Perceptual video processing:

by
Dinesh Naik
(B.E, M.Tech, Ph.D)

Assistant Professor, Dept. of Information Technology
National Institute of Technology Karnataka, Surathkal

January 24, 2024

Presentation Outline

1. Machine Learning
2. Evolutionary Deep Learning
3. Pruning and Quantization for Deep Neural Network
4. Conclusion
5. References

Machine learning

- Machine Learning is the subfield of computer science that gives "computers the ability to learn without being explicitly programmed."- Arthur Samuel.
- Machine learning algorithms classify into two groups :
- Supervised Learning algorithms
- Unsupervised Learning algorithms

Introduction (The Evolution of Deep Learning)

- Cybernetics — During 1940–1960
- Connectionism — During 1980–1990
- Deep Learning — Since 2006

Introduction (The Evolution of Deep Learning)

- Cybernetics — During 1940–1960

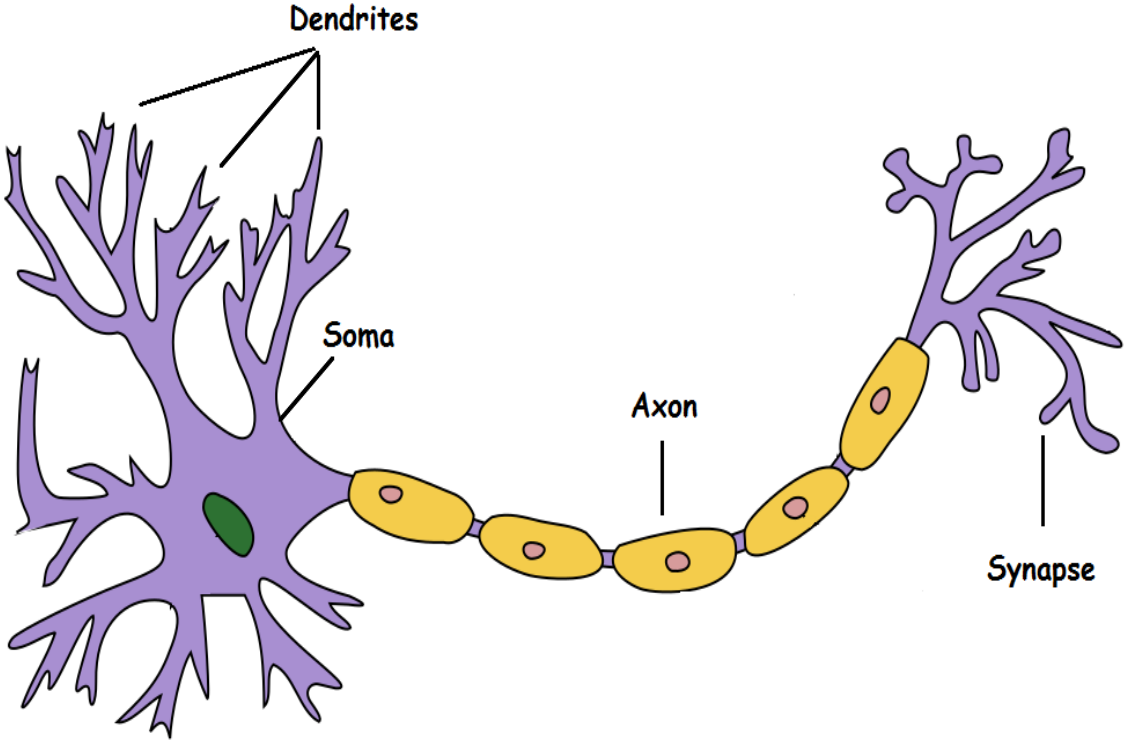


Figure: A Biological Neuron

- Dendrite: Receives signals from other neurons
- Soma: Processes the information
- Axon: Transmits the output of this neuron
- Synapse: Point of connection to other neurons

Introduction (The Evolution of Deep Learning)

- Connectionism — During 1980–1990
- Or Parallel Distributed Processing became popular in 1980s. This approach was inspired by cognitive sciences.
- Concept of Artificial Neural Network(ANNs) was introduced during this wave.

