, NN solutions to improve quality become more effective in such a scenario.

Quality Enhancement

During the encoding, the quantization step reduces the precision of the transformed coefficients, adding error to the encoded sequence in exchange for bitrate reduction. The quality enhancement category of solutions applies CNNs after the decoding process to improve video quality. The solutions neither change the standard signals nor require additional processing before the encoding. Therefore, they are compatible with existing standards and encoded bitstreams. Note that our selection excludes works aiming to recover information from encoded sequences, such as reconstructing a face from a low-resolution image, which falls within the scope of forensics. Table 5 summarizes BD-Rate and complexity results from the works relative to their respective baseline encoder.

Wang, Chen and Chao (2017), Li, Tan and Yan (2018), Zhao et al. (2020), He et al. (2018), Feng et al. (2019), and Yang et al. (2019) present similar strategies for quality enhancement. They all design a CNN model trained to learn the residue between the reconstructed and original frames. Zhao et al. (2020) design a CNN model using inception units and separating the enhancement of luma and chroma frames. He et al. (2018) present a wide range of CNN design explorations. Their best CNN design uses a frame mask created by averaging the samples of each block partition. This mask is then merged with reconstructed frame features through element-wise addition.

Feng et al. (2019) present a novel solution that does not focus on specific CNN design decisions but rather on the model's input. Instead of using only the decoded frame data, the authors also use the data from the predicted and reconstructed frames (before in-loop filtering). The authors state that these frames provide useful information for quality enhancement that is not considered by the in-loop filters and are lost during the

Table 5 – Quality enhancement related work results. The BD-Rate results from Zhao et al. (2020) were weighted from Y, U, and V results and this work applies different trained models for luma and chroma channels. For the other works, we only report the results from luma channels, but they are even better for chroma channels.

Work	Encoder	Configuration	BD-Rate (%)
Wang, Chen and Chao (2017)	HM-16.0	AI	-5.0
		LDP	-6.4
		LDB	-5.3
		RA	-5.5
Li, Tan and Yan (2018)	HM-16.19	AI	-5.5
Zhao et al. (2020)	HM-16.9	AI	-8.9
		LDP	-11.8
		LDB	-13.1
		RA	-10.3
He et al. (2018)	HM-16.0	LDP	-9.7
Yang et al. (2019)	HM-16.0	RA	-8.3
		RA, 480p@30Hz, real-time	-6.8
		RA, 720p@60Hz, real-time	-2.8
Wang et al. (2018)	HM	AI	-6.5
		LDP	-8.0
		LDB	-6.4
		RA	-6.7

decoding process. Although non-normative, the solution is limited to the HEVC standard and introduces intrusiveness in decoding implementations.

Unlike the other works that do not consider the frame type, Yang et al. (2019) design separate CNN models to enhance the quality of intra and inter frames. First, they train a model to enhance the quality of intra frames. The features from that model's layers are then merged with those from the inter-frame enhancement model. It is also important to highlight that Yang et al. (2019) are the only ones to address CNN complexity and decoding constraints, proposing a method for real-time decoding. They calculate the number of blocks that can be enhanced under a given time constraint. For intra frames, state that the quality enhancement correlates with the number of bits allocated to represent a block. The higher the bit allocation, the more effective the proposed solution is. For inter frames, the decision is based on the mean absolute deviation of each block. Blocks with higher mean absolute deviation values are prioritized for enhancement. The authors present a look-up table specifying the number of intra and inter blocks to be enhanced considering different target complexity reduction rates.

Although Yang et al. (2019) propose a method for inter frame enhancement, as the others, their process still relies on data from a single frame. Soh et al. (2018) state that using data from individual frames alone makes it impossible to remove motion-related ar-

tifacts. They propose to enhance a frame using samples from previously and subsequently decoded frames. They apply a simple ME algorithm to neighboring frames to select the best blocks for enhancement. They evaluate the mean absolute difference between the current and ME-selected blocks to avoid unreliable data due to drastic frame-to-frame changes. If the value exceeds a fixed threshold, the block is replaced by the samples of the one to be enhanced. The CNN processes each block in separate parallel branches, which are then concatenated and refined using two inception units. To avoid blocking artifacts, their framework enhances blocks with overlapping regions. The samples from overlapping regions are reconstructed using different weights to smooth the transitions.

Wang et al. (2018) also use multiple frames in the enhancement process, but they propagate the information from previous frames using Long Short-Term Memory (LSTM). To explore object scaling redundancies (i.e., variations in object size across frames), the authors concatenate the current and stored data from the previous frames and process them in three different convolutional branches, each with a different number of layers (similar to the inception unit). The output of each branch is concatenated and processed with a single convolutional layer to generate the enhanced frame.

Tong et al. (2019) propose to separate the frames of a GOP into three categories: intra frames are called high-quality frames, the frames that reference only high-quality ones are called medium-quality frames, and the remaining are called low-quality frames. High- and medium-quality frames are enhanced with a CNN model that, as in previously cited works, receives a frame as input and aims to reduce the error relative to the original frame. Their approach is novel because of the method they use to enhance low-quality frames, which can be summarized as follows. First, a CNN model generates two optical flows: one from the high-quality and one from the low-quality frame. Then, two compensated frames are generated. The high-quality and medium-quality compensated frames are then used in a different CNN model to enhance the low-quality frame.

Although the works report significant improvements in coding efficiency – from -5.0% up to -13.1% –, Yang et al. (2019) were the only ones to show a strategy for real-time decoding. Because this class of works directly impacts the decoding complexity, we notice little effort to integrate NN solutions with real-time decoding constraints. The exploration of NN methods for quality enhancement under real-time constraints remains an open research topic. Studies on that scope can present BD-Rate results in a more realistic scenario and provide insights for NN development on constrained platforms – such as mobile devices.

Block Matching

The candidates are usually evaluated with the conventional RDO model in the block-matching context. Because candidate selection relies only on cost minimization, it can only select a local optimum. At first glance, this may not significantly affect the

current block prediction. However, this selection can negatively influence the prediction of subsequent blocks. Choi et al. (2017) highlight this problem in the inter frame interpolation process. They state that the RDO model cannot accurately track object motion across a sequence of frames. In their proposed solution, a CNN model receives two blocks and predicts a matching score between them. The authors claim that the CNN-based block matching can produce more reliable MVs for the inter frame interpolation process. Choi, Heo and Park (2019)'s work frame interpolation over previous methods in the field, although it adopts the same CNN application strategy. These works may be categorized under the resolution scaling category. However, we argue that CNN-based block matching is more flexible since it can be applied to conventional IME algorithms or frame rate-up conversion strategies. These represent inter-sample interpolation methods and are part of normative solutions.

4.2.2.2 Normative Approach

Intra Sample Prediction

Some works propose NN models to predict the block samples as an alternative to compete with standard prediction modes. Because it is necessary to signal extra modes, standard definitions must be changed to support them. One might reuse existing standard signals, but a standard decoder can still not reproduce the video correctly because the reconstruction process changes. Therefore, sample prediction using NN solutions falls into the normative group of the proposed taxonomy. Table 6 summarizes the BD-Rate and complexity results from the works relative to their baseline encoder.

Table 6 – Intra sample prediction related work results.

Work	Encoder	Configuration	BD-Rate (%)	Encoding Complexity	Decoding Complexity
Li et al. (2018b)	HM-16.9	AI	-1.10	1.48×	2.90×
Li et al. (2018a)	HM-16.9	AI	-2.30 -3.40	3.08× 91.47×	8.13× 230.11×
Pfaff et al. (2018)	JEM-7.0	AI QPs 22-37 AI QPs 27-42	-3.01 -2.99	2.18× 2.41×	2.48× 2.86×
Helle et al. (2019)	VTM-1.0	AI	-3.79	2.84×	1.47×
Li et al. (2018a)	HM-12.0	AI	-3.10 (U) -2.00 (V)	- -	- -

Analogous to conventional intra sample prediction, solutions falling into this class can only rely on already encoded samples within the frame being encoded to generate new candidates. Li et al. (2018b) propose a MLP to predict the samples using the same reference samples of conventional intra mode prediction. In their proposal, the generated

reference is an additional option to the ones already tested by the HEVC encoder. Li et al. (2018a) extend the evaluations separating angular intra modes from DC and planar ones to train two models. They show that prediction can be improved using two different models. Pfaff et al. (2018) show a similar approach as Li et al. (2018b), but they increase the number of reference samples and present a strategy to generate each mode efficiently. The difference between each mode lies only in the set of weights and biases of the output layer, which reduces the complexity by avoiding recalculations within the model. The new intra modes are also complementary to HEVC ones, but since the number of modes increases too much, the signaling in the final bitstream became a problem. Thus, they also propose a method to efficiently signal the decision using a second neural network at the decoder side. Helle et al. (2019) notice that Pfaff et al. (2018)'s proposal could be simplified by predicting the Intra mode in the frequency domain. The simplification was possible because, based on statistical analysis, they decided to zero the frequency components that do not significantly improve the coding efficiency.

Li et al. (2018a) propose to predict chroma samples using neighboring luma and chroma reference samples and a sub-sampled reconstructed luma block. A MLP receives luma and chroma reference samples, and the linear output of the model is reorganized as an frame. A different branch processes the sub-sampled luma block using two convolutional layers. Both outputs are merged using the element-wise product, and another convolution layer predicts the final chroma samples.

Inter Sample Prediction

Applying intra techniques to an inter frame is possible, but this strategy does not exploit temporal redundancies. We found three different approaches that use CNN models to predict inter samples by exploiting temporal redundancies. Table 7 summarizes the BD-Rate and complexity results of the works in comparison to their baseline encoder.

Table 7 – Inter sample prediction related work results.

Work	Encoder	Configuration	BD-Rate (%)	Encoder Complexity	Decoding Complexity
Zhao et al. (2018)	HM-16.15	RA	-3.0	1.65×	44.70×
		LDB	-1.6	-	-
Zhao et al. (2018)	HM-16.6	RA	-3.2	-	-
Zhao et al. (2019)	HM-16.6	RA	-5.5	-	-
	JEM-7.1		-0.7	-	-

Zhao et al. (2018) present a method to enhance bi-prediction. The handcrafted method calculates the bi-prediction block as an element-wise average of two uni-predicted blocks. The authors argue that a CNN model can improve coding efficiency by calculating each sample as a non-linear combination of samples from both uni-predicted blocks.

In another work, Zhao et al. (2018) propose an inter prediction mode based on virtual reference frames. A CNN generates the virtual frames using blocks from a past and a future frame related to the one being encoded. The novel mode uses only the co-located block from the virtual frame for RDO evaluations, comparing it with conventional inter prediction modes. In (Zhao et al., 2019), the authors expanded the latter method by proposing an enhancement process that uses a CNN to improve the virtual reference frame.

Jimbo, Wang and Yashima (2019) propose a different method that does not predict samples directly but rather transformation matrices. The transformation matrices allow parallel translation and scaling operations[2] with arbitrary reference sample accuracy, enhancing the prediction process. The CNN model receives one block from a past frame and one block from a future frame and predicts four transformation matrices: two per frame block. The final candidate is calculated as the average of the transformed blocks and evaluated against the standard mode prediction to select the best candidate. Jimbo, Wang and Yashima (2019) claim that the proposed model is selected more often than traditional ones, but they do not objectively evaluate coding efficiency and hence such work was not included in Table 7.

Fractional Position Sample Prediction

The predictions presented in the previous subsection operate on integer position samples, aiming to minimize the residue relative to the original block. The NN models have a straightforward objective: the residue must be as close to zero as possible, meaning that the NN predicts the block to be as similar as possible to the one from a raw frame. The challenge with fractional position samples is in defining the expected output. Suppose the problem is formulated to minimize the residue in the same way as intra mode or integer position sample predictions. In that case, it leads to the same objective and thus becomes redundant. Moreover, a NN cannot be designed to generate samples using the expected standard filters. Even though handcrafted proposals use signal processing theory (Yan et al., 2018), no definition exists for what constitutes a fractional position sample value outside of a standard. We have found three different solutions to solve the fractional position sample problem:

- Sub-sample frames with a blurring process;
- Interpolation filters with invertibility property;
- Redesign the concept of the fractional prediction process.

Table 8 summarizes the BD-Rate and complexity results from the works relative to their respective baseline encoders

[2] The proposal is related to affine inter prediction of VVC.

Table 8 – Fractional sample prediction related work results. Yan et al. (2019b) tested their modes without the standard filters and as an additional prediction mode to be compared with them. Yan et al. (2019a) show a broad range of CNN designs and evaluate the complexity of each one for LDP configuration, but we only report the one with lower complexity impact.

Work	Encoder	Configuration	BD-Rate (%)	Encoding Complexity (%)	Decoding Complexity (%)
Yan et al. (2017)	HM-16.7	LDP	-0.9	-	-
Xia et al. (2018)	HM-16.15	LDP	-1.9	-	-
Liu et al. (2019)	HM-16.4	LDP	-2.2	-	-
		LDB	-1.2	-	-
		RA	-0.9	-	-
Yan et al. (2019b)	HM-16.7	LDB w/o standard filters	-2.7	-	-
		LDB w/ standard filters	-2.9	-	-
		RA w/o standard filters	-2.9	-	-
		RA w/ standard filters	-3.6	-	-
Yan et al. (2019a)	HM-16.7	LDP	-3.9	-	-
		LDB	-2.7	-	-
		RA	-1.3	-	-
		LDP, low-complexity	-2.2	3.58×	15.53×
Zhang et al. (2019)	HM-16.7	LDP	-0.9	-	-

Among the selected works, Yan et al. (2017) was the first one to use high-resolution, blurred frames to generate fractional position samples. They use a low-pass filter to blur the frame, and three half-pixel fractional positions are taken from that frame by sub-sampling pixels at different positions. They train three models, each one specialized to predict samples of a specific position. During inference, a model receives an integer position sample block and predicts the fractional position samples. The predicted candidates are evaluated following the same execution flow as the HM. Xia et al. (2018) use that same strategy to generate fractional position samples. The blurred frame is sub-sampled in 16 positions, generating 16 candidate blocks. It is important to note that sub-sampled frames will have 1/16 of the original frame resolution. The authors also sub-sample the original frame before encoding to maintain proportional resolution consistency, allowing them to use the discarded integer position as training references. They design the CNN to have the first layers shared between all the 16 fractional position sample variations, and only the last layers differ to generate samples at specific positions. Liu et al. (2019) continue the exploration of Xia et al. (2018)'s work by presenting a mathematical formulation to analyze the theoretical impact of different blur kernel sizes.

Yan et al. (2019b) address the fractional position sample problem presenting a CNN-based interpolation filter with invertibility property. The invertibility property allows the same filter that generates fractional position samples to recover the integer position ones. With this property in mind, the authors propose an end-to-end model to generate fractional position samples by learning from the error at the integer position domain. They have made two decisions to ensure that the CNN interpolates fractional position samples. The first is to incorporate, in the loss function, the error from conventional interpolation filter samples (like the ones used by the HEVC standard) in addition

to the error based on the integer position samples after converting the fractional ones into integer ones. The second decision consists in defining the whole flow as follows: $T^{-1}(F(T(F(can))))$, where T, T^{-1}, and F represent a geometric flipping transform, its inverse, and the CNN interpolation model, respectively. They trained three models by changing only the geometry flipping to generate different fractional position samples.

Yan et al. (2019a) redefine the concept of FME in video compression. Rather than focusing on generating samples at fractional positions, they view FME as a set of functions designed to reduce distortion between the integer position candidate samples and the original block. Their formulation is generic enough to comprise standard fractional interpolation methods but also offers the flexibility needed to overcome the limitations of handcrafted methods. The authors implement each function as a CNN. They propose a single CNN model that aims to predict the samples of the original block. They varied the integer sample positions used during training to produce different interpolation functions (15 in total). With this method, the models learn distinct weights and biases even with identical architecture and training targets, resulting in different outputs. In bi-predicted blocks, the first reference is selected from the set of uni-predicted candidates. The second reference is obtained using another set of functions. The target output for these functions is based on the difference between the original block – weighted by a factor of two (due to arithmetic operations on the average of the two references) – and the first reference. Zhang et al. (2019) adopted an approach similar to that of Yan et al. (2019a), but their CNN design differs by accepting two distinct inputs: the predicted samples and the residue. Each input is processed through separate branches, whose outputs are concatenated and then passed through additional convolutional layers to produce the final output.

Transform Step

Methods applying NN in the transform step are barely explored in the context of video compression, and further research can be conducted in future work. Lee et al. (2018) propose replacing the transform step and its inverse with MLPs. Similar to handcrafted methods, the model transforms spatial samples into frequency-domain representations, and the same model can perform the inverse transform. The MLP design is simple, but the main contribution lies in the loss function formulation. They add a regularization term to the loss function using different weights for various transformed coefficient errors. The key idea behind the regularization is to guide the NN model to learn how to compact energy, similar to the DCT transform. The top-left coefficients have small weights, while the bottom-right ones have large weights, causing the network to concentrate the energy at the top-left coefficients. A second regularization term induces an orthogonal transform, which generates fewer non-zero coefficients in the frequency domain, thus leading to a higher compression rate. The authors also propose to change the quantization step to perform uniform quantization.

Loop Operations

Works related to loop operations share principles with those focused on quality enhancement. Both provide solutions to improve video coding efficiency by processing reconstructed frames, aiming to alleviate distortion caused by quantization. However, solutions involving loop operations modify the frames used in future predictions, which affects the compression process and alters the codec's standard behavior. Table 9 summarizes the BD-Rate and complexity results from the works relative to their respective baseline encoders.

Table 9 – Loop operations related work results. Some works do not summarize the results and thus, we present only the best and worst cases.

Work	Encoder	Configuration	BD-Rate (%)	Encoding Complexity		Decoding Complexity	
				CPU	GPU	CPU	GPU
Park and Kim (2016)	HM-16.0	AI LDP RA	-4.8 -1.9 up to -2.8 -1.6 up to -2.6	-	-	-	-
Kang, Kim and Lee (2017)	HM-16.7	AI	-8.5	-	-	-	-
Song et al. (2018c)	JEM-7.0	AI, ALF on RA, ALF on	-3.57 -1.23	1.09× 0.99×	93× -	128.87× 2.84×	1.57× -
Yeh et al. (2018)	HM-16.12	AI	-2.8 -5.1	1.23× 12.54×	- 1.98×	- -	- -
Zhang et al. (2018)	HM-12.0	I frames P frames B frames	-4.1 up to -7.8 -5.54 up to -7.6 -3.4 up to -5.7	26.4s up to 1094s	1.56s up to 64.6s	-	-
Chen et al. (2019)	AV1 jan	Intra Inter	-10.03 -4.10	- -	0.78× up to 1.13× 0.94× up to 0.99×	- -	- -
Jia et al. (2019)	HM-16.9	AI LDP LDB RA	-4.1 -4.7 -6.0 -6.0	-	1.13×	-	116.56×
Yu et al. (2019)	HM-16.12	AI LDP LDB	-6.7 -7.8 -7.6	5.28× 2.1×	1.28× 1.22×	186.89× 240.21×	12.72× 21.86×
Li et al. (2019b)	HM-16.5	RA	-11.62	-	-	-	-
Zhang et al. (2019)	HM-16.18	AI LDB RA	-2.8 -2.9 -3.0	-	-	-	-
Li et al. (2018b)	HM-12.1	AI AI, QPs 32-47 AI w/ standard filters, QPs 32-47	-2.1 -1 up to -12.7 -1 up to -9.6	7.47×	- -	250.21×	-
Lin et al. (2019)	HM-12.1	LDP, QPs 32-47 LDB, QPs 32-47 RA, QPs 32-47 RA	-3.5 up to -6.4 -2.6 up to -5.1 -3.8 up to -6.8 -1.3 up to -4.4	2.93× - 3.22×	1.55× - 1.75×	266.09× - 326.27×	17.94× - 21.55×

Park and Kim (2016) propose using a CNN to further improve the quality of references after the conventional in-loop filtering step. The authors state that their model was designed based on super-resolution NNs, using high-dimension convolutional kernels (9×9 in their work). Instead of adding an enhancement step after in-loop filtering, Kang, Kim and Lee (2017)'s solution replaces existing in-loop filters by a CNN with three stages. For the first stage, they create a frame mask based on block partitioning boundaries. Convolutional layers process that mask before being merged with the reconstructed frame. The second stage performs coarse-grain restoration by convoluting the first stage output

at half resolution. The first and second-stage outputs are concatenated and then processed at the original scale in the third stage, which performs fine-grain restoration.