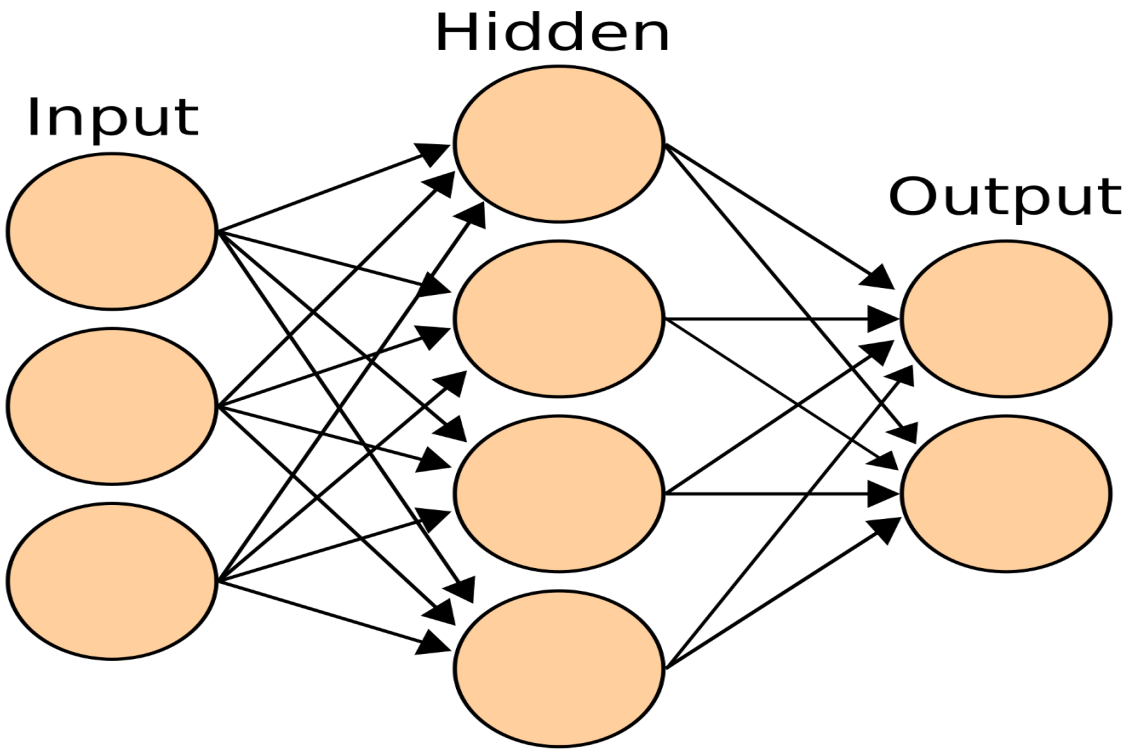


Figure: ANN

Introduction (The Evolution of Deep Learning)

- Deep Learning — Since 2006
- After two dips, the third wave emerged in 2006 with a breakthrough. Geoffrey Hinton used Greedy Layer-wise Training to train Deep Belief Networks.