Song et al. (2018c) also propose replacing existing in-loop filters for intra frame reconstruction only. The authors address the problem of model representation size and floating-point operations. They add regularization terms to the loss function, aiming to reduce network weights' values, which are more likely to become zero. They handle the floating-point operation problems by quantizing and clipping them to a fixed bit-width representation.

Yeh et al. (2018) propose two CNN solutions to improve the quality of intra frames. The first model processes one CTU at a time and aims to be complexity-aware. The model itself is simple, but they propose an algorithm aiming to reduce even more the complexity by selecting which CTUs should be enhanced with the model. The selection method can be dynamically adjusted based on complexity constraints. In contrast, the second model was designed to achieve higher coding efficiency, being applied to all CTUs. The authors propose a heterogeneous CNN architecture to improve the performance by combining Graphics Processing Unit (GPU) and Central Processing Unit (CPU) in the execution flow with pipelining techniques.

To replace in-loop filtering with CNN models, Zhang et al. (2018) performed experiments combining different CNN layer depths and highway connection configurations, measuring the impact across different QP values and frame types. Chen et al. (2019) presented a similar approach, but they define a pre-determined subset of frames for CNN filtering. They selected frames referenced by most other frames in a GOP, maximizing the enhancement effect without applying CNN filtering to the entire sequence.

Jia et al. (2019) main contribution is a content-aware CNN model for in-loop filtering. Multiple CNN candidate models are trained, and each CTU is then filtered with the model with the best results. In this way, each CNN model becomes specialized to filter a type of content better. The authors propose a CNN model trained to predict the best CNN model for a given block to avoid additional signalization of the model that must be selected for reconstruction.

Yu et al. (2019) state that flexible partitioning structures like quadtree on HEVC generate similar artifacts – such as block artifacts – at multiple scales. Hence, it is important to consider the multi-scale similarity of artifacts or similar artifacts at multiple scales. To address this issue, the authors propose a CNN model combining two different structures in the design: multi-reconstruction branches and recursive residual paths – this structure is referred to as an inception unit in this work. Since the model operates on the encoder side, the authors highlight the relevance of reducing its complexity. They propose to reuse parameters in different parts of the CNN model and use small convolution kernels, with the largest (5×5) used only in the first convolution layer. Unlike the previous works, Yu et al. (2019) provide no information about where the CNN operates in the encoding flow.

Li et al. (2019b) also propose to use multiple CNN models, but these models can be selected as additional modes competing with handcrafted ones. Each model is specialized to explore either temporal or spatial correlations. For temporal exploration, each reference frame is evaluated using six metrics, and the most similar ones relative to the frame being enhanced are selected as reference frames. A CNN model processes each reference frame for motion compensation. The following convolutional layers are similar for both temporal and spatial models. The original frame, a compensated reference frame, and a mask representing the block partitioning scheme are concatenated. The concatenated result of each reference frame is processed in parallel branches, and the outputs are concatenated – a step not required in the spatial model. Then, additional convolutional layers generate the filtered version of the input frame. The Temporal and spatial CNNs are very simple, but the model used for motion compensation was built similarly to the one presented by Kang, Kim and Lee (2017). The input is first processed with a quarter resolution, then at half resolution, and finally at full resolution to generate the motion-compensated frame.

Balaji and Thyagharajan (2018) propose using 3D filters combined with MLP models for in-loop filtering in the scalable HEVC extension. During the in-loop filtering step, the frame is denoised by applying 3D filters to stacks of similar 2D blocks from different frames. In the proposed flow, the filtering process is repeated until a given PSNR threshold is achieved, followed by the backpropagation algorithm (i.e., online learning). Zhang et al. (2019) also use 3D filters. First, the authors propose dynamically defining the GOP size while encoding. The key idea is to group successive frames that are more strongly correlated, which can improve the 3D filtering process. Frames are grouped based on the absolute standard deviation of the residue. If the value surpasses a threshold, no more frames are added to the group, and a new one is created. The group is then processed with a Haar filter to separate the low- and high-frequencies of the frames. Four CNN models were trained, each designed to map a combination of low- and high-frequency frames. There is an inverse Haar filtering process to generate the filtered frame.

Works proposing modification on loop operations mainly relies on in-loop filtering solutions using CNNs. However, Li et al. (2018b) propose a technique to upsample intra blocks encoded at lower resolution to improve coding efficiency. Although their work is not directly related to the previously presented approaches, we include it in this category because the solution acts as additional processing before the in-loop filtering operation. A block can be encoded in the proposed framework at full or sub-sampled resolution. To ensure consistent resolution, a subsampled block must be upsampled in the reconstruction loop. The block upsampling can be performed using the CNN model proposed by the authors or a DCT-based interpolation filter. The filter that minimizes the RDO cost is then chosen for reconstruction. After reconstructing the whole frame, an additional filter uses the samples around each sub-sampled block to improve frame quality. Li et al. (2018b) have also conducted experiments to select adequate encoding parameters for full

and low resolutions. Lin et al. (2019) propose three CNN models for block upsampling, expanding Li et al. (2018b) to inter blocks. The first one performs the reconstruction using only the samples from the low-resolution block. The second one uses additional samples from low- and full-resolution blocks on a single reference frame. The third one shares the definition from the second model but uses two reference frames. They present necessary adjustments to derive parameters for merge candidate list construction correctly, MV and QP setting.

Loop operation solutions suffer from the same problem as Quality Enhancement: they negatively impact decoding complexity. On top of that, loop operations are normative, which is challenging to adopt in practical applications. We observe that NN can provide better filtering coefficients than those used in in-loop filtering due to training on large-scale data samples. However, the filters must be standardized and designed to support efficient implementation in both software and hardware.

Entropy Coding

Arithmetic coding has the potential to achieve optimal compression, but it requires accurate probability predictions for the symbols found in a sequence. CABAC was optimized with handcrafted binarization and empirically defined context models. Ma et al. (2018) question the accuracy of the handcrafted approach. To improve the compression, Ma et al. (2018) propose replacing the conventional binarization and context modeling with a CNN model that predicts the probability distribution of the intra DC coefficient. The model is fed with neighboring reconstructed blocks and already encoded AC coefficients of the current block. Ma et al. (2018) show a BD-Rate reduction of -1.6% of using HM-12.0.

Song et al. (2018b) also have concerns about the handcrafted approach adopted in CABAC and propose a CNN-based solution to estimate probability distributions as well, but for intra mode coding. Similar to Ma et al. (2018), the authors use neighboring reconstruct blocks in convolutional layers. In the fully connected layers, they add information about the most probable mode from HEVC encoding. The network predicts the probabilities of each intra mode used by the arithmetic coding process along with the syntax elements.

Puri, Lasserre and Le Callet (2017) state that prior works present substantial gains by using different transform matrices in the transform step, allowing the possibility of selecting the best one to compact the residue energy. However, this approach demands additional signalization to index the selected transform matrix among the possible ones. Puri, Lasserre and Le Callet (2017) propose a CNN to predict the probability distribution of each transform index based on the coefficients after transform and quantization. The model prediction defines a variable-length code in the binarization process, aiming to

reduce the number of bits necessary to signal the decision. Puri, Lasserre and Le Callet (2017) show up to -1.76% BD-Rate using HM-15.0.

This category of work faces three main problems. The first two are the negative impact on decoding performance and the challenge to standardize NN solutions. The most critical issue is that all proposals target the compression of specific syntax elements. This implies that multiple models may be necessary to enhance the compression for different symbols, further complicating the standardization problem. Even with the adoption of NN, the limited exploration in this area prevents us from identifying which symbols are best suited for such solutions.

Dynamic Resolution Scaling

This work category shares similarities with static resolution scaling but requires modifications to the standard to support dynamic decision signaling. Even if the bitstream remains decodable by a standard, it must be encapsulated with additional signaling and processing to reconstruct the final video. Table 10 summarizes the BD-Rate and complexity results from the works relative to their baseline encoder.

Table 10 – Dynamic resolution scaling related work results.

Work	Encoder	Configuration	BD-Rate (%)	Encoder Complexity	Decoding Complexity
Afonso, Zhang and Bull (2019)	HM-16.14	RA	-14.5	-	-
Afonso, Zhang and Bull (2018)	HM-16.18	RA	-11.8	-51%	5.3×

Afonso, Zhang and Bull (2018) propose dynamically adjusting the sub-sampling scale and QP to improve encoding performance. The solution uses a handcrafted sub-sampling method. After decoding, the frame is first upsampled and then processed by a CNN model. In the proposed framework, spatial and temporal features are extracted from raw frames and, along with a base QP, are used to decide the best trade-off between QP range and frame resolution. If a lower resolution is chosen to improve the coding efficiency, the QP is adjusted accordingly. The key idea is that better rate-distortion performance can be achieved by reducing both the QP and the resolution (i.e., producing smaller frames of higher quality) rather than encoding full-resolution frames with a higher QP (which results in larger but lower-quality frames). A decoded frame is scaled to the original resolution and then processed by a CNN model to improve quality. The authors report a BD-Rate of -11.8% using RA configuration on HM-16.18. The encoding complexity decreases by 51% while decoding complexity increases by a factor of 5.3×.

Afonso, Zhang and Bull (2019) propose a framework that sub-samples both spatial and temporal video resolutions before encoding and performs upsampling after decoding. Temporal scaling uses a visual quality metric (Zhang; Mackin; Bull, 2017) instead of a NN-based method. If removing a frame results in an insignificant visual impact, the

frame is dropped from the sequence. As for spatial scaling, the authors use a statistical approach to define an adaptive QP threshold. If the input QP exceeds the predicted threshold, spatial sub-sampling is applied, and the QP is reduced accordingly, aiming to maintain a similar bitrate as if no sub-sampling had been applied. On the decoder side, temporal upsampling is performed by averaging neighbor frames and spatial upsampling using a CNN model specifically trained with HEVC compressed frames. The authors report a BD-Rate reduction of -14.5% using RA configuration on HM-16.14, but no complexity results are provided.

The Dynamic Resolution Scaling category of solutions competes with Static Resolution Scaling ones. Dynamic scaling can better adapt to scene changes throughout a video sequence. However, dynamic approaches require standardization, whereas static solutions can produce bitstreams conforming to existing standards. If a future standard incorporates dynamic resolution scaling, NN-based methods could be widely adopted for performing sub-sampling and upsampling operations.

4.2.3 Learning-based Coding Models

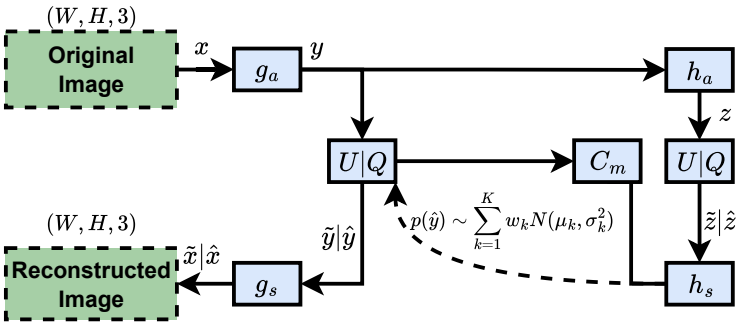
This category of work proposes a compression model entirely based on learning strategies. Li and Trocan (2018) propose an enconder for single-pixel prediction using MLPs combined with conventional entropy encoding. There are three prediction modes: intra, inter, and multiview. As the work in this thesis does not focus on multiview video compression, we present details only for the first two modes. In intra prediction mode, the first pixel is copied directly since no prior information is available to predict it. Following a raster-scan order, the remaining pixels are predicted based on all previously encoded pixels. For a frame with N pixels, $N-1$ MLP models are used to perform the predictions. In inter prediction mode, since at least one reference frame is available, all MLP models use this frame as input to predict the samples of the next frame. In this case, there are N MLP models, one to predict each sample position of the next frame. For both intra and inter prediction, the output stream encodes the residues between the original frame and the predicted samples. The proposed model outperforms HEVC in terms of both quality and compression rate, although the execution time increases by a factor of 2.17×.

Despite the challenges in standardizing a solution entirely based on NN, this category of work plays an essential role in academic research. These approaches explore alternative paradigms that test the capabilities of NN-based methods for video compression, revealing significant potential for novel solutions in this area.

4.3 SINGLE- AND MULTI-LAYER COMPRESSION

As observed in the presented systematic review and taxonomy, many studies have introduced learned models into handcrafted video codecs. However, only a few have pro-

Figure 27 – Cheng et al. (2020) learned image codec proposal. Module g_a transforms the image to the latent space y. On inference, the quantization Q produces a signal $\hat{y}$. In the training stage, the uniform noise (U) approximates Q and produces a signal $\tilde{y}$. The g_s then transforms the latent space back into an image. The hyperprior analysis (h_a) and synthesis (h_s) provide side information to predict the probabilities for arithmetic coding given a context model (C_m).



posed fully learning-based coding models. Notably, many breakthroughs in learning-based compression are originated from image compression research, which was not included in our search string. In this section, we expand the review on learning-based coding models for image and video, including multi-layer compression approaches related to our proposal.

In image compression, the quantization step is essential to allow lossy compression, thus achieving high compression rates. However, the fact that such a step is not differentiable poses a great challenge to NN-based compression solutions. To mitigate this problem, Ballé, Laparra and Simoncelli (2016a) and Ballé, Laparra and Simoncelli (2016b) proposed a framework that approximates the quantization effects using uniform noise addition during training. After those works were published, other ones have shown remarkable results, further advancing the the state-of-the-art in handcrafted encoders. Ballé et al. (2018) proposed applying a hyperprior coding to predict the σ values of a zero-mean Gaussian distribution estimating the arithmetic coding probabilities (eq. 4.1 with K=1, w_1=1, and μ_1=0).

The works that followed have proposed further contributions to hyperprior coding. Minnen, Ballé and Toderici (2018) and Lee, Cho and Beack (2019) proposed methods to predict the mean used in the Gaussian distribution, relying on the context of previously decoded data (eq. 4.1 with K=1 and w_1=1). Cheng et al. (2020) proposed the model depicted in Fig. 27, generalizing the probability estimation considering the weighted sum of different Gaussian distributions as follows:

$$p(\hat{y}) \sim \sum_{k=1}^{K} w_k N(\mu_k, \sigma_k^2) \tag{4.1}$$

Several early applications of NN for video compression targeted specific handcrafted video encoder modules while still relying on the well-known quantization step from those encoders (Zhang; Kwong; Wang, 2020). Eventually, end-to-end solutions for video compression appeared, such as the DVC (Lu et al., 2019), a framework that maps handcrafted video encoder modules to equivalent learned models. Instead of using pretrained models for optical-flow prediction as proposed in DVC, the SSF model (Agustsson

et al., 2020) has motion estimation and compensation modules designed for video compression. The VCT model (Mentzer et al., 2022), in turn, explores the Transformer's temporal capabilities to simplify the learned video compression modules.

4.3.1 Multi-Layer Compression

Currently, there are three major handcrafted multi-Layer video compression standards: Scalable HEVC (SHVC) (Boyce et al., 2016), VVC (Bross et al., 2021a), and Low Complexity Enhancement Video Coding (LCEVC) (Battista et al., 2022). SHVC is a set of scalable tools based on the HEVC standard, requiring the reconstructed frames from the Base Layer (BL) and – if available – the motion information. The possibility of using only the reconstructed frames makes SHVC agnostic to the BL standard. In VVC, the successor of HEVC, the support for multi-layer compression was included as one of its features. LCEVC, in turn, is also agnostic to the BL standard, but there is a major difference between this and the aforementioned standards: LCEVC does not use any tools associated with existing coding standards. The LCEVC lossy compression was designed with low-complexity tools to encode the residues between the upsampled frames and the source frames.

End-to-end learned models are also being explored for multi-layer compression. Su et al. (2020) proposed a model to achieve quality scalability, where each layer has a model inspired by (Ballé et al., 2018), but allowing different layers to share parameters with recurrent structures. While the first layer compresses an image, the other layers compress residues to improve the previous layer quality, distributing the Rate-Distortion (RD) trade-offs throughout the multiple layers.

Mei et al. (2022) proposed a model for learned multi-layer image compression, exploring quality and spatial scalability. Their model has a learned module to process the decoded latent space features from a given layer, generating a predictor for the Enhancement Layer (EL) above. Those same features – upsampled when using spatial scalability – are concatenated to the results for all the EL above to improve the reconstruction. Both Su et al. (2020) and Mei et al. (2022) works have a fully learned end-to-end model, adapting the loss function for multi-layer compression. These works still use the RD optimization, but the rate and distortion are the sum of each layer's estimated rate and distortion, weighted by a given factor.

Multi-layer compression has also been proposed recently as a solution to produce standard-compliant bitstream using a handcrafted solution in the BL while isolating learned solutions to the EL. Lee et al. (2020) proposed a solution that employs a learned EL to transmit residual features used to enhance the BL reconstruction quality. The decoder synthesizes the enhanced images by combining the BL reconstruction and the EL compressed residues. Bonnineau et al. (2021) overall approach is similar, but their contribution is centered on spatial scalability. Benjak et al. (2023) address video compres-

Table 11 – Main Characteristics of Reviewed Works on Multi-layer compression and Ours.

Works	Single Layer/ BL Compression	EL Compression	Standard Compliance (BL, EL)	Spatial Scalability	Asymmetric Enc/Dec
Ballé, Laparra and Simoncelli (2016a) Ballé, Laparra and Simoncelli (2016b) Ballé et al. (2018) Minnen, Ballé and Toderici (2018) Lee, Cho and Beack (2019) Cheng et al. (2020) Lu et al. (2019) Agustsson et al. (2020) Mentzer et al. (2022)	Learned	N/A	No, N/A	N/A	No
Boyce et al. (2016) Battista et al. (2022)	Agnostic	Handcrafted	Yes, Yes	Yes	No
Su et al. (2020) Mei et al. (2022)	Learned	Learned	No, No	Yes	No
Lee et al. (2020)	Handcrafted	Learned	Yes, No	No	No
Bonnineau et al. (2021) Benjak et al. (2023)	Handcrafted	Learned	Yes, No	Yes	No
AMLC (Ours)	**Handcrafted**	**Learned**	**Yes, No**	**Yes**	**Yes**

sion, presenting a solution for both spatial and temporal scalability. Their work combines learned models for super-resolution, motion estimation, and residual compression. Those models are more complex than the ones in the aforementioned works, requiring several steps and presenting restrictions for training, such as a reduced training dataset for fine-tuning the whole model. In this work, we explore the multi-layer compression structure and the end-to-end enhancement-layer optimization to propose an asymmetric codec decoupling inter-layer dependencies at the encoding side. The following chapters present our proposed solution's architecture – named AMLC –, experiments, and analysis.