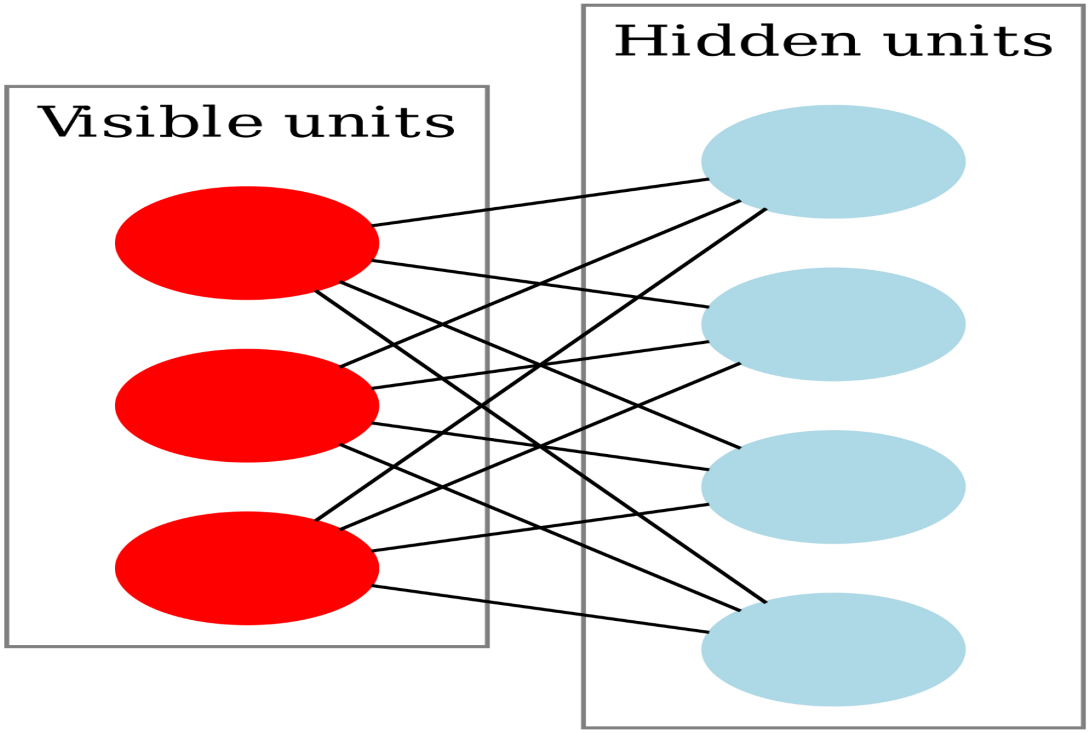


Figure: DBN

- **What is Image segmentation?**
  Image segmentation is a computer vision task in which we label specific regions of an image according to what's being shown.

Figure: Result of segmentation

Introduction (Semantic Segmentation)

- Traditional Methods
- Region-Based Models
- Fully Convolutional Network Based Models
- Refinement Network
- Weakly Supervised and Semisupervised Approaches

Introduction (Visual/Image Captioning)

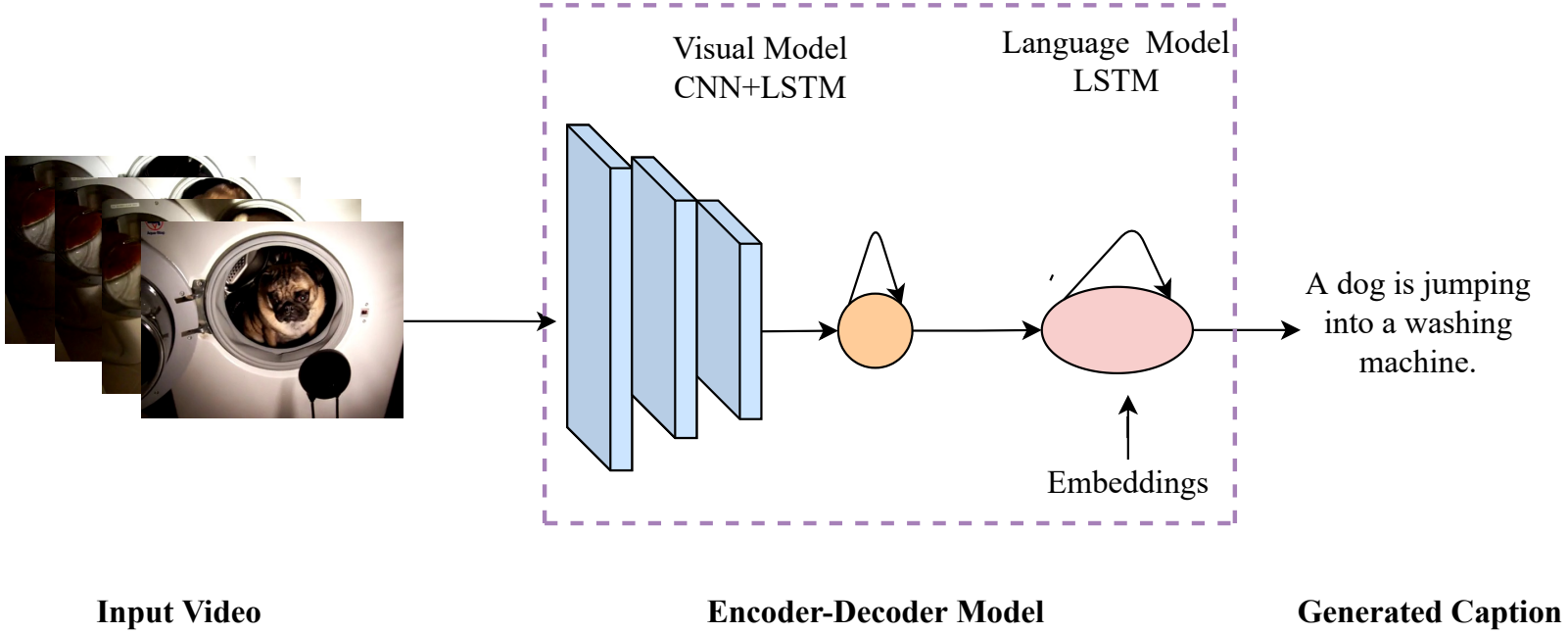


Figure: The overview video captioning framework.

Introduction (Visual/Image Captioning)

- Deep Neural Networks
- Recurrent Neural Network
- Long-Short-Term-Memory
- Attention Mechanism
- Word Embedding

Evolutionary Computation

- EC is also an important branch in AI that is efficient in optimizing complex optimization problems.
- In general, EC imitates the evolution rule of "survival of the fittest" from nature to evolve candidate solutions, so as to obtain more satisfactory solutions for optimization problems.
- EC algorithms mainly include evolutionary algorithm (EA) and swarm intelligence (SI) algorithms.
- To be more specific, representative EAs include genetic algorithm (GA), evolutionary strategy (ES) , genetic programming (GP), evolutionary programming (EP), differential evolution (DE), and estimation of distribution algorithm (EDA), while representative SI algorithms include ant colony optimization (ACO) and particle swarm optimization (PSO).

Evolutionary Computation

Table 1
Comparisons of Different Surveys about EC for DL.

Survey	Perspective and focus	Taxonomy and category	Year
Liu et al.' survey [44]	EC for neural architecture search	Encoding space/encoding strategy/population updating/efficient evaluation/application	2021
Zhou et al.' survey [45]	EC for the construction of DNNs	Optimization on architecture/parameter/both architecture and parameter/miscellaneous issues like different tasks and objectives	2021
Zhou et al.' survey [46]	advances of evolutionary neural architecture search	shallow and deep networks/and innovation/booming/trends of evolutionary DNNs	2021
Darwish et al.' survey [47]	EC for both architecture search and hyperparameter optimization	Evolutionary algorithm/swarm intelligence for DNN/CNN/DBN/RNN	2020
Bharti et al.' survey [48]	EC for both architecture search and hyperparameter optimization	EC for LSTM/CNN/GAN/AE/ESN	2020
This survey	EC in the whole DL procedure	EC for data process/model search/model training/model evaluation and utilization in the DL procedure	–

Figure: Comparisons of Different Surveys about EC for DL.