5 BASE-LAYER COMPRESSION AND SPATIAL-TEMPORAL UPSAMPLING

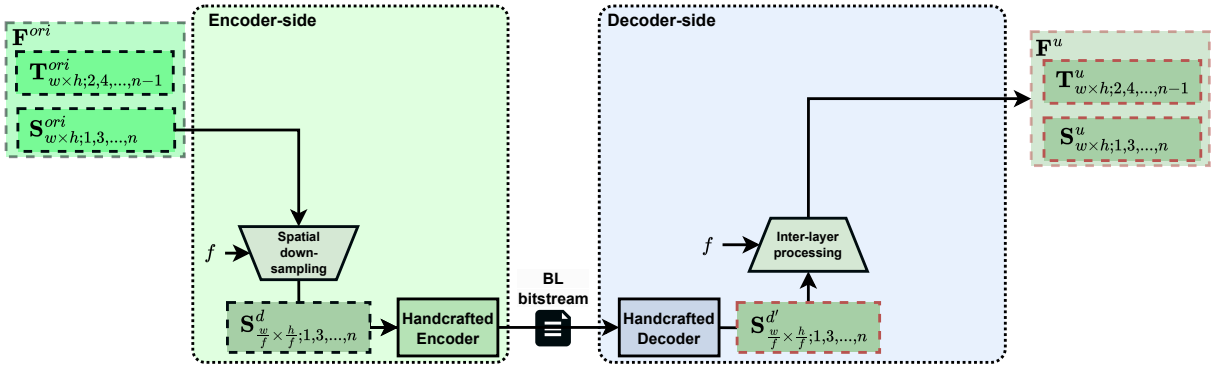
In this chapter, we address the specific goals related to exploring different spatial and temporal upsampling solutions in a lossy video compression scenario. The model depicted in Figure 28 is part of the AMLC model, excluding the learned EL. The symbols used in this chapter are described in Table 12.

Table 12 – Description of symbols used in Chapter 5

Symbol	Meaning
$\mathbf{F}^{\text{ori}}$	Original frame sequence
$\mathbf{T}^{\text{ori}}$	Original frames for temporal downscaling (even index)
$\mathbf{S}^{\text{ori}}$	Original frame for spatial downscaling (odd index)
f	Spatial downscaling factor (e.g., 2 or 4)
$\mathbf{S}^{\text{d}}$	Spatially downscaled frames obtained from $\mathbf{S}^{\text{ori}}$ using factor f
$\mathbf{S}^{\text{d'}}$	Spatially downscaled reconstructed frames
$\mathbf{T}^{\text{u}}$	Temporally interpolated frames
$\mathbf{S}^{\text{u}}$	Spatially upscaled frames
$\mathbf{F}^{\text{u}}$	Final upsampled video created by interleaving $\mathbf{S}^{\text{u}}$ and $\mathbf{T}^{\text{u}}$

First, the encoder-side performs temporal and spatial downsampling over the $\mathbf{F}^{\text{ori}}$. The temporal downsampling consists in removing the $\mathbf{T}^{\text{ori}}$ frames from the original video. The remaining frames – the $\mathbf{S}^{\text{ori}}$ sequence – are spatially downsampled in both dimensions by a factor of f, resulting in $\mathbf{S}^{\text{d}}$ frames. Then, the BL encoder, a video coding standard, compresses $\mathbf{S}^{\text{d}}$ frames into the BL bitstream. On the decoding side, the BL reconstructs the media at the low resolution. An Inter-layer processing module produces the upscaled $\mathbf{S}^{\text{u}}$ and interpolated $\mathbf{T}^{\text{u}}$ frames from the $\mathbf{S}^{\text{d}}$ sequence. Intuitively, it is possible to produce $\mathbf{F}^{\text{u}}$ by intercalating $\mathbf{S}^{\text{u}}$ and $\mathbf{T}^{\text{u}}$ frames.

Figure 28 – Implementation of the AMLC model without the EL.



Source: the author.

After removing data from $\mathbf{F}^{\text{ori}}$ at the encoding side, it is impossible to reconstruct the original video. Hence, $\mathbf{F}^{\text{u}}$ will contain distortions in comparison to the $\mathbf{F}^{\text{ori}}$ sequence.

We conducted experiments combining different spatial and temporal upsampling modules to measure the quality impact caused by that error.

Section 5.1 explains the adopted spatial upsampling solutions. Because we had developed and trained a frame interpolation model for temporal upsampling, Section 5.2 is dedicated to presenting the construction of such a model. We then present the results and discussions in Section 5.4.

5.1 SPATIAL UPSAMPLING

For spatial upsampling, we decided to use FSRCNN (Dong; Loy; Tang, 2016), a learned model proposed to achieve faster execution times without degrading too much the quality compared to the EDSR model proposed by Lim et al. (2017). In addition to the super-resolution using the FSRCNN, we also performed experiments using the bicubic method (Keys, 1981), a handcrafted algorithm for the task. We apply each upsampling solution considering scaling factors of 2× and 4×.

Bicubic interpolation operates in the spatial domain by estimating the intensity of each pixel as a weighted average of the nearest 4×4 neighboring pixels from the original image. The interpolation weights are derived from a cubic convolution function, ensuring that the resulting surface is smooth and continuous. This property produces more natural visual transitions compared to simpler methods, avoiding the block artifacts of nearest-neighbor interpolation and the excessive blurriness often introduced by bilinear interpolation.

Despite its popularity, bicubic interpolation has notable limitations when applied to image super-resolution. Because it relies solely on existing pixel values and deterministic mathematical rules, it cannot reconstruct fine textures or details that are absent in the low-resolution image. Consequently, images upscaled with bicubic interpolation often appear smooth but lack sharpness, particularly when the scaling factor is large. These limitations motivated the development of learning-based methods, such as SRCNN and FSRCNN, which can infer missing high-frequency details from training data rather than relying exclusively on interpolation.

The FSRCNN model is an efficient deep learning architecture designed for single-image super-resolution. Unlike SRCNN, which applies convolution on images that are first upscaled using bicubic interpolation, FSRCNN performs feature extraction directly on the original low-resolution input. This design choice allows FSRCNN to employ smaller filters in the initial feature extraction stage. After the feature extraction, a shrinking step is applied, using 1×1 convolution layers to reduce the depth of the feature space. The central part of the model is the non-linear mapping stage, which replaces the single wide mapping layer in SRCNN with multiple smaller 3×3 convolutional layers. This adjustment strikes a balance between accuracy and efficiency: deeper stacks of small filters achieve comparable representational power while requiring fewer parameters.

To restore the reduced feature dimensionality, FSRCNN adds an expanding layer, effectively the inverse of the shrinking step. Finally, the network produces the high-resolution output through a learned deconvolution layer. In contrast to the fixed bicubic interpolation used in SRCNN, FSRCNN learns upsampling filters directly from data, making them adaptive and task-specific.

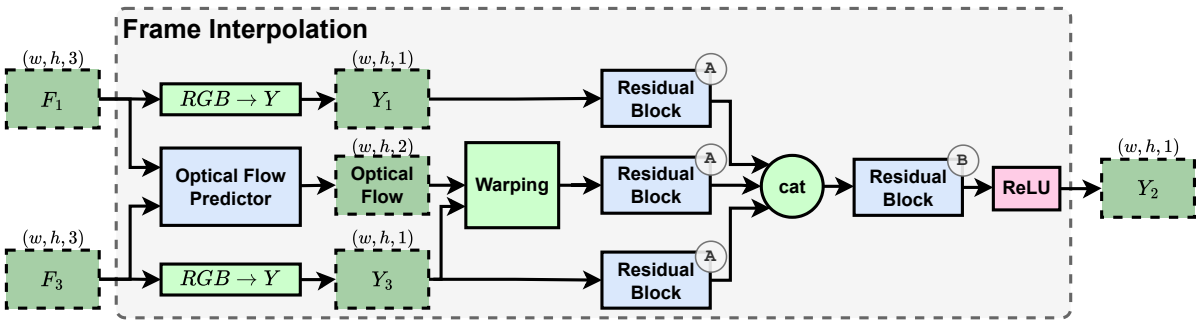
In summary, FSRCNN introduces three key changes compared to SRCNN: it avoids preprocessing by working directly on the low-resolution input, reduces and later restores feature dimensionality through shrinking and expanding layers, and replaces fixed interpolation with a learned deconvolution layer. Together, these modifications make FSRCNN faster and more effective for super-resolution tasks.

5.2 TEMPORAL UPSAMPLING

Frame Interpolation solutions rely on kernel estimation or optical flow estimation. Kernel-based solutions predict the kernel weights for each pixel in the frame. These kernels are processed as a convolution operation over two input frames to generate the middle one. However, because the kernel covers only a small image patch, it can capture movements only within that patch. In contrast, optical flow-based methods predict a motion vector for every pixel position without being constrained to a local patch. Since we propose removing frames from the original sequence, we artificially create a video sequence in which moving objects exhibit larger spatial displacement between frames. In this scenario, a kernel-based approach would require larger kernels to capture motion in our model than those needed for the original video. Therefore, we adopted an optical flow-based solution.

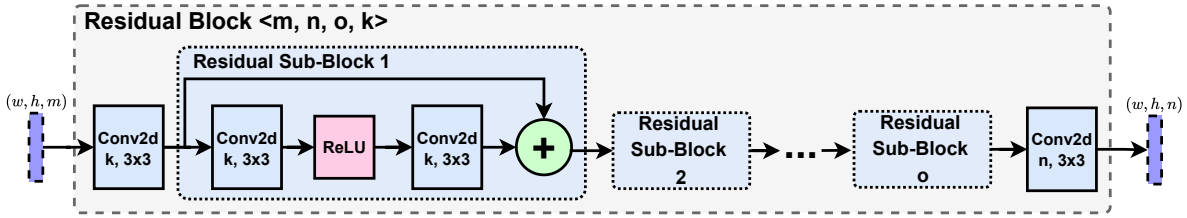
Our Frame Interpolation Module (Figure 29) has two main components: The Optical Flow Predictor and the Residual Blocks. A Residual Block (Figure 30) receives an input with depth m and produces an output with depth n. The first three Residual Blocks

Figure 29 – The proposed Frame Interpolation Module. The module takes two consecutive frames ($\mathbf{F}_1$ and $\mathbf{F}_3$) as input and predicts the luma channel of the intermediate frame ($\mathbf{Y}_2$). The Optical Flow Predictor, Residual Blocks, and ReLU layers contain trainable parameters. Residual Blocks marked as `A` are designed to preserve the input dimensions, while the Residual Block marked as `B` is configured to produce the output corresponding to the luma channel.



Source: the author.

Figure 30 – The Residual Block Module has 4 parameters: m and n are the input and output channel dimensions, respectively; o is the number of residual sub-blocks instances; and k is the number of 3×3 kernels in a convolutional layer.



Source: the author.

in the pipeline (A) have $m = n$, keeping the input dimensions in the output frame. The last Residual Block instance (B) receives a stack of features from the three first residual blocks and produces a single frame. In this case, the n parameter is equal to the output frame depth. Each Residual Block has o Residual Sub-blocks. The Residual Sub-blocks have convolutional layers with k kernels of 3×3 size, followed by a Batch Normalization Layer. Both o and k are Residual Sub-blocks parameters, and all instances within a Residual Block inherit the parameters of that Block. We configured all convolutional layers with a stride equal to 1, and enabled padding to keep the original video resolution.

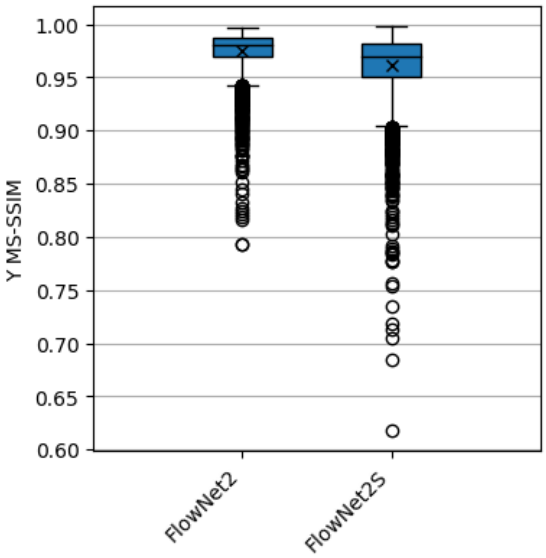
Among the available choices for the Optical Flow Predictor, we compared the Frame Interpolation Module results using the simplest (FlowNet2S) and the most complex (FlowNet2) implementations of FlowNet (Ilg et al., 2017). We instantiate the Frame Interpolation Module using Residual Blocks with $o = 4$ and $k = 32$. We trained the model for 50 epochs with Adam optimizer, setting the learning rate to $1e^{-5}$ and using a mini-batch of size 4. We use the MS-SSIM loss function, which better preserves the image structures than the loss functions L1- and L2-norm. We initialized both versions of the FlowNet using the checkpoints provided by Ilg et al. (2017), and we did not freeze their parameters for training.

We use the Vimeo90k dataset built by Xue et al. (2019), which has data specifically designed for the Frame Interpolation task. Vimeo90K contains 51,313 frame triplets for training and another 3,782 for validation. Both versions of FlowNet were initialized with the respective checkpoints provided by Ilg et al. (2017), and we did not freeze their parameter along with the training of the whole model.

Figure 31 shows the Y MS-SSIM distributions of each model using the *Vimeo90K* validation dataset. The averages of FlowNet2 and FlowNet2S are 0.97 and 0.96, and both have the lower quartile above 0.95. The difference may seem small, but Figure 31 clearly shows that the FlowNet2 distribution is much closer to 1.0, and even the outliers concentrate above 0.87.

At this point, FlowNet2 is the best option as the Optical Flow Predictor. However, the FlowNet2 quality gains come with a considerable complexity cost. Our Frame Interpolation Module using FlowNet2 has 162,819,830 parameters, while the same archi-

Figure 31 – Quality results distributions of Frame Interpolation Module when using FlowNet2 and FlowNet2S. The 'x' marks the average results.



Source: the author.

tecture using FlowNet2S has 38,977,502. Considering only the quality, FlowNet2S is not the best option. However, it can still provide good MS-SSIM results using only 23.9% of parameters relative to the implementation using FlowNet2. Because the FlowNet2S is much smaller, we can explore more complex structures in the overall Frame Interpolation Module architecture rather than allocating most of the available memory to the Optical Flow Predictor. Based on such observations, we proceeded with our research using FlowNet2S as the Optical Flow Predictor.

5.3 EXPERIMENTAL SETUP

In this section, we describe the experimental setup to validate the model depicted in Figure 29. **Baseline.** We adopted the x265 implementation of the HEVC video coding standard as the Baseline codec. We encode each video sequence using eight QPs: 12, 17, 22, 27, 32, 37, 42, and 47. We use QPs 42 and 47 only in our baseline executions, aiming to have RD data on the low bitrate range that our model achieves at QPs 32 and 37. We use the FFmpeg tool to encode/decode the sequences, keeping the default x265 configurations. In these experiments, we adopted the same encoder and configurations as our handcrafted BL codec, using QPs from 12 to 37.

Training dataset. We used the Vimeo90k – with samples at 448×256 resolution – for training and validation. When we apply data augmentation to the training, we add two copies of each sample containing a random combination of the following operations: Rotation, horizontal flip, vertical flip, and reversed sequence.

Model instances. Table 13 lists the baseline and all the experimental config-

Table 13 – Experiments labels and configurations.

Label	Temporal Up-sampling Module	Spatial Up-sampling Module	Spatial Up-sampling Factor
x265	-	-	-
T	FI-Mod	-	-
2S-Bicubic	-	Bicubic	2
2S-FSRCNN	-	FSRCNN	2
4S-Bicubic	-	Bicubic	4
4S-FSRCNN	-	FSRCNN	4
T+2S-Bic	FI-Mod	Bicubic	2
T+2S-FSRCNN	FI-Mod	FSRCNN	2
T+4S-Bicubic	FI-Mod	Bicubic	4
T+4S-FSRCNN	FI-Mod	FSRCNN	4

urations of the proposed model. We made experiments using two spatial upsampling methods: the Bicubic and Fast Super-Resolution Convolutional Neural Network (FSRCNN). We run both methods using the implementations available on *OpenCV* (Bradski, 2000), and we do not perform additional training on the FSRCNN. The temporal upsampling process is performed with our Frame Interpolation Module (FI-Mod), trained for 100 epochs with Adam Optimizer, learning rate at $1e^{-5}$, and mini-batch of size 4. We allocate the memory saved from using FlowNet2S instead of the FlowNet2 to Residual Blocks with more kernels. We kept the number of residual sub-blocks at $o = 4$ and set the number of kernels to $k = 64$. We used the Vimeo90k dataset and kept the Optical Flow Predictor parameter frozen until the 40th epoch. From then on, we trained with the augmented Vimeo90k dataset and enabled the training of all model parameters.

Test dataset. We restrict the test dataset to the first 100 frames of the 20 video sequences from the HEVC CTC (Bossen, 2013), listed in Table 14. We consider additional sequences from the VVC CTC and from Ultra Video Group (UVG) database (Mercat; Viitanen; Vanne, 2020), both having videos with 4K resolution.

We can use all video sequences in the baseline (x265) and temporal down-sampling (T) experiments. However, not all videos can be compressed using spatial down-sampling factors 2 and 4 (2S and 4S, respectively). Depending on the input video resolution, the down-scaling reduce the chroma channels to a size that the BL encoder cannot process. Table 14 also informs the videos for which we have results in 2S and 4S experiments. Notice that results for temporal and spatial down-sampling combinations (T+2S and T+4S) are also conditioned to the spatial down-sampling restrictions.

Evaluation metrics. We evaluate the compression results using only the bitstream size as the rate metric and the MS-SSIM of Y channel as the quality metric. We report the Rate-Distortion curves of each sample and then provide a summarized analysis separating the Rate and Distortion and comparing the results from all sequences by

Table 14 – CTC video sequences names, their respective resolution. The 'x's in the right-most columns indicate if the sequence can be encoded with the baseline conditions (x265), the temporal down-sampling (T), and the spatial down-sampling using factors 2 (2S) and 4 (4S).

Video	Resolution	x265	T	2S	4S
BasketballPass	416×240	x	x		
BlowingBubbles	416×240	x	x		
BQSquare	416×240	x	x		
RaceHorses	416×240	x	x		
BasketballDrill	832×480	x	x	x	
BasketballDrillText	832×480	x	x	x	
BQMall	832×480	x	x	x	
PartyScene	832×480	x	x	x	
RaceHorsesC	832×480	x	x	x	
ChinaSpeed	1024×768	x	x	x	x
FourPeople	1280×720	x	x	x	x
Johnny	1280×720	x	x	x	x
KristenAndSara	1280×720	x	x	x	x
SlideEditing	1280×720	x	x	x	x
SlideShow	1280×720	x	x	x	x
BasketballDrive	1920×1080	x	x	x	x
BQTerrace	1920×1080	x	x	x	x
Cactus	1920×1080	x	x	x	x
Kimono	1920×1080	x	x	x	x
ParkScene	1920×1080	x	x	x	x

quantization parameter.