Deep learning model

- In general, widely-used DL models can be roughly divided into five categories according to their basic model, i.e., convolutional neural network (CNN), deep belief network (DBN), stacked auto-encoder (SAE), recurrent neural network (RNN) , and generative adversarial network (GAN).
- CNN is a widely-used DNN structure consisting of convolutional layers, pooling layers, and fully connected layers.
- DBN can be developed by stacking restricted Boltzmann machines (RBMs). In general, DBNs can pre-train the network in an unsupervised fashion and then fine-tunes it in a supervised way according to the labeled data, so as to obtain a more satisfactory features representation.
- SAE is composed of multiple auto-encoders (AEs) in a stacking manner. Generally speaking, an AE consists of two symmetrical components (i.e., encoder and decoder).

Deep learning model(contd.

- RNN is a special neural network that contains recurrent connections.
- GAN is a deep learning model that contains two components, i.e., generator and discriminator.
- The generator learns how to generate data similar to real training data to fool the discriminator, while the discriminator learns how to figure out the data from a real training set or generated by the generator.
- After training, the generator will have a great probability for data generation, which is meaningful in many real-world applications.

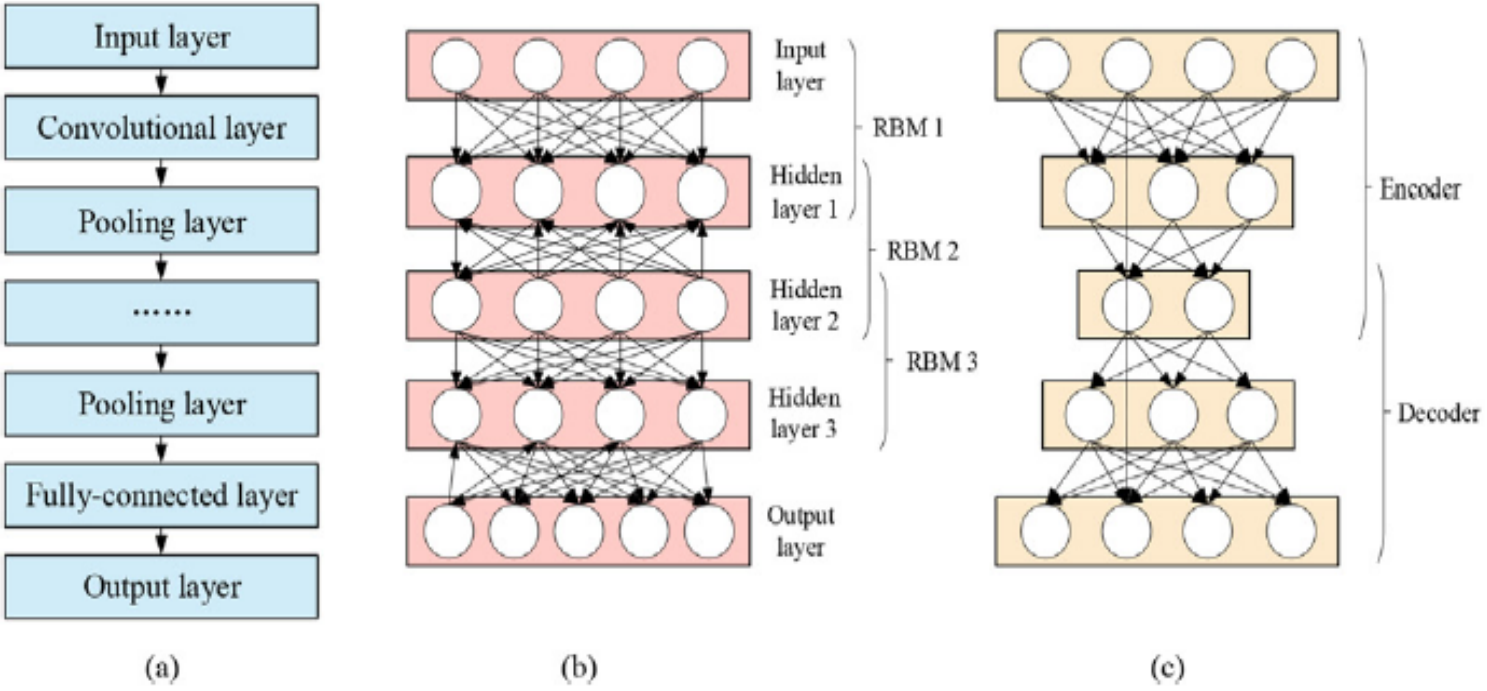


Figure: Examples of some different deep learning models: (a) CNN; (b) DBN; (c) AE.