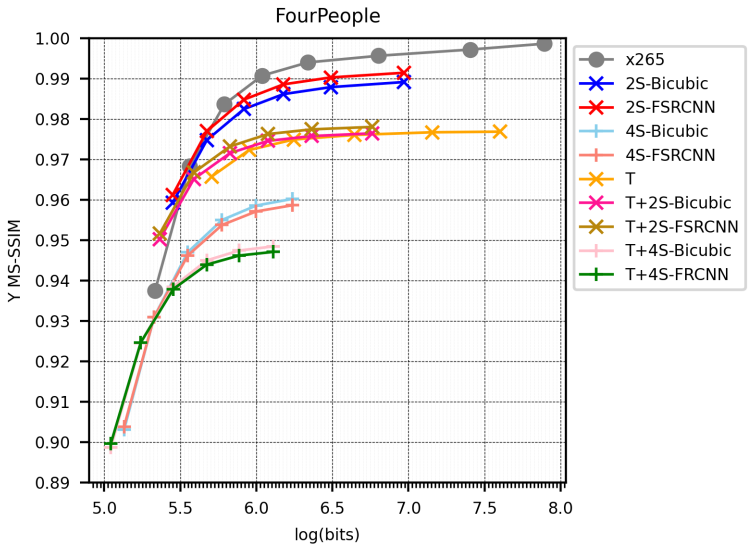
5.4 RESULTS AND DISCUSSIONS

Figure 32 shows results for the *FourPeople* video sequence. As expected, most of the points obtained with the implementation of our model without the EL are below the *x265* curve. However, it is possible to see that *2S* and *T+2S* configurations combined with QPs 32 and 37 have similar or better RD results relative to the *x265* curve. Such observation emphasizes the potential of the proposed model at lower bitrates.

Considering a fixed spatial upsampling factor, we can directly compare *Bicubic* and *FSRCNN* based on quality results only. For the upscaling factor of 2, the *FSRCNN* has slightly better quality results than the *Bicubic*. This scenario changes when using a factor of 4, and the *Bicubic* overcomes the *FSRCNN*. Despite the differences, both methods have similar quality distributions, as better observed in Figure 33.

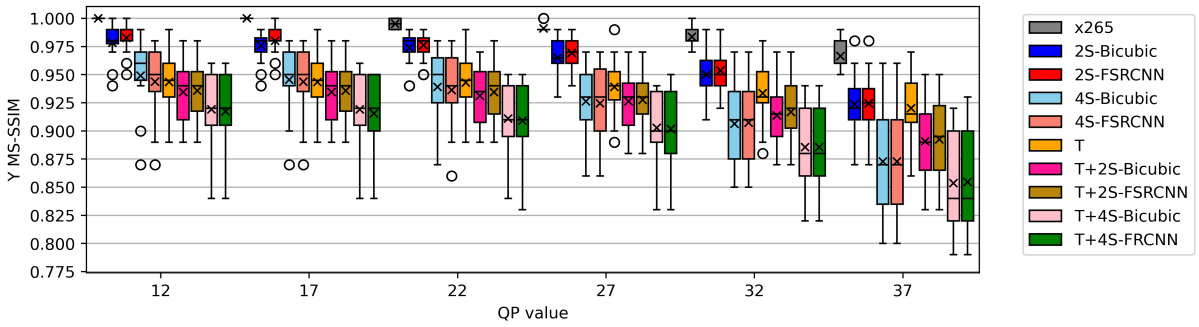
When we analyze only the temporal dimension, Figure 33 shows us that the quality distributions of *T* are lower than the *2S* experiment results. Furthermore, Figure 34 shows us that the bitrate reduction relative to the *x265* is under 50%, while all the other configurations are above the 60% line. At this point, we could consider removing the

Figure 32 – RD curves of *FourPeople* video sequence.



Source: the author.

Figure 33 – Y MS-SSIM distributions of each experiment and QP.

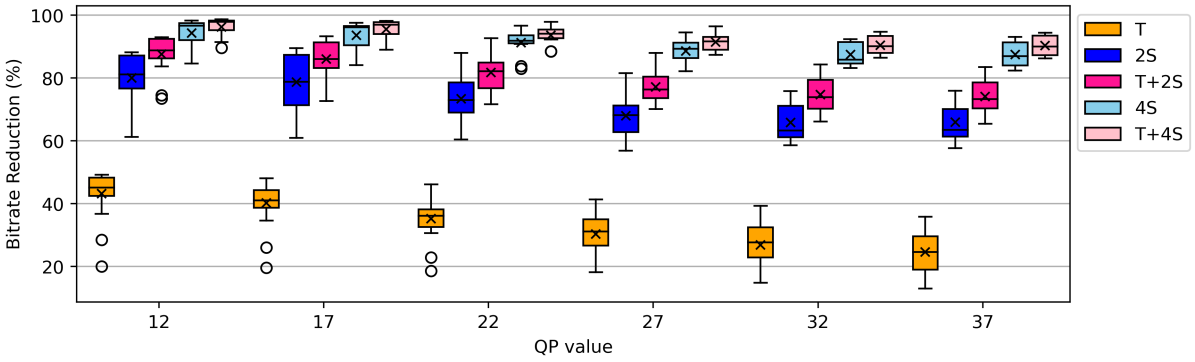


Source: the author.

temporal dimension from our model. Exploring the dynamic aspects of a video sequence to reconstruct an entire frame is more challenging than interpolating pixels within a single frame to increase the spatial dimension. However, by combining temporal and spatial dimensions in our model, we can improve the bitrate reduction relative to exploring only the spatial dimension.

One notable benefit of combining spatial and temporal upsampling is the improved RD results compared to exploring only the temporal dimension. As illustrated in Figure 33, the quality distribution of the *T+2S* results overlaps with that of the *T* experiment for QPs 12 through 32. When considering both quality and bitrate, as shown in Figure 32, *T+2S* consistently demonstrates equal or superior performance compared to the *T* experiment. We identified this trend across 12 out of the 16 video sequences where a spatial scaling factor of 2 is applicable, and *T+2S* results remain comparable to *T* for the remaining sequences. This behavior can be explained by the fact that reducing the spatial resolution of the video brings moving objects spatially closer across consecutive

Figure 34 – Bitrate reduction relative to the *x265* experiment.



Source: the author.

frames, allowing the learned Frame Interpolation Module to more effectively capture their displacement.

In summary, the experimental results demonstrate that the proposed model achieves competitive or superior rate–distortion performance relative to x265 baseline, particularly at lower bitrates with *2S* and *T+2S* configurations. Spatial upsampling consistently provides more stable quality improvements than temporal interpolation, yet the joint exploration of spatial and temporal dimensions yields the most significant bitrate reductions while maintaining acceptable perceptual quality (Y-MS-SSIM above 0.8 even under high QP values). These results confirm the effectiveness of integrating temporal and spatial components within the enhancement layer and suggest that further optimization of the upsampling modules could strengthen coding efficiency across a wider range of sequences and operating points. Although the combination of temporal and spatial scaling shows potential for further improvements, this work continues with spatial scaling using a factor of 2. This configuration not only achieves the closest rate–distortion performance to the baseline but also constitutes a prerequisite for any temporal solution. The complete implementation of this configuration, together with detailed comparisons against existing multi-layer codecs, are presented in the next chapter.

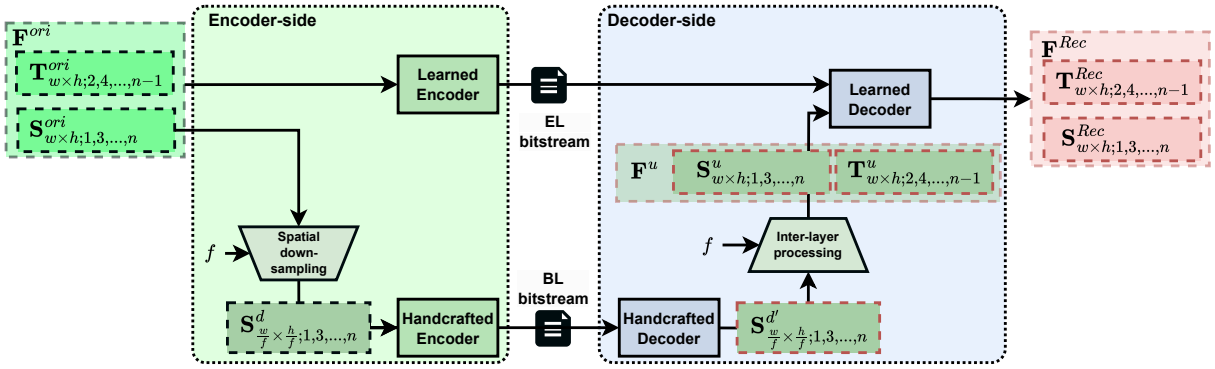
6 ASYMMETRIC MULTI-LAYER CODEC

In this chapter, we explore the full implementation of AMLC, depicted in Figure 35. The symbols used in this chapter are described in Table 15. Compared to previous chapter, we add the EL to compress the media in its original resolution, transmitting additional information. The EL decoder combines data from that bitstream and the $\mathbf{F}^{\mathrm{u}}$ sequence to improve the $\mathbf{F}^{\mathrm{rec}}$ reconstruction quality. Adding more data to the final bitstream naturally increases the bitrate. However, it also allows us to better handle the rate-distortion trade-offs.

Table 15 – Description of symbols used in Chapter 6

Symbol	Meaning
$\mathbf{F}^{\mathrm{ori}}$	Original frame sequence
$\mathbf{T}^{\mathrm{ori}}$	Original frames for temporal downscaling (even index)
$\mathbf{S}^{\mathrm{ori}}$	Original frame for spatial downscaling (odd index)
f	Spatial downscaling factor (e.g., 2 or 4)
$\mathbf{S}^{\mathrm{d}}$	Spatially downscaled frames obtained from $\mathbf{S}^{\mathrm{ori}}$ using factor f
$\mathbf{S}^{\mathrm{d'}}$	Spatially downscaled reconstructed frames
$\mathbf{T}^{\mathrm{u}}$	Temporally interpolated frames
$\mathbf{S}^{\mathrm{u}}$	Spatially upscaled frames
$\mathbf{F}^{\mathrm{u}}$	Final upsampled video created by interleaving $\mathbf{S}^{\mathrm{u}}$ and $\mathbf{T}^{\mathrm{u}}$
$\mathbf{S}^{\mathrm{rec}}$	Spatially reconstructed frames
$\mathbf{T}^{\mathrm{rec}}$	Temporally reconstructed frames
$\mathbf{F}^{\mathrm{rec}}$	Final reconstructed video created by interleaving $\mathbf{S}^{\mathrm{rec}}$ and $\mathbf{T}^{\mathrm{rec}}$

Figure 35 – Proposed framework (AMLC).



Source: the author.

Although we combine different temporal and spatial upsampling modules in Chapter 5, the effectiveness of temporal solutions depends on the coding losses introduced by the spatial ones. This dependency justifies the necessity of first implementing an intra-frame codec. Therefore, this chapter only explores spatial upsampling. Additionally, because we are compressing single frames, this chapter shifts the scope to image compression solutions. Because we operate with video content datasets and handcrafted

video codecs as AMLC BL, the proposed method applies to both image and video representations.

6.1 AMLC MODEL

Figure 35 shows the proposed model called AMLC. Similarly to other multi-layer models, AMLC has a BL, an EL, and an interlayer processing block. The BL uses a handcrafted codec, and we chose the HEVC reference software in our case study. Our decision to use HEVC instead of the latest standard (VVC (Bross et al., 2021b)) allowed us to conduct a greater number of experiments due to the difference in execution time between the two reference softwares. However, the AMLC model abstracts the BL implementation, allowing for future experiments to be conducted with different handcrafted implementations under the same method described in this work.

The inter-layer processing depends on the scalability type adopted in the system. In this work, we consider only spatial scalability, which applies to both image and video compression, implementing the inter-layer processing with the bicubic upsampling method. Despite advances in learned Super-Resolution (SR) (Dong et al., 2014; Liang et al., 2022; Chen et al., 2023), we chose to employ a simpler solution for inter-layer processing. This approach avoids using an additional model that could require re-training and would increase the overall complexity, such as observed in Benjak et al. (2023).

Figure 36 provides more details on the EL decoding implementation. The learned encoder and decoder correspond to the implementation presented by (Cheng et al., 2020) (presented in Section 4.3) that uses *attention* blocks, the state-of-the-art solution for image compression. The concatenation of the Interlayer Prediction (ILP) and the learned decoder output feeds the Merge block. Our Merge block implementation – depicted in Figure 37 – is similar to the U-net (Ronneberger; Fischer; Brox, 2015). Although the U-net was proposed for an image segmentation problem, a similar module, commonly called feature extraction or feature merging, appears in other image processing tasks research such as Video Frame Interpolation (VFI) (Niklaus; Liu, 2018; Niklaus; Liu, 2020). To avoid checkerboard visual artifacts in the frames, we implement Down and Up blocks (figure 38) using average pooling and bilinear upsampling layers, respectively.

Figure 36 – EL decoding solution for the AMLC model depicted in figure 35.

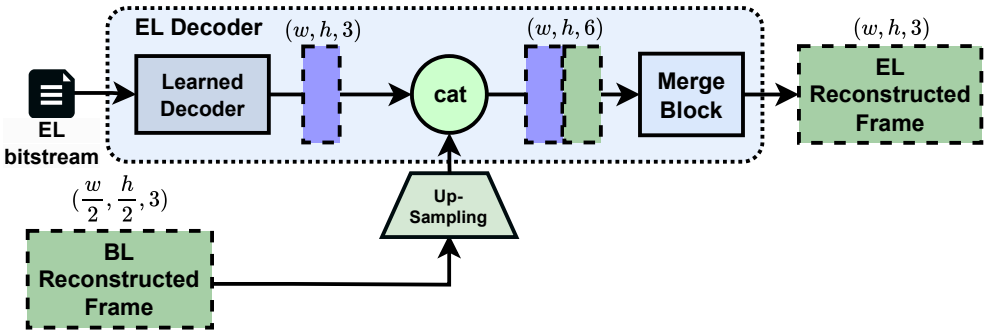


Figure 37 – Merge block. On the left side, the Down blocks decrease the width (w) and height (h) dimensions and increase the number of feature maps by a factor of 2. The Up blocks on the right side perform the opposite process. The Conv2d parameters are the kernel size k, stride s, and $pad = 1$ for padding. The number of feature maps depends on the parameter n.

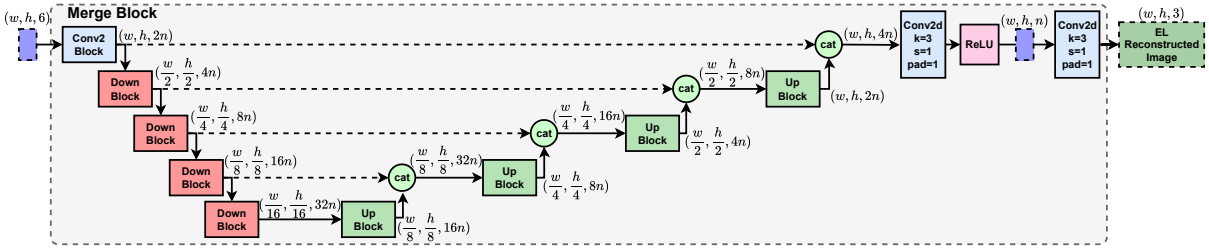
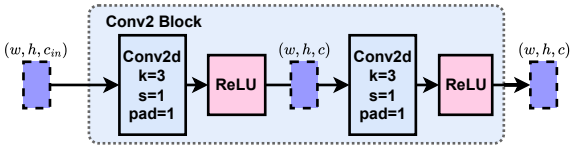
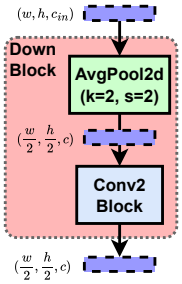


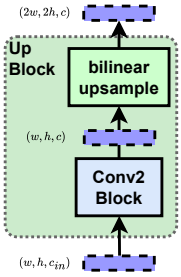
Figure 38 – Merge block component details. The number of feature maps C_{in} and c can be inferred from the dimensions presented in figure 37.



(a) Conv2 Block



(b) Down Block



(c) Up Block

Source: the author.

6.1.1 Two Stage Training and Loss Functions

The AMLC EL has two different learned components: the learned compressor and the Merge block. While there are pre-trained models available for the learned compressor, we had to train the Merge block from scratch. In order to reduce the training effort, we relied on the transfer learning technique, splitting the training into two stages and adapting the loss functions accordingly.

In the first stage, we transferred and frozen the weights of a learned image compressor into AMLC EL compression block and updated only the Merge block weights. We defined the training loss as follows:

$$Loss = D(Ori, Rec) \tag{6.1}$$

, where D can be either calculated for MSE or MS-SSIM with the respective functions $255^2 \times \text{MSE}(.)$ – assuming 8-bit samples – or $1 - \text{MS-SSIM}(.)$(Bégaint et al., 2020).

Because the compressor is frozen, the compression rate is constant and has no role in this optimization.

The second stage consists in fine-tuning the whole model, which now includes the EL compressor training. In this stage, the EL compressor is no longer responsible for coding frames, but learned features to improve the final frame reconstruction. Because training affects the compression, our loss function has to consider the RD trade-off such as follows:

$$Loss = \lambda D(Ori, Rec) + R_{\text{Total}} \tag{6.2}$$

, where λ is the Lagrangian multiplier. Considering the multi-layer compression, we formulate the bitrate component (R_{Total}) as the sum of both the R_{EL} and R_{BL}. R_{EL} is the bitrate from the EL compressor, which we estimate along the training following the formulations from Cheng et al. (2020) and Bégaint et al. (2020). We obtain the R_{BL} bitrate directly from the BL compressed file, which is constant for any given sample.

Related works set lambda (λ) values to achieve different RD trade-off levels on each training of their learned compressor. Although we use a learned compressor implementation from the literature, we cannot directly use the λ values reported by the authors (Bégaint et al., 2020; Cheng et al., 2020) because we have changed the Loss equation. We select new λ values verifying the following condition:

$$|D_{S1} - D_{S2}| < threshold \tag{6.3}$$

where D_{S1} and D_{S2} denote the evaluation distortion at the end of the first training stage and the beginning of the second, respectively. The *threshold* is empirically set to 1 when training with the MSE loss.

6.2 EXPERIMENTAL SETUP

In this section, we detail the experimental configurations, which include baseline implementations, model parameters, datasets, and evaluation metrics.

6.2.1 Experimental Configuration

Infrastructure. We performed training, inference, and runtime results gathering in a machine equipped with 16GB of RAM, a Ryzen 7 5700X CPU, and a single RTX3060 GPU with 12GB of VRAM.

Baselines. We use the SHVC reference software (SHM12.4[1]) in the All-Intra configuration as a handcrafted multi-layer baseline codec. For the fully learned multi-layer codec (FL-codec), we use the Cheng2020 (Cheng et al., 2020) in simulcast mode, following the method described by Mei et al. (2022). To generate the RD curves, we set

[1] Available at: https://hevc.hhi.fraunhofer.de/svn/svn_SHVCSoftware/

the BL and EL operating at the same quality levels in both cases, using the QPs 27, 32, 37, and 42 for SHVC and levels 1, 3, 4, and 6 for FL-codec.

Dataset. We used the Vimeo90k (Xue et al., 2019) for training and evaluation, processing the data as follows. From each frame triplet from the dataset, we select the second frame only. By using the FFmpeg (FFmpeg, 2020), we converted each sample from RGB to YUV420. We converted them back to RGB by copying the chroma samples – without any interpolation procedure – and then using the color conversions adopted in the JPEG AI Quality Assessment Framework (JPEG, 2023). Notice that the final RGB image reflects the degradation from the chroma sub-sampling necessary for YUV420 conversion. However, the adopted conversion guarantees two properties: 1) the following YUV420 and RGB conversion will not accumulate any error; 2) because the learned compression model training uses RGB data, we now ensure that AMLC cannot take advantage of any color information from RGB that was not available in the YUV420 representation. The described dataset manipulation is relevant for our work because development and assessment handcrafted compression methods use YUV420 data (Bossen, 2013) – which is the case of AMLC BL encoder and the baseline reference.