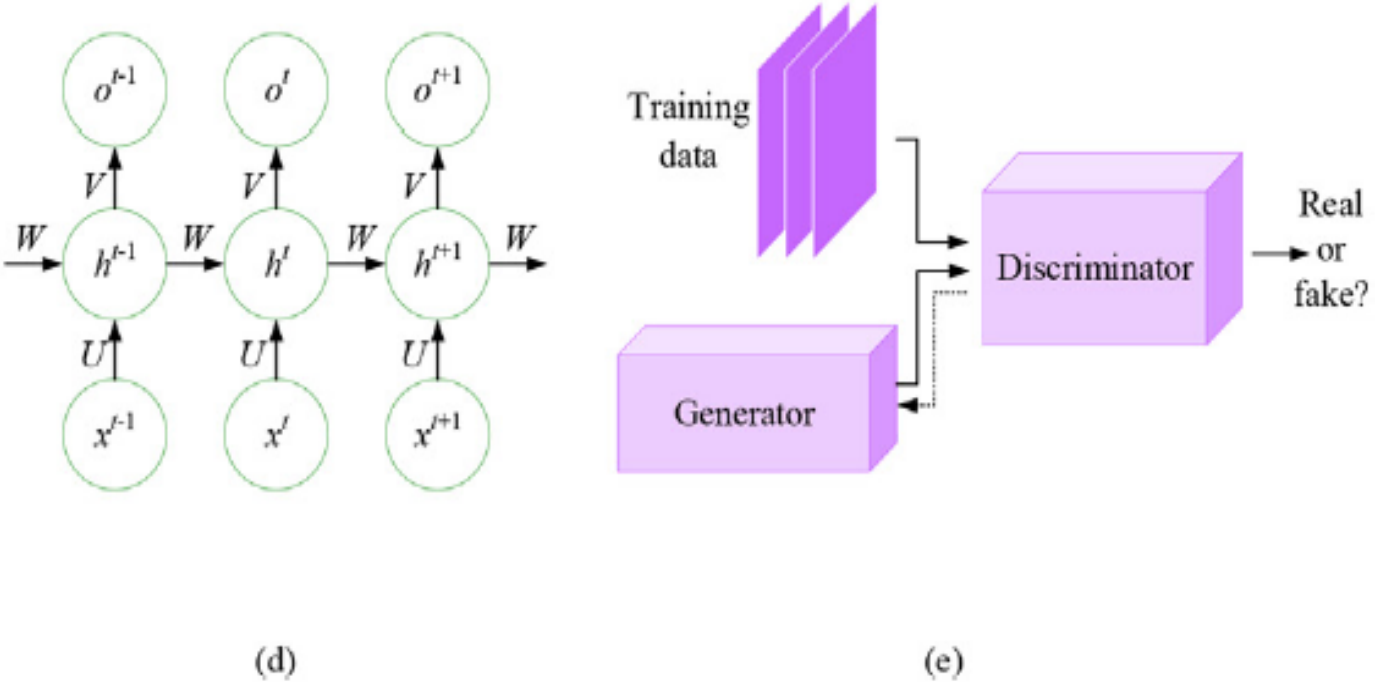


Figure: Examples of some different deep learning models:(d) RNN; and (e) GAN.

Presentation Outline

1. Machine Learning
2. Evolutionary Deep Learning
3. Pruning and Quantization for Deep Neural