In addition to the Vimeo90k, we assess the performance of the AMLC model using the validation split from CLIC-Professional and CLIC-Mobile datasets (Toderici et al., 2020). CLIC-Professional has professionally captured images, ranging from 512×384 up to 2048×1366 resolution. CLIC-Mobile has smartphone images captured by regular users, ranging from 996×756 up to 1520×2048 resolution. We apply the center crop to CLIC dataset samples, forcing each of them to have both width and height multiples of 64, which ensures all codecs can operate without requiring padding.

Base Layer. The converted datasets have the original frames to train the EL compressor, but we still need to pair them with the BL reconstruction for model training. We downsampled the YUV420 frames with FFmpeg using the bicubic method, scaling down each dimension to 0.5× and then compressed each frame using the HEVC Test Model HM-16.4[2] on All-Intra configuration setting the BL QPs with the values of 22, 27, 32, and 37 (Bossen, 2013). We trained AMLC considering the results from each QP as an individual dataset to observe how the EL handles the BL encoded at different RD trade-offs.

Model Instances. We pre-loaded the learned compressor weights using the CompressAI checkpoints (Bégaint et al., 2020). Although Cheng et al. (2020) model has six compression levels available, we limited our research to four, selecting two with the highest compression results (levels 1 and 3), and two with the highest quality results (levels 4 and 6). Level 1 and 3 models set the kernel number to 128, whereas levels 4 and 6 set it to 192 (Bégaint et al., 2020; Cheng et al., 2020). We set $N = 8$ in the Merge block for all model instances. Combining the BL QPs and EL compression levels results

[2] Available at: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/

in 16 model instances for training.

Training Parameters. We used center crop patches of 256×256 resolution, batch size 12, and the Adam optimizer. We set the learning rate of the first training stage to 1e-4, reducing it to 1e-5 in the second stage. We made experiments using the MSE as the distortion metric in eq. 6.1 and 6.2. Table 16 presents the λ values empirically found for each model configuration.

Table 16 – λ Training Parameters for Each Combination of BL QP and EL Level.

BL QP	EL level			
	1	3	4	6
22	0.0054	0.0163	0.0248	0.0508
27	0.0051	0.0126	0.0185	0.0435
32	0.0037	0.0088	0.0153	0.0410
37	0.0022	0.0077	0.0141	0.0410

6.2.2 Evaluation metrics

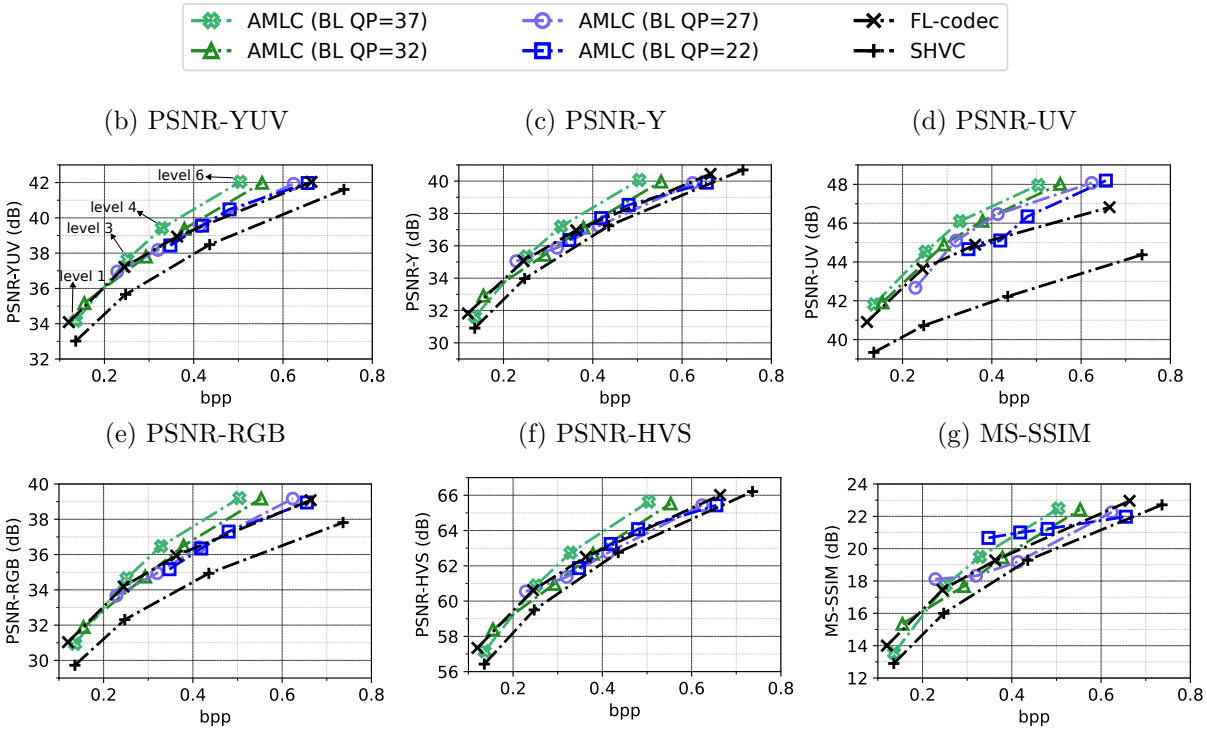
Bitrate. We can obtain the bitrate by adding the total file size from each layer's bitstream. We also use a bitrate metric that is relative to image resolution, called bits per pixel (bpp).

Quality. In this work we employed six different objective quality metrics to assess the AMLC results: the PSNR-YUV with the luma channel weighted by 6 and the chroma channels each weighted at 1 and its components luma (Y) and chroma (U and V averaged); the PSNR-RGB, which is commonly used to assess learned compression models (Bégaint et al., 2020); the PSNR-HVS-M (Ponomarenko et al., 2007), which is still based on PSNR but takes the HVS characteristics into consideration (frequency and contrast); we use the MS-SSIM metric (Wang; Simoncelli; Bovik, 2003), which also tries to improve the correlation with the HVS by considering the image structural information. We compute all the metrics using the JPEG AI Quality Assessment Framework (JPEG, 2023).

Coding Efficiency. After measuring the rate and quality, we can compare the coding efficiency of different encoders using the BD with polynomial fitting (Bjøntegaard, 2001; Herglotz et al., 2024). For BD computation purposes, we generate the convex hull RD curve from the different coding configurations of AMLC. Although the original BD formulation is based on the PSNR metric, it can be applied to MS-SSIM after converting it as follows (Herglotz et al., 2024):

$$\text{MS-SSIM}_{\text{dB}} = -10 \log_{10} (1 - \text{MS-SSIM}) \tag{6.4}$$

Figure 39 – Average RD results of the SHVC, FL-codec and the proposed AMLC model on the Vimeo90k dataset for each quality metric. The QP is used to specify an anchor BL coding parameter for AMLC.



6.3 EXPERIMENTAL RESULTS

In this section, we evaluate the experimental results by analyzing RD curves, coding efficiency results, complexity, and a subjective evaluation.

6.3.1 RD Evaluation

Figure 39 shows the average RD results on Vimeo90k. The AMLC curves are generally above SHVC curves for all quality metrics: the BL QP 37 curve shows the best results, while the BL QP 22 is the closest one to the SHVC curve. At the highest EL quality level (level 6), AMLC model with BL QP 37 has higher compression rates without losing quality compared to lower QPs. Although counter-intuitive, this happens because the smaller the BL bitstream size, the larger the space available for EL optimizations, which is the only result evaluated by the quality metric. As the EL rate decreases, lower BL QPs sustain better quality results than the others. The BL QP 22 curve, for instance, does not present the best RD results, but it has the best quality results at lower EL levels, as clearly – but not exclusively – observed in figure 39 (f).

When compared to the FL-codec, the AMLC achieves higher compression rates while maintaining similar quality results when combining BL QPs 32 and 37, and EL

Table 17 – Average BD results comparing the AMLC against the SHVC.

Quality Metrics	Vimeo90k		CLIC-Mobile		CLIC-Professional	
	BD-Quality (dB)	BD-Rate (%)	BD-Quality (dB)	BD-Rate (%)	BD-Quality (dB)	BD-Rate (%)
PSNR-YUV	1.69	-27.86	0.75	-18.06	0.85	-20.10
PSNR-Y	1.14	-17.05	0.27	-6.47	0.41	-8.06
PSNR-UV	3.66	-62.41	2.40	-56.71	2.55	-62.42
PSNR-RGB	1.99	-34.19	0.89	-20.60	1.04	-25.67
PSNR-HVS	1.14	-16.96	0.39	-7.76	0.41	-8.20
$\text{MS-SSIM}_{\text{dB}}$	1.98	-26.57	1.26	-22.56	1.35	-23.91

levels 4 and 6. However, to fairly compare both implementations, it is first necessary to assess their BL implementation performance. In the following subsection, we present the coding efficiency comparisons to objectively demonstrate the AMLC advantages.

6.3.2 Coding Efficiency Comparisons

Table 17 reports the AMLC coding efficiency results, comparing it to the SHVC. As one could expect from the Vimeo90k RD-curves, AMLC solution has significant coding efficiency gains compared to SHVC, demonstrating the benefits of using an EL learned solution. It is noticeable that the hightest gains in all datasets are for the UV components. Nevertheless, the gains over the SHVC are not only better on PSNR-UV, PSNR-RGB, and MS-SSIM, but also on PSNR-Y and PSNR-HVS.

Mei et al. (2022)'s ablation study has shown that training and evaluating a learned model with YUV420 media is not as promising as conducting the research in RGB domain: their proposal has slightly worse results than the SHVC codec, particularly for PSNR-Y. In this work, the YUV420 domain is relevant because handcrafted video codecs typically operate and are optimized on that domain. Although we do not train with YUV420 as suggested by Mei et al. (2022), we guarantee a lossless YUV420 to RGB conversion, and vice versa. Therefore, we train on RGB without using data removed with the YUV420 chroma sub-sampling. AMLC coding efficiency gains over SHVC for both chroma (UV) and luma (Y) indicate that our training method is a viable alternative to directly training on YUV420.

HEVC and SHVC introduced tools to improve the coding efficiency of samples with higher resolution than the ones of Vimeo90k samples (Sullivan et al., 2012). AMLC model, in turn, was trained only with the low resolution frames from Vimeo90k dataset. Therefore, SHVC has higher odds to achieve better results in the CLIC datasets. Relative to the results on the Vimeo90k dataset, AMLC has reduced coding efficiency gains compared to SHVC on CLIC datasets. Nevertheless, the AMLC still performs better in all these cases, with an improvement of -6.47% on BD-rate in the worst case.

Table 18 – Average Handcrafted BL Coding Efficiency Loss Compared to the Learned One.

Metric	PSNR-YUV	PSNR-RGB	PSNR-HVS
BD-Rate (%)	23.42	11.48	33.68
BD-Quality (dB)	-1.13	-0.64	-1.5

Table 19 – Average BD Results Comparing the AMLC Against the FL-codec.

Quality Metrics	Vimeo90k		CLIC-Mobile		CLIC-Professional	
	BD-Quality (dB)	BD-Rate (%)	BD-Quality (dB)	BD-Rate (%)	BD-Quality (dB)	BD-Rate (%)
PSNR-YUV	0.16	-3.61	0.14	-3.61	0.12	-2.90
PSNR-Y	0.07	-0.80	0.08	-1.40	0.09	-1.15
PSNR-UV	0.73	-13.04	0.56	-14.60	0.51	-12.76
PSNR-RGB	0.09	-2.73	0.01	0.31	-0.03	0.57
PSNR-HVS	0.04	-0.36	0.14	-2.47	0.08	-0.96
MS-SSIM	0.58	-8.08	0.68	-12.95	0.62	-11.27

Before comparing the AMLC results with the FL-codec, it is necessary to first understand the coding efficiency results of their respective BL codecs. Table 18 summarizes the BD results comparing the handcrafted and learned BLs for the PSNR-based metrics. The handcrafted BL is less efficient than the learned BL by 33.68% and -1.5dB in the worst case. However, from Table 19, we observe that AMLC delivers similar or slightly better coding efficiency results than FL-codec across different quality metrics and dataset combinations. Taking the PSNR-RGB evaluation on CLIC-Professional, for instance, AMLC loses by 0.6% and 0.03dB compared to the FL-codec. However, this is still a remarkable achievement when considering the handcrafted BLs coding efficiency disadvantage.

Finally, we made an equivalent implementation of the AMLC model, but adding residual connections at both encoding- and decoding-side, similarly to a usual multi-layer. Table 20 summarizes the coding efficiency results comparing the AMLC model to the modified one. Both present similar coding efficiency in terms of quality for all evaluated metrics. Considering the coding efficiency comparisons in terms of bitrate, the AMLC is slightly better for PSNR-YUV and MS-SSIM, while losing for PSNR-RGB and PSNR-HVS. Nevertheless, the close results between those models are evidence that the encoding-side inter-layer dependency elimination, adopted in AMLC, does not affect negatively the overall coding efficiency performance results compared to a usual multi-layer compression implementation.

One can optimize learned models for other metrics through loss function tuning and training it to maximize the MS-SSIM quality instead of the MSE, for instance. By training AMLC with MS-SSIM, we might improve MS-SSIM results. However, the coding efficiency gains on that metric, even considering the MSE-based training, is evidence of

Table 20 – Average AMLC Coding Efficiency Results on Vimeo90k Compared to an Equivalent Implementation Using Residual Connections at the Encoding-side.

Metric	PSNR-YUV	PSNR-RGB	PSNR-HVS	MS-SSIM
BD-Rate (%)	-2.12	2.17	4.0	-0.24
BD-Quality (dB)	0.14	-0.08	-0.2	0.02

Table 21 – Average Encoding and Decoding Times (seconds) on Vimeo90k.

model	Lowest Quality		Highest Quality	
	Encode	Decode	Encode	Decode
Handcrafted-BL	0.1147	0.0022	0.1957	0.0035
Learned-BL	0.2051	0.6700	0.4079	0.9002
Learned-EL	0.6818	2.3430	1.4164	3.1704
Merge Block	N/A	0.0217	N/A	0.0217
SHVC	0.5993	0.0115	0.9691	0.0168
FL-codec	0.8869	3.0130	1.8243	4.0706
AMLC	0.6818	2.3647	1.4164	3.1921

AMLC's robustness. Therefore, we did not conduct the MS-SSIM-based training.

6.3.3 Complexity

Table 21 reports the encoding and decoding time for Vimeo90k, in seconds, of different codecs and components. The Handcrafted-BL is the BL codec used in both SHVC and AMLC. The FL-Codec uses both Learned-BL and Learned-EL, which are the Cheng2020 model operating at different layers. Meanwhile, the AMLC EL combines both Learned-EL and the Merge Block. Compared to the learned model, the handcrafted BL faster execution times compensate for its disadvantage in coding efficiency loss: the handcrafted codec is at least 2× and 257× faster to encode and decode, respectively. These results, aligned with existing well-established standards and fast implementations, motivated us to choose a handcrafted codec as the BL.

Table 22 has a similar report as Table 21 but considers the compression of frames at 1080p resolution. When compressing at a higher resolution, the handcrafted BL is 2.34× faster than the learned BL to encode. However, notice that the handcrafted codec cannot output more than one image every second. This is because we use the HEVC reference software to generate the results, which was developed as the golden model for coding efficiency but is not the fastest implementation for that standard. However, even the reference software focuses on a low-complexity decoding time. In such a case, the handcrafted decoder is 357.64× faster than the learned decoder. These results, aligned with existing well-established standards and fast implementations, motivated us to choose a handcrafted codec as the BL.

Considering the whole multi-layer coding implementation, the handcrafted codec

Table 22 – Average Encoding and Decoding Times on 1920×1080 resolution frames in Seconds.

model	Lowest Quality		Highest Quality	
	Encode	Decode	Encode	Decode
Handcrafted-BL	1.4851	0.0177	2.5186	0.0356
Learned-BL	2.9180	9.5179	5.8763	12.7322
Learned-EL	11.5741	38.0184	23.6484	52.2156
Merge Block	N/A	0.9395	N/A	0.9417
SHVC	8.7611	0.1160	15.5980	0.1912
FL-codec	14.5112	47.5553	29.5438	64.9669
AMLC	11.5741	38.0551	23.6484	52.2703

also overcomes – as expected – both the AMLC and the FL-codec in terms of complexity. Compared to the AMLC, the SHVC is, in the worst case, 1.13× and 205.6× faster to encode and decode on Vimeo90k, respectively. Once again, that speed-up ratio does not change significantly on the encoding side at higher resolutions, but the SHVC is at least 328× faster to decode than the learned solutions.

The asymmetric proposed approach and the use of handcrafted BL result in an advantage to AMLC over the FL-codec: AMLC is at least 1.28× and 1.27× faster to encode and decode, respectively. Although the asymmetric approach helps reduce execution times, the learned compressor – which is the same implementation for both AMLC and FL-codec – is still the most time-consuming module. Meanwhile, the Merge Block has a negligible impact on execution time: it represents less than 1% of the total decoding time. Therefore, the EL learned compressor is a candidate for future works focused on complexity optimization.

6.3.4 Visual Comparisons

Figure 40 presents random samples from Vimeo90k validation dataset. At the highest quality configuration, the AMLC produces similar visual quality results compared to the other methods, but with a higher compression rate. These samples and figure 41 also demonstrate that the AMLC is not producing visual artifacts such as checkboard patterns and color shifting.

Figure 40 – Random samples from Vimeo90k with each codec configure to produce the highest quality.

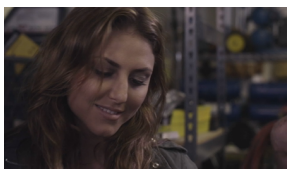
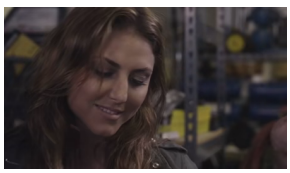
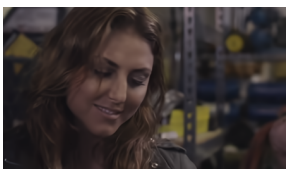
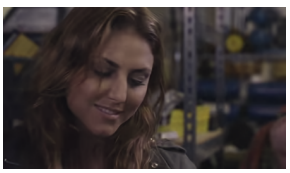
Original	SHVC	FL-Codec	AMLC
			
	bpp: .5572; psnr-yuv: 46.46; ms-ssim: .9976	bpp: .3018; psnr-yuv: 44.21; ms-ssim: .9958	bpp: .2331; psnr-yuv: 44.18; ms-ssim: .9952
			
	bpp: .5017; psnr-yuv: 47.67; ms-ssim: .9982	bpp: .2918; psnr-yuv: 45.21; ms-ssim: .9967	bpp: .2175; psnr-yuv: 45.42; ms-ssim: .9965
			
	bpp: .2687; psnr-yuv: 49.17; ms-ssim: .9969	bpp: .1445; psnr-yuv: 47.17; ms-ssim: .9950	bpp: .1077; psnr-yuv: 47.17; ms-ssim: .9946
			
	bpp: .5984; psnr-yuv: 46.04; ms-ssim: .9968	bpp: .3345; psnr-yuv: 44.05; ms-ssim: .9935	bpp: .2476; psnr-yuv: 43.94; ms-ssim: .9926

Figure 41 – Patches from CLIC-Professional images, demonstrating that AMLC is not producing color shifting or checkboard patterns artifacts.

7 CONCLUSIONS

In this work, we presented a taxonomy for NN solutions for video compression. At the time, few works had proposed fully-learned compression models. We then expanded the discussion on related works to include learned image compression, which was more advanced in that area than video compression. We observed a few works that proposed multi-layer learned compression solutions.

Multi-layer compression is employed to achieve scalability. The media is first compressed at a lower resolution at the BL, while the EL transmits additional bitstreams to reconstruct the media at higher resolutions. In this work, we explored multi-layer compression with a different purpose: to combine handcrafted and learned compression solutions, using the former at the BL and the latter at the EL.

Our first experiments – without the EL compression – allowed us to conclude that exploring both temporal and spatial upsampling for reconstruction produced similar quality results to employing temporal upsampling alone but with a higher compression rate. Although the spatial-temporal upsampling combination has advantages, we observed that the spatial upsampling by a factor of 2 produced the closest results to the single-layer handcrafted codec adopted as the baseline. Therefore, we explored the full implementation relying only on spatial upsampling.

When proposing the EL compression, we took advantage of the end-to-end compression to simplify the encoding process by removing inter-layer dependencies. The proposed model, named AMLC, successfully reduces complexity compared to a fully learned multi-layer solution, and it is faster for both the coding and decoding processes. Moreover, the handcrafted BL solution is at least 2× faster for encoding and up to 257× faster for decoding. It also leverages well-established platforms and coding standards with efficient software and hardware implementations.

Regarding coding efficiency, the AMLC model outperforms the SHVC reference software, achieving a BD-Rate gain of -62.42%. Furthermore, AMLC overcomes the challenges posed by using a less coding-efficient BL and the lack of inter-layer dependencies on the encoding side. It produces similar coding efficiency to the fully learned multi-layer model and its equivalent implementation with residual connections on both the encoding and decoding sides. This observation emphasizes the advantage of using an asymmetric learned compression solution at the EL. The analysis corroborates the reported gains considering multiple quality metrics and datasets.

7.1 FUTURE WORKS

Although we brought an asymmetric multi-layer solution to the discussion, we recognize that future improvements are possible. We implemented a model and validated the AMLC solution against the relevant codecs, but the asymmetric approach and the

model presented can serve as a framework. Therefore, we envisage experiments with other components, aiming to reduce complexity – by adopting a less complex learned compression solution for the EL and Merge Block.

We performed experiments with AMLC on single frames in this work. Considering the observations on the framework's partial implementation, expanding AMLC with the temporal and spatial upsampling combination is natural. However, carefully considering the learned coding complexity, which is already significant for single-frame coding, is required. Moreover, video content may vary along different scenes, affecting the bitrate. This work and related ones demonstrate that learned compression can achieve higher coding efficiency than handcrafted codecs. However, extending AMLC to videos requires more research on rate control algorithms to address bitrate and bandwidth variations when transmitting the media.

APPENDIX A – SEARCH STRINGS

This appendix contains the search string used for the systematic review.

A.1 VIDEO AND IMAGE COMPRESSION USING NN SOLUTIONS

The first search string focus on NN solution but comprises 'machine learning' strategies – not excluding decision tree and SVM solutions – for both video and image compression. The following search string was used on the Scopus platform:

TITLE-ABS-KEY ((video OR image) AND (compression OR encoding OR coding) AND ("machine learning" OR "neural network" OR "NN" OR "convolutional neural network" OR cnn OR autoencoder OR autoencoders) AND NOT (medical OR forensic OR texture)) AND (LIMIT-TO (SRCTYPE , "p") OR LIMIT-TO (SRCTYPE , "j")) AND (LIMIT-TO (DOCTYPE , "cp") OR LIMIT-TO (DOCTYPE , "ar")) AND (LIMIT-TO (PUBYEAR , 2020) OR LIMIT-TO (PUBYEAR , 2019) OR LIMIT-TO (PUBYEAR , 2018) OR LIMIT-TO (PUBYEAR , 2017)) AND (LIMIT-TO (LANGUAGE , "English")) AND (LIMIT-TO (EXACTSRCTITLE , "IEEE Access") OR LIMIT-TO (EXACTSRCTITLE , "Proceedings International Conference On Image Processing Icip") OR LIMIT-TO (EXACTSRCTITLE , "Proceedings Of SPIE The International Society For Optical Engineering") OR LIMIT-TO (EXACTSRCTITLE , "IEEE Transactions On Image Processing") OR LIMIT-TO (EXACTSRCTITLE , "Data Compression Conference Proceedings") OR LIMIT-TO (EXACTSRCTITLE , "Multimedia Tools And Applications") OR LIMIT-TO (EXACTSRCTITLE , "IEEE Transactions On Circuits And Systems For Video Technology") OR LIMIT-TO (EXACTSRCTITLE , "IEEE Transactions On Neural Networks And Learning Systems") OR LIMIT-TO (EXACTSRCTITLE , "Eurasip Journal On Image And Video Processing") OR LIMIT-TO (EXACTSRCTITLE , "Journal Of Real Time Image Processing") OR LIMIT-TO (EXACTSRCTITLE , "Mm 2017 Proceedings Of The 2017 ACM Multimedia Conference") OR LIMIT-TO (EXACTSRCTITLE , "Signal Image And Video Processing") OR LIMIT-TO (EXACTSRCTITLE , "ACM Transactions On Graphics") OR LIMIT-TO (EXACTSRCTITLE , "Digital Signal Processing A Review Journal") OR LIMIT-TO (EXACTSRCTITLE , "IEEE Transactions On Broadcasting") OR LIMIT-TO (EXACTSRCTITLE , "2016 Picture Coding Symposium Pcs 2016") OR LIMIT-TO (EXACTSRCTITLE , "2017 IEEE International Conference On Multimedia And Expo Workshops Icmew 2017") OR LIMIT-TO (EXACTSRCTITLE , "2018 25th IEEE International Conference On Electronics Circuits And Systems Icecs 2018") OR LIMIT-TO (EXACTSRCTITLE , "2018 IEEE International Conference On Multimedia And Expo Workshops Icmew 2018") OR LIMIT-TO (EXACTSRCTITLE , "ACM Transactions On Multimedia Computing Communications And Applications") OR LIMIT-TO

(EXACTSRCTITLE , "IEEE Signal Processing Letters") OR LIMIT-TO (EXACTSRCTITLE , "International Conference On Communication Technology Proceedings ICCT") OR LIMIT-TO (EXACTSRCTITLE , "Multidimensional Systems And Signal Processing") OR LIMIT-TO (EXACTSRCTITLE , "Signal Processing") OR LIMIT-TO (EXACTSRCTITLE , "2016 6th International Conference On Image Processing Theory Tools And Applications Ipta 2016") OR LIMIT-TO (EXACTSRCTITLE , "2017 IEEE Global Conference On Signal And Information Processing Globalsip 2017 Proceedings") OR LIMIT-TO (EXACTSRCTITLE , "2017 International Symposium On Intelligent Signal Processing And Communication Systems Ispacs 2017 Proceedings") OR LIMIT-TO (EXACTSRCTITLE , "2018 5th International Conference On Signal Processing And Integrated Networks Spin 2018") OR LIMIT-TO (EXACTSRCTITLE , "2018 IEEE 20th International Workshop On Multimedia Signal Processing Mmsp 2018") OR LIMIT-TO (EXACTSRCTITLE , "25th European Signal Processing Conference Eusipco 2017") OR LIMIT-TO (EXACTSRCTITLE , "9th International Symposium On Signal Image Video And Communications Isivc 2018 Proceedings"))

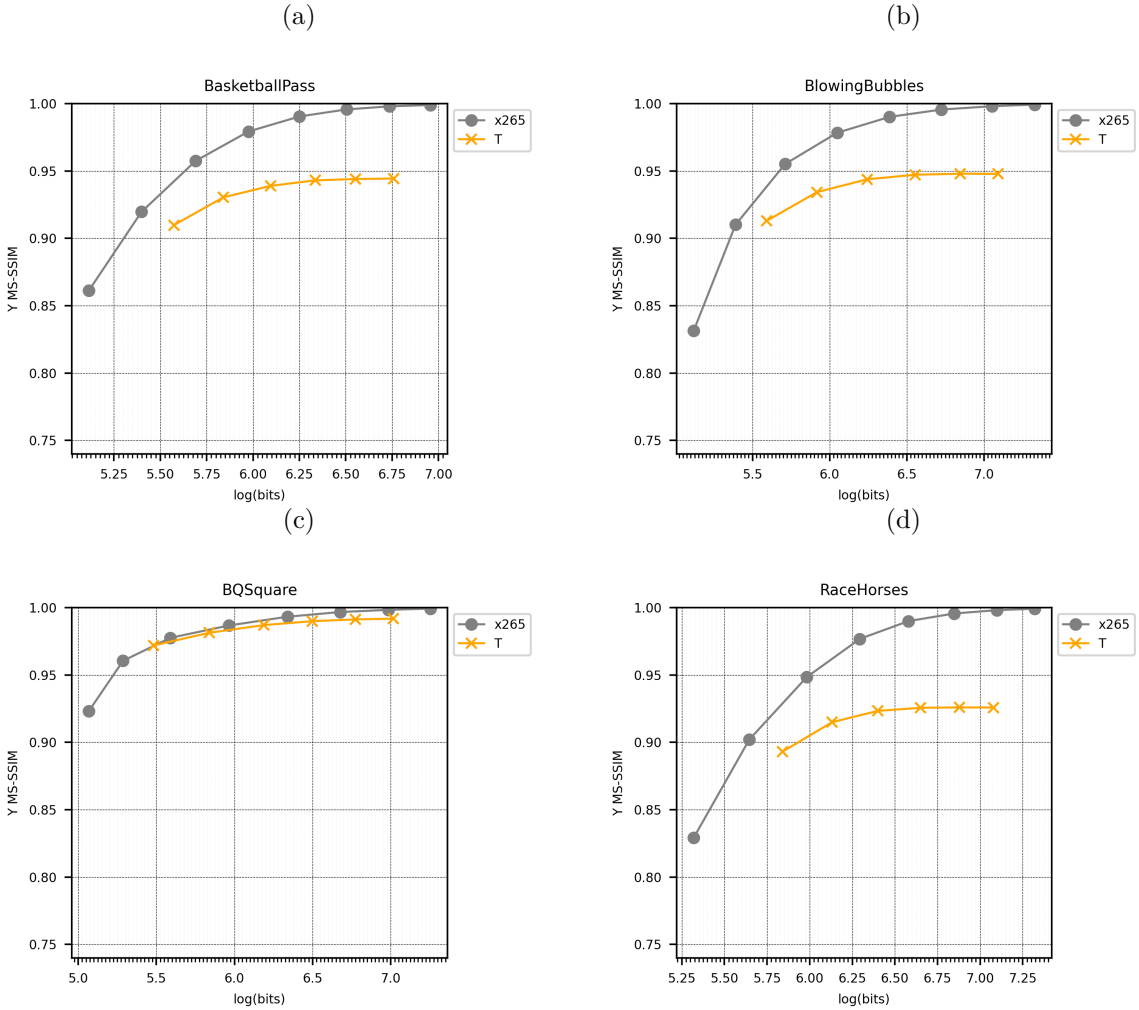
A.2 VIDEO COMPRESSION USING NN SOLUTIONS

The second search string was based on the previous one, but restricting the scope on video compression topic. The following search string was used on the Scopus platform: TITLE-ABS-KEY ((video) AND (compression OR encoding OR coding) AND ("machine learning" OR "neural network" OR "NN" OR "convolutional neural network" OR cnn OR autoencoder OR autoencoders) AND NOT (medical OR forensic)) AND (LIMIT-TO (SRCTYPE,"p") OR LIMIT-TO (SRCTYPE,"j")) AND (LIMIT-TO (DOCTYPE,"cp") OR LIMIT-TO (DOCTYPE,"ar")) AND (LIMIT-TO (PUBYEAR,2020) OR LIMIT-TO (PUBYEAR,2019) OR LIMIT-TO (PUBYEAR,2018) OR LIMIT-TO (PUBYEAR,2017)) AND (LIMIT-TO (EXACTSRCTITLE,"IEEE Access") OR LIMIT-TO (EXACTSRCTITLE,"Proceedings International Conference On Image Processing Icip") OR LIMIT-TO (EXACTSRCTITLE,"Proceedings Of SPIE The International Society For Optical Engineering") OR LIMIT-TO (EXACTSRCTITLE,"IEEE Transactions On Image Processing") OR LIMIT-TO (EXACTSRCTITLE,"Data Compression Conference Proceedings") OR LIMIT-TO (EXACTSRCTITLE,"Multimedia Tools And Applications") OR LIMIT-TO (EXACTSRCTITLE,"IEEE Transactions On Circuits And Systems For Video Technology") OR LIMIT-TO (EXACTSRCTITLE,"IEEE Transactions On Neural Networks And Learning Systems") OR LIMIT-TO (EXACTSRCTITLE,"Eurasip Journal On Image And Video Processing") OR LIMIT-TO (EXACTSRCTITLE,"Journal Of Real Time Image Processing") OR LIMIT-TO (EXACTSRCTITLE,"Mm 2017 Proceedings Of The 2017 ACM Multimedia Conference") OR LIMIT-TO (EXACTSRCTITLE,"Signal Image And Video Processing") OR LIMIT-TO (EXACTSRCTITLE,"ACM Transactions On Graphics") OR LIMIT-

TO (EXACTSRCTITLE,"Digital Signal Processing A Review Journal") OR LIMIT-TO (EXACTSRCTITLE,"IEEE Transactions On Broadcasting") OR LIMIT-TO (EXACTSRCTITLE,"2016 Picture Coding Symposium Pcs 2016") OR LIMIT-TO (EXACTSRCTITLE,"2017 IEEE International Conference On Multimedia And Expo Workshops Icmew 2017") OR LIMIT-TO (EXACTSRCTITLE,"2018 25th IEEE International Conference On Electronics Circuits And Systems Icecs 2018") OR LIMIT-TO (EXACTSRCTITLE,"2018 IEEE International Conference On Multimedia And Expo Workshops Icmew 2018") OR LIMIT-TO (EXACTSRCTITLE,"ACM Transactions On Multimedia Computing Communications And Applications") OR LIMIT-TO (EXACTSRCTITLE,"IEEE Signal Processing Letters") OR LIMIT-TO (EXACTSRCTITLE,"International Conference On Communication Technology Proceedings ICCT") OR LIMIT-TO (EXACTSRCTITLE,"Multidimensional Systems And Signal Processing") OR LIMIT-TO (EXACTSRCTITLE,"Signal Processing") OR LIMIT-TO (EXACTSRCTITLE,"2016 6th International Conference On Image Processing Theory Tools And Applications Ipta 2016") OR LIMIT-TO (EXACTSRCTITLE,"2017 IEEE Global Conference On Signal And Information Processing Globalsip 2017 Proceedings") OR LIMIT-TO (EXACTSRCTITLE,"2017 International Symposium On Intelligent Signal Processing And Communication Systems Ispacs 2017 Proceedings") OR LIMIT-TO (EXACTSRCTITLE,"2018 5th International Conference On Signal Processing And Integrated Networks Spin 2018") OR LIMIT-TO (EXACTSRCTITLE,"2018 IEEE 20th International Workshop On Multimedia Signal Processing Mmsp 2018") OR LIMIT-TO (EXACTSRCTITLE,"25th European Signal Processing Conference Eusipco 2017") OR LIMIT-TO (EXACTSRCTITLE,"9th International Symposium On Signal Image Video And Communications Isivc 2018 Proceedings")) AND (LIMIT-TO (LANGUAGE,"English"))

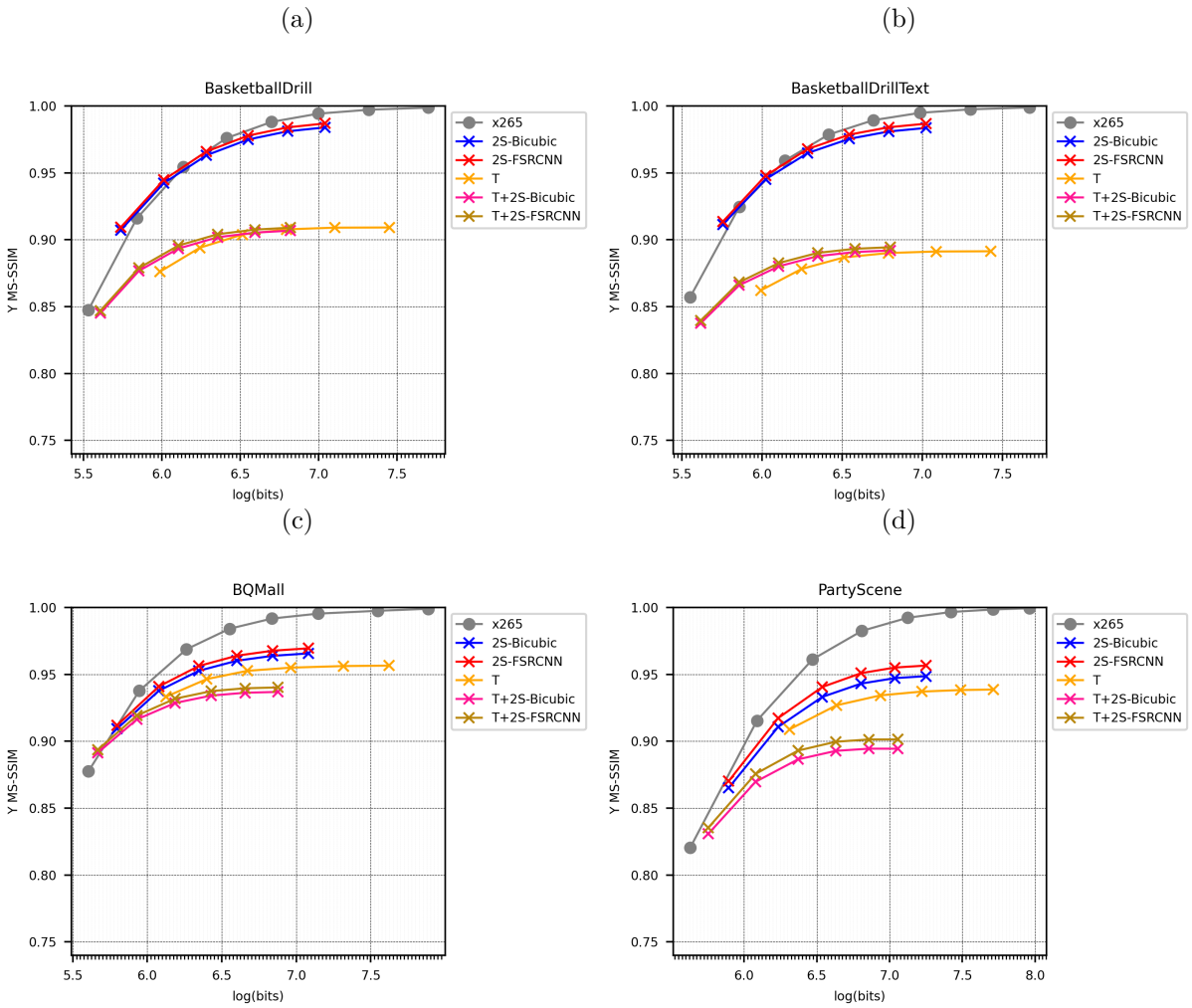
APPENDIX B – RATE-DISTORTION CURVES

Figure 42 – Rate-Distortion Curves A.



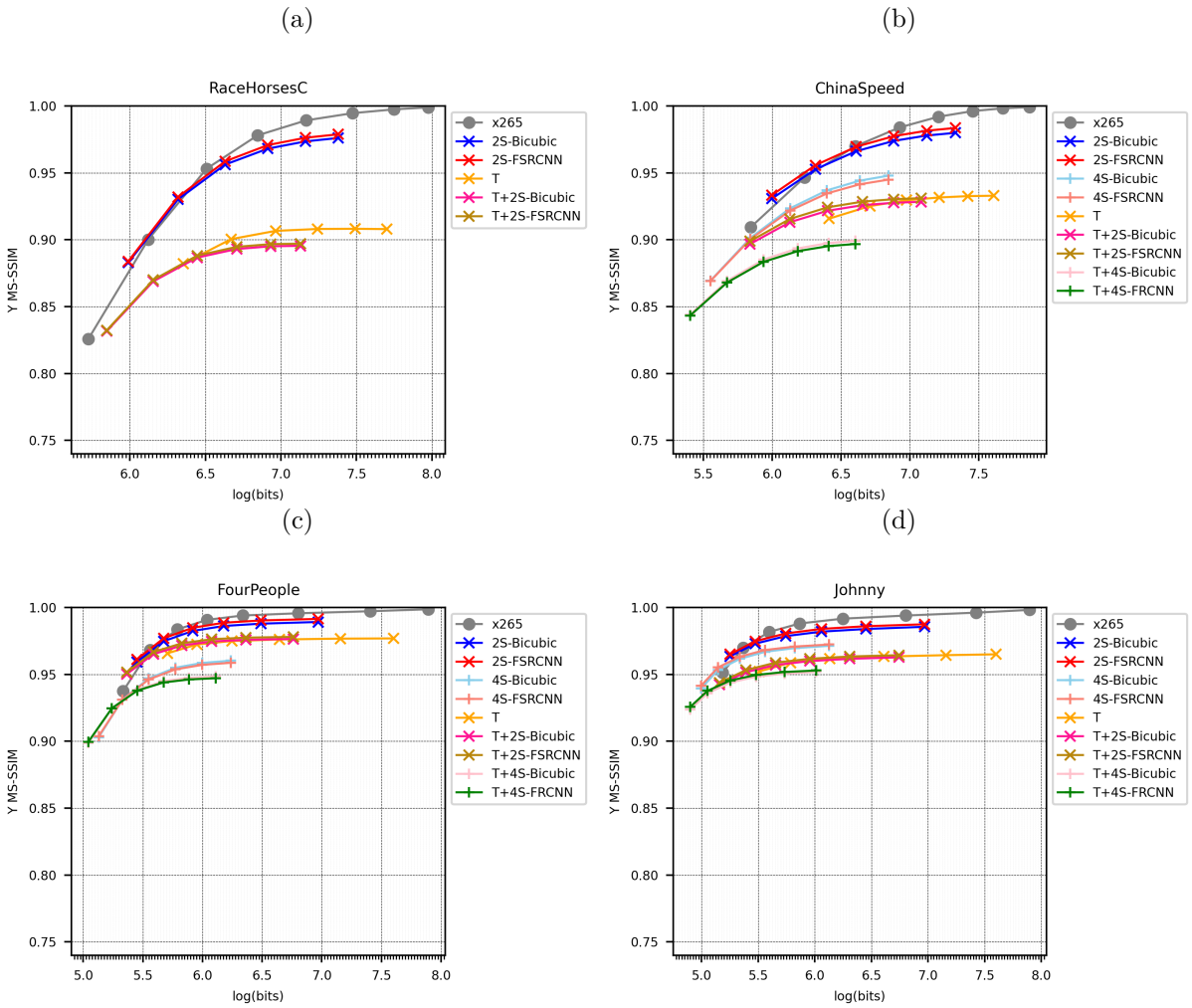
Source: the author (2021).

Figure 43 – Rate-Distortion Curves B.



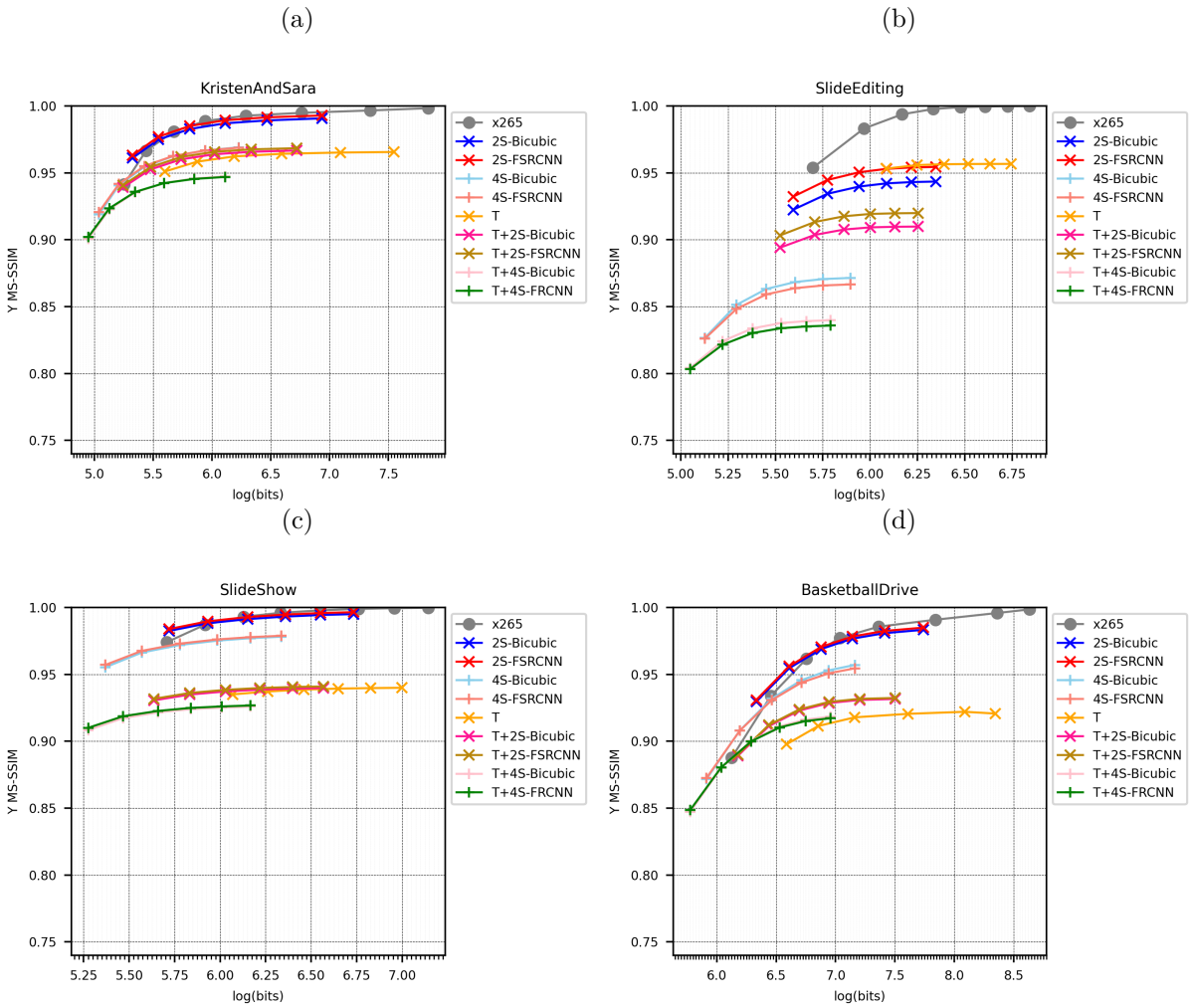
Source: the author (2021).

Figure 44 – Rate-Distortion Curves C.



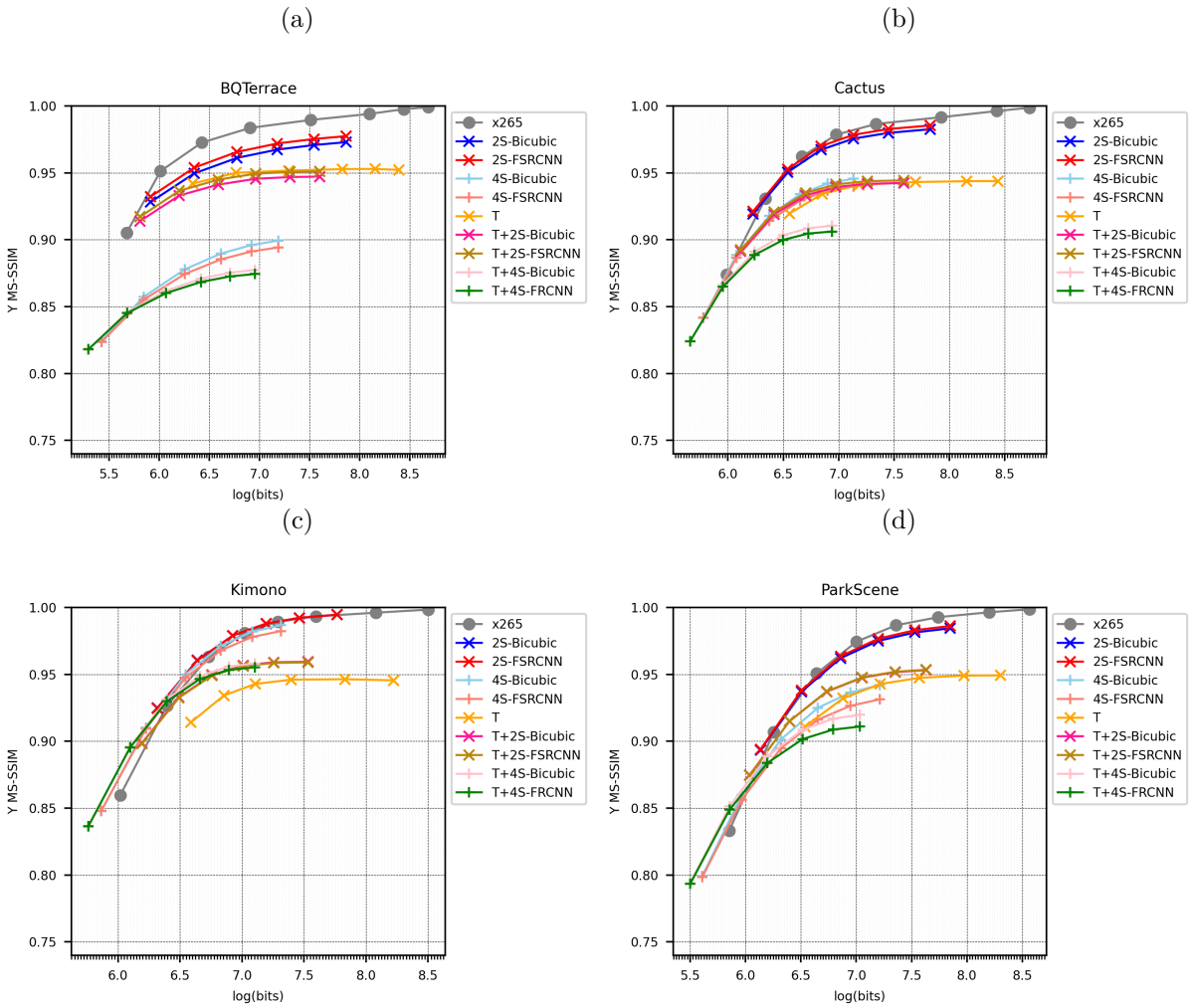
Source: the author (2021).

Figure 45 – Rate-Distortion Curves D.



Source: the author (2021).

Figure 46 – Rate-Distortion Curves E.



Source: the author (2021).

BIBLIOGRAPHY

AFONSO, M.; ZHANG, F.; BULL, D. R. Spatial resolution adaptation framework for video compression. In: TESCHER, A. G. (Ed.). **Applications of Digital Image Processing XLI**. SPIE, 2018. v. 10752, p. 107520L. Available from Internet: https://doi.org/10.1117/12.2320520.

AFONSO, M.; ZHANG, F.; BULL, D. R. Video Compression Based on Spatio-Temporal Resolution Adaptation. **IEEE Transactions on Circuits and Systems for Video Technology**, Institute of Electrical and Electronics Engineers Inc., v. 29, n. 1, p. 275–280, 2019. ISSN 10518215.

AGGARWAL, C. C. **Neural Networks and Deep Learning: A Textbook**. 1st. ed. [S.l.]: Springer Publishing Company, Incorporated, 2018. ISBN 3319944622.

AGUSTSSON, E. et al. Scale-Space Flow for End-to-End Optimized Video Compression. In: **2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2020. p. 8500–8509.

Ahmed, N.; Natarajan, T.; Rao, K. R. Discrete cosine transform. **IEEE Transactions on Computers**, C-23, n. 1, p. 90–93, 1974.

BALAJI, L.; THYAGHARAJAN, K. K. An enhanced performance for H.265/SHVC based on combined AEGBM3D filter and back-propagation neural network. **Signal, Image and Video Processing**, Springer London, v. 12, n. 5, p. 809–817, 2018. ISSN 18631703.

BALLÉ, J.; LAPARRA, V.; SIMONCELLI, E. P. End-to-end optimization of nonlinear transform codes for perceptual quality. In: **2016 Picture Coding Symposium (PCS)**. [S.l.: s.n.], 2016a. p. 1–5.

BALLÉ, J.; LAPARRA, V.; SIMONCELLI, E. P. End-to-end optimized image compression. **CoRR**, abs/1611.01704, 2016b.

BALLÉ, J. et al. Variational image compression with a scale hyperprior. In: **International Conference on Learning Representations**. [S.l.: s.n.], 2018.

BATTISTA, S. et al. Overview of the Low Complexity Enhancement Video Coding (LCEVC) Standard. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 32, n. 11, p. 7983–7995, 2022.

BÉGAINT, J. et al. CompressAI: a PyTorch Library and Evaluation Platform for End-to-End Compression Research. **arXiv preprint arXiv:2011.03029**, 2020.

BENJAK, M. et al. Learning-based scalable video coding with spatial and temporal prediction. In: **IEEE International Conference on Visual Communications and Image Processing (VCIP)**. [S.l.: s.n.], 2023. p. 1–5.

BJØNTEGAARD, G. **Calculation of average PSNR differences between RD-curves**. 13th VCEG Meeting, Austin, Texas, USA, 2001.

BONNINEAU, C. et al. CAESR: Conditional Autoencoder and Super-Resolution for Learned Spatial Scalability. In: **International Conference on Visual Communications and Image Processing (VCIP)**. [S.l.: s.n.], 2021. p. 1–5.

BOSSEN, F. **JCTVC-L1100: Common test conditions and software reference configurations**. 12th JCT-VC Meeting: Geneva, Switzerland, 2013.

BOYCE, J. M. et al. Overview of SHVC: Scalable Extensions of the High Efficiency Video Coding Standard. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 26, n. 1, p. 20–34, 2016.

BRADSKI, G. The OpenCV Library. **Dr. Dobb's Journal of Software Tools**, 2000.

BROSS, B. et al. Overview of the versatile video coding (vvc) standard and its applications. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 31, n. 10, p. 3736–3764, 2021.

BROSS, B. et al. Overview of the Versatile Video Coding (VVC) Standard and its Applications. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 31, n. 10, p. 3736–3764, 2021.

CHEN, G. et al. AV1 in-loop Filtering using a Wide-Activation Structured Residual Network. **Proceedings - International Conference on Image Processing, ICIP**, IEEE, v. 2019-Septe, p. 1725–1729, 2019. ISSN 15224880.

CHEN, X. et al. Activating More Pixels in Image Super-Resolution Transformer. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2023. p. 22367–22377.

CHENG, Z. et al. Learned Image Compression with Discretized Gaussian Mixture Likelihoods and Attention Modules. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2020.

Choi, G. et al. A new motion estimation method for motion-compensated frame interpolation using a convolutional neural network. In: **2017 IEEE International Conference on Image Processing (ICIP)**. [S.l.: s.n.], 2017. p. 800–804. ISSN 2381-8549.

CHOI, G.; HEO, P.; PARK, H. Triple-Frame-Based Bi-Directional Motion Estimation for Motion-Compensated Frame Interpolation. **IEEE Transactions on Circuits and Systems for Video Technology**, Institute of Electrical and Electronics Engineers Inc., v. 29, n. 5, p. 1251–1258, 2019. ISSN 10518215.

CHUNG, C. H.; PENG, W. H.; HU, J. H. HEVC/H.265 coding unit split decision using deep reinforcement learning. **2017 International Symposium on Intelligent Signal Processing and Communication Systems, ISPACS 2017 - Proceedings**, v. 2018-Janua, p. 570–575, 2017.

CISCO. **Cisco Visual Networking Index: Forecast and Methodology, 2016-2021**. [S.l.], 2017.

CISCO. **Cisco Visual Networking Index: Forecast and Trends, 2017-2022**. [S.l.], 2019.

DONG, C. et al. Learning a Deep Convolutional Network for Image Super-Resolution. In: **Computer Vision – ECCV 2014**. [S.l.]: Springer International Publishing, 2014. p. 184–199. ISBN 978-3-319-10593-2.

DONG, C.; LOY, C. C.; TANG, X. Accelerating the super-resolution convolutional neural network. In: LEIBE, B. et al. (Ed.). **Computer Vision – ECCV 2016**. Cham: Springer International Publishing, 2016. p. 391–407.

ERICSSON. **Ericsson Mobility Report**. Stockholm, Sweden, 2022.

FENG, L. et al. Coding prior based high efficiency restoration for compressed video. **2019 IEEE International Conference on Image Processing (ICIP)**, IEEE, p. 769–773, 2019.

FENG, L. et al. A dual-network based super-resolution for compressed high definition video. In: **Advances in Multimedia Information Processing – PCM 2018**. [S.l.]: Springer International Publishing, 2018. p. 600–610. ISBN 978-3-030-00776-8.

FENG, Z. et al. Fast intra CTU depth decision for HEVC. **IEEE Access**, IEEE, v. 6, p. 45262–45269, 2018. ISSN 21693536.

FFMPEG, D. **FFmpeg project**. 2020. Available from Internet: http://ffmpeg.org/.

GALPIN, F. et al. CNN-Based Driving of Block Partitioning for Intra Slices Encoding. In: Bilgin A. Storer J.A., S.-S. J. M. M. W. (Ed.). **Data Compression Conference Proceedings**. [S.l.]: Institute of Electrical and Electronics Engineers Inc., 2019. v. 2019-March, p. 162–171. ISBN 9781728106571. ISSN 10680314.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [S.l.]: MIT Press, 2016. http://www.deeplearningbook.org.

HE, X. et al. Enhancing HEVC Compressed Videos with a Partition-Masked Convolutional Neural Network. **Proceedings - International Conference on Image Processing, ICIP**, IEEE, p. 216–220, 2018. ISSN 15224880.

HELLE, P. et al. Intra Picture Prediction for Video Coding with Neural Networks. In: Bilgin A. Storer J.A., S.-S. J. M. M. W. (Ed.). **Data Compression Conference Proceedings**. [S.l.]: Institute of Electrical and Electronics Engineers Inc., 2019. v. 2019-March, p. 448–457. ISBN 9781728106571. ISSN 10680314.

HERGLOTZ, C. et al. The Bjøntegaard Bible Why Your Way of Comparing Video Codecs May Be Wrong. **IEEE Transactions on Image Processing**, Institute of Electrical and Electronics Engineers (IEEE), v. 33, p. 987–1001, 2024. ISSN 1941-0042.

HUANG, C. et al. Efficient CU and PU decision based on neural network and gray level co-occurrence matrix for intra prediction of screen content coding. **IEEE Access**, Institute of Electrical and Electronics Engineers Inc., v. 6, p. 46643–46655, 2018. ISSN 21693536.

HUANG, Z. et al. Real-Time Intermediate Flow Estimation for Video Frame Interpolation. In: SPRINGER. **Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XIV**. [S.l.], 2022. p. 624–642.

ILG, E. et al. Flownet 2.0: Evolution of optical flow estimation with deep networks. In: **IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2017.

ISO Central Secretary. **Information technology – Coded representation of immersive media – Part 3: Versatile video coding**. Geneva, Switzerland, 2020.

ITU-T. **Recommendation ITU-T H.264: Advanced video coding for generic audiovisual services**. [S.l.], 2017.

ITU-T. **Recommendation ITU-T H.266: Versatile video coding**. Geneva, Switzerland, 2020.

JIA, C. et al. Content-Aware Convolutional Neural Network for In-Loop Filtering in High Efficiency Video Coding. **IEEE Transactions on Image Processing**, Institute of Electrical and Electronics Engineers Inc., v. 28, n. 7, p. 3343–3356, 2019. ISSN 10577149.

JIMBO, S.; WANG, J.; YASHIMA, Y. Block adaptive CNN/HEVC interframe prediction for video coding. In: Kemao Q. Hayase K., S. S. L. Y.-L. L. P. Y. L. W.-N. Y. L. (Ed.). **Proceedings of SPIE - The International Society for Optical Engineering**. [S.l.]: SPIE, 2019. v. 11049. ISBN 9781510627734. ISSN 0277786X.

JIN, Z. et al. CNN oriented fast QTBT partition algorithm for JVET intra coding. **2017 IEEE Visual Communications and Image Processing, VCIP 2017**, v. 2018-Janua, p. 1–4, 2018.

JIN, Z. et al. Fast QTBT partition algorithm for intra frame coding through convolutional neural network. **IEEE Access**, IEEE, v. 6, p. 54660–54673, 2018. ISSN 21693536.

JPEG. **JPEG AI Quality Assessment Framework**. 2023. Available from Internet: gitlab.com/wg1/jpeg-ai/jpeg-ai-qaf.

KANG, J.; KIM, S.; LEE, K. M. Multi-modal/multi-scale convolutional neural network based in-loop filter design for next generation video codec. In: **2017 IEEE International Conference on Image Processing (ICIP)**. [S.l.: s.n.], 2017. p. 26–30.

KAPPELER, A. et al. Super-resolution of compressed videos using convolutional neural networks. **Proceedings - International Conference on Image Processing, ICIP**, IEEE, v. 2016-Augus, p. 1150–1154, 2016. ISSN 15224880.

KEYS, R. Cubic convolution interpolation for digital image processing. **IEEE Transactions on Acoustics, Speech, and Signal Processing**, v. 29, n. 6, p. 1153–1160, 1981.

KIM, I. K. et al. Block partitioning structure in the hevc standard. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 22, n. 12, p. 1697–1706, 12 2012. ISSN 1051-8215.

KIM, J. et al. Dynamic frame resizing with convolutional neural network for efficient video compression. In: TESCHER, A. G. (Ed.). **Applications of Digital Image Processing XL**. SPIE, 2017. v. 10396, p. 103961R. Available from Internet: https://doi.org/10.1117/12.2270737.

KIM, K.; RO, W. W. Fast CU Depth Decision for HEVC Using Neural Networks. **IEEE Transactions on Circuits and Systems for Video Technology**, Institute of Electrical and Electronics Engineers Inc., v. 29, n. 5, p. 1462–1473, 2019. ISSN 10518215.

KUANAR, S.; RAO, K. R.; CONLY, C. Fast mode decision in HEVC intra prediction, using region wise CNN feature classification. **2018 IEEE International Conference on Multimedia and Expo Workshops, ICMEW 2018**, IEEE, p. 1–4, 2018.

LAUDE, T.; OSTERMANN, J. Deep learning-based intra prediction mode decision for HEVC. **2016 Picture Coding Symposium, PCS 2016**, 2017.

LEE, J.; CHO, S.; BEACK, S.-K. **Context-adaptive Entropy Model for End-to-end Optimized Image Compression**. 2019.

LEE, J. et al. High efficient energy compaction network for image transform. In: A.G., T. (Ed.). **Proceedings of SPIE - The International Society for Optical Engineering**. [S.l.]: SPIE, 2018. v. 10752. ISBN 9781510620759. ISSN 0277786X.

LEE, W.-C. et al. A Hybrid Layered Image Compressor with Deep-Learning Technique. In: **2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP)**. [S.l.: s.n.], 2020. p. 1–6.

LI, F.; TAN, W.; YAN, B. Deep Residual Network for Enhancing Quality of the Decoded Intra Frames of Hevc. In: **Proceedings - International Conference on Image Processing, ICIP**. [S.l.]: IEEE Computer Society, 2018. p. 3918–3922. ISBN 9781479970612. ISSN 15224880.

LI, H.; TROCAN, M. Deep neural network based single pixel prediction for unified video coding. **Neurocomputing**, v. 272, p. 558–570, 2018.

LI, J. et al. Fully Connected Network-Based Intra Prediction for Image Coding. **IEEE Transactions on Image Processing**, IEEE, v. 27, n. 7, p. 3236–3247, 2018. ISSN 10577149.

LI, J. et al. Intra prediction using fully connected network for video coding. In: **Proceedings - International Conference on Image Processing, ICIP**. [S.l.]: IEEE Computer Society, 2018. v. 2017-Septe, p. 1–5. ISBN 9781509021758. ISSN 15224880.

LI, N. et al. Reinforcement learning based coding unit early termination algorithm for high efficiency video coding. **Journal of Visual Communication and Image Representation**, v. 60, p. 276 – 286, 2019. ISSN 1047-3203.

LI, T. et al. A DenseNet Based Approach for Multi-frame In-loop Filter in HEVC. In: Bilgin A. Storer J.A., S.-S. J. M. M. W. (Ed.). **Data Compression Conference Proceedings**. [S.l.]: Institute of Electrical and Electronics Engineers Inc., 2019. v. 2019-March, p. 270–279. ISBN 9781728106571. ISSN 10680314.

LI, Y. et al. A Hybrid Neural Network for Chroma Intra Prediction. In: **Proceedings - International Conference on Image Processing, ICIP**. [S.l.]: IEEE Computer Society, 2018. p. 1797–1801. ISBN 9781479970612. ISSN 15224880.

LI, Y. et al. Convolutional Neural Network-Based Block Up-Sampling for Intra Frame Coding. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 28, n. 9, p. 2316–2330, 2018. ISSN 10518215.

LI, Z. et al. **Toward A Practical Perceptual Video Quality Metric**. 2016. Accessed May 11, 2020. Available from Internet: https://netflixtechblog.com/toward-a-practical-perceptual-video-quality-metric-653f208b9652.

LIANG, J. et al. Recurrent Video Restoration Transformer with Guided Deformable Attention. **arXiv preprint arXiv:2206.02146**, 2022.

LIM, B. et al. Enhanced deep residual networks for single image super-resolution. In: **2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)**. [S.l.: s.n.], 2017. p. 1132–1140.

LIN, J. et al. Convolutional neural network-based block up-sampling for hevc. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 29, n. 12, p. 3701–3715, 2019.

LIU, J. et al. One-for-All: Grouped Variation Network-Based Fractional Interpolation in Video Coding. **IEEE Transactions on Image Processing**, Institute of Electrical and Electronics Engineers Inc., v. 28, n. 5, p. 2140–2151, 2019. ISSN 10577149.

LIU, Z. et al. CNN oriented fast HEVC intra CU mode decision. **Proceedings - IEEE International Symposium on Circuits and Systems**, IEEE, v. 2016-July, p. 2270–2273, 2016. ISSN 02714310.

LU, G. et al. DVC: An End-To-End Deep Video Compression Framework. In: **2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2019. p. 10998–11007.

MA, C. et al. Convolutional Neural Network-Based Arithmetic Coding of DC Coefficients for HEVC Intra Coding. In: **Proceedings - International Conference on Image Processing, ICIP**. [S.l.]: IEEE Computer Society, 2018. p. 1772–1776. ISBN 9781479970612. ISSN 15224880.

MEI, Y. et al. Learning-Based Scalable Image Compression With Latent-Feature Reuse and Prediction. **IEEE Transactions on Multimedia**, v. 24, p. 4143–4157, 2022.

MENTZER, F. J. et al. VCT: A Video Compression Transformer. In: **NeurIPS 2022**. [S.l.: s.n.], 2022.

MERCAT, A.; VIITANEN, M.; VANNE, J. Uvg dataset: 50/120fps 4k sequences for video codec analysis and development. In: _____. **Proceedings of the 11th ACM Multimedia Systems Conference**. New York, NY, USA: Association for Computing Machinery, 2020. p. 297–302. ISBN 9781450368452.

MINNEN, D.; BALLé, J.; TODERICI, G. Joint Autoregressive and Hierarchical Priors for Learned Image Compression. In: **Advances in Neural Information Processing Systems 31 (NeurIPS 2018)**. Red Hook, NY, USA: Curran Associates Inc., 2018. p. 10794–10803.

MPEG. **Exploration 41: Enhanced Compression beyond VVC Capability**. 2025. Accessed: 2025-08-30. Available from Internet: https://www.mpeg.org/standards/Explorations/41/.

NIKLAUS, S.; LIU, F. Context-Aware Synthesis for Video Frame Interpolation. In: **2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2018. p. 1701–1710.

NIKLAUS, S.; LIU, F. Softmax Splatting for Video Frame Interpolation. In: **Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2020. p. 5437–5446.

PARK, W. S.; KIM, M. CNN-based in-loop filtering for coding efficiency improvement. **2016 IEEE 12th Image, Video, and Multidimensional Signal Processing Workshop, IVMSP 2016**, IEEE, p. 1–5, 2016.

PFAFF, J. et al. Neural network based intra prediction for video coding. In: TESCHER, A. G. (Ed.). **Applications of Digital Image Processing XLI**. SPIE, 2018. v. 10752, p. 1075213. Available from Internet: https://doi.org/10.1117/12.2321273.

PONOMARENKO, N. et al. On Between-Coefficient Contrast Masking of DCT Basis Functions. **Proc of the 3rd Int Workshop on Video Processing and Quality Metrics for Consumer Electronics**, 01 2007.

PURI, S.; LASSERRE, S.; Le Callet, P. CNN-based transform index prediction in multiple transforms framework to assist entropy coding. In: **25th European Signal Processing Conference, EUSIPCO 2017**. [S.l.]: Institute of Electrical and Electronics Engineers Inc., 2017. v. 2017-Janua, p. 798–802. ISBN 9780992862671.

RICHARDSON, I. E. **The H.264 Advanced Video Compression Standard, Second Edition**. [S.l.]: John Wiley & Sons Ltd, 2010.

RONNEBERGER, O.; FISCHER, P.; BROX, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In: NAVAB, N. et al. (Ed.). **Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015**. Cham: Springer International Publishing, 2015. p. 234–241. ISBN 978-3-319-24574-4.

Soh, J. W. et al. Reduction of video compression artifacts based on deep temporal networks. **IEEE Access**, v. 6, p. 63094–63106, 2018. ISSN 2169-3536.

SONG, N. et al. CNN oriented fast PU mode decision for HEVC hardwired intra encoder. **2017 IEEE Global Conference on Signal and Information Processing, GlobalSIP 2017 - Proceedings**, v. 2018-Janua, p. 239–243, 2018.

SONG, R. et al. Neural network-based arithmetic coding of intra prediction modes in HEVC. **2017 IEEE Visual Communications and Image Processing, VCIP 2017**, v. 2018-Janua, p. 1–4, 2018.

SONG, X. et al. A Practical Convolutional Neural Network as Loop Filter for Intra Frame. In: **Proceedings - International Conference on Image Processing, ICIP**. [S.l.]: IEEE Computer Society, 2018. p. 1133–1137. ISBN 9781479970612. ISSN 15224880.

SOULEF, B. et al. Fast cu partition-based machine learning approach for reducing hevc complexity. **Journal of Real-Time Image Processing**, 12 2019.

SU, R. et al. Scalable Learned Image Compression With A Recurrent Neural Networks-Based Hyperprior. In: **2020 IEEE International Conference on Image Processing (ICIP)**. [S.l.: s.n.], 2020. p. 3369–3373.

SULLIVAN, G. J. et al. Overview of the high efficiency video coding (hevc) standard. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 22, n. 12, p. 1649–1668, 12 2012.

SULLIVAN, G. J.; WIEGAND, T. Rate-distortion optimization for video compression. **IEEE Signal Processing Magazine**, v. 15, n. 6, p. 74–90, 11 1998.

SUN, Y. et al. A Machine Learning Approach to Accurate Sequence-Level Rate Control Scheme for Video Coding. In: **Proceedings - International Conference on Image Processing, ICIP**. [S.l.]: IEEE Computer Society, 2018. p. 1013–1017. ISBN 9781479970612. ISSN 15224880.

Sze, V.; Budagavi, M. A comparison of cabac throughput for hevc/h.265 vs. avc/h.264. In: **SiPS 2013 Proceedings**. [S.l.: s.n.], 2013. p. 165–170.

SZE, V.; BUDAGAVI, M.; SULLIVAN, G. J. (Ed.). **High Efficiency Video Coding (HEVC)**. [S.l.]: Springer International Publishing, 2014.

TODERICI, G. et al. **Workshop and Challenge on Learned Image Compression (CLIC2020)**. 2020. Available from Internet: http://www.compression.cc.

TONG, J. et al. Learning-Based Multi-Frame Video Quality Enhancement. **Proceedings - International Conference on Image Processing, ICIP**, IEEE, v. 2019-Septe, p. 929–933, 2019. ISSN 15224880.

WANG, T.; CHEN, M.; CHAO, H. A Novel Deep Learning-Based Method of Improving Coding Efficiency from the Decoder-End for HEVC. **Data Compression Conference Proceedings**, Part F1277, p. 410–419, 2017. ISSN 10680314.

WANG, T. et al. The multi-scale deep decoder for the standard HEVC bitstreams. **Data Compression Conference Proceedings**, v. 2018-March, p. 197–206, 2018. ISSN 10680314.

WANG, Z.; SIMONCELLI, E.; BOVIK, A. Multiscale Structural Similarity for Image Quality Assessment. In: **The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003**. [S.l.: s.n.], 2003. v. 2, p. 1398–1402 Vol.2.

WANG, Z. et al. Fast QTBT Partitioning Decision for Interframe Coding with Convolution Neural Network. In: **Proceedings - International Conference on Image Processing, ICIP**. [S.l.]: IEEE Computer Society, 2018. p. 2550–2554. ISBN 9781479970612. ISSN 15224880.

WIEGAND, T. et al. Overview of the h.264/avc video coding standard. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 13, n. 7, p. 560–576, 7 2003. ISSN 1051-8215.

XIA, S. et al. A group variational transformation neural network for fractional interpolation of video coding. In: Bilgin A. Storer J.A., S.-S. J. M. M. W. (Ed.). **Data Compression Conference Proceedings**. [S.l.]: Institute of Electrical and Electronics Engineers Inc., 2018. v. 2018-March, p. 127–136. ISBN 9781538648834. ISSN 10680314.

Xu, J.; Joshi, R.; Cohen, R. A. Overview of the emerging hevc screen content coding extension. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 26, n. 1, p. 50–62, 2016.

XU, M. et al. Reducing complexity of HEVC: A deep learning approach. **IEEE Transactions on Image Processing**, IEEE, v. 27, n. 10, p. 5044–5059, 2018. ISSN 10577149.

XUE, T. et al. Video enhancement with task-oriented flow. **International Journal of Computer Vision (IJCV)**, Springer, v. 127, n. 8, p. 1106–1125, 2019.

YAN, N. et al. A convolutional neural network approach for half-pel interpolation in video coding. **Proceedings - IEEE International Symposium on Circuits and Systems**, p. 1–4, 2017. ISSN 02714310.

YAN, N. et al. Convolutional neural network-based invertible half-pixel interpolation filter for video coding. In: **Proceedings - International Conference on Image Processing, ICIP**. [S.l.]: IEEE Computer Society, 2018. p. 201–205. ISBN 9781479970612. ISSN 15224880.

YAN, N. et al. Convolutional Neural Network-Based Fractional-Pixel Motion Compensation. **IEEE Transactions on Circuits and Systems for Video Technology**, Institute of Electrical and Electronics Engineers Inc., v. 29, n. 3, p. 840–853, 2019. ISSN 10518215.

YAN, N. et al. Invertibility-driven interpolation filter for video coding. **IEEE Transactions on Image Processing**, v. 28, n. 10, p. 4912–4925, 2019. ISSN 19410042.

YANG, R. et al. Enhancing Quality for HEVC Compressed Videos. **IEEE Transactions on Circuits and Systems for Video Technology**, Institute of Electrical and Electronics Engineers Inc., v. 29, n. 7, p. 2039–2054, 2019. ISSN 10518215.

YEH, C.-H. et al. HEVC intra frame coding based on convolutional neural network. **IEEE Access**, Institute of Electrical and Electronics Engineers Inc., v. 6, p. 50087–50095, 2018. ISSN 21693536.

YU, L. et al. Quality Enhancement Network via Multi-Reconstruction Recursive Residual Learning for Video Coding. **IEEE Signal Processing Letters**, IEEE, v. 26, n. 4, p. 557–561, 2019. ISSN 10709908.

ZHANG, D. et al. A novel in-loop filtering mechanism of HEVC based on 3D sub-bands and CNN processing. **Signal, Image and Video Processing**, Springer London, 2019. ISSN 18631703.

ZHANG, F.; AFONSO, M.; BULL, D. R. Enhanced Video Compression Based on Effective Bit Depth Adaptation. **Proceedings - International Conference on Image Processing, ICIP**, IEEE, v. 2019-Septe, n. Vvc, p. 1720–1724, 2019. ISSN 15224880.

Zhang, F.; Mackin, A.; Bull, D. R. A frame rate dependent video quality metric based on temporal wavelet decomposition and spatiotemporal pooling. In: **2017 IEEE International Conference on Image Processing (ICIP)**. [S.l.: s.n.], 2017. p. 300–304. ISSN 2381-8549.

ZHANG, H. et al. Advanced cnn based motion compensation fractional interpolation. In: **2019 IEEE International Conference on Image Processing (ICIP)**. [S.l.: s.n.], 2019. p. 709–713.

ZHANG, Y.; KWONG, S.; WANG, S. Machine learning based video coding optimizations: A survey. **Information Sciences**, Elsevier Inc., v. 506, p. 395–423, 2020. ISSN 00200255.

ZHANG, Y. et al. Residual Highway Convolutional Neural Networks for in-loop Filtering in HEVC. **IEEE Transactions on Image Processing**, Institute of Electrical and Electronics Engineers Inc., v. 27, n. 8, p. 3827–3841, 2018. ISSN 10577149.

ZHAO, H. et al. A cnn-based post-processing algorithm for video coding efficiency improvement. **IEEE Access**, IEEE, v. 8, p. 920–929, 2020. ISSN 21693536.

ZHAO, L. et al. Enhanced Ctu-Level Inter Prediction with Deep Frame Rate Up-Conversion for High Efficiency Video Coding. In: **Proceedings - International Conference on Image Processing, ICIP**. [S.l.]: IEEE Computer Society, 2018. p. 206–210. ISBN 9781479970612. ISSN 15224880.

Zhao, L. et al. Enhanced motion-compensated video coding with deep virtual reference frame generation. **IEEE Transactions on Image Processing**, v. 28, n. 10, p. 4832–4844, 2019.

ZHAO, Z. et al. Enhanced Bi-prediction with Convolutional Neural Network for High Efficiency Video Coding. **IEEE Transactions on Circuits and Systems for Video Technology**, 2018.

ZHOU, S.; YE, Z.; WANG, Y. Fast HEVC CU/PU mode decision based on ANN and texture analysis. **2016 6th International Conference on Image Processing Theory, Tools and Applications, IPTA 2016**, 2017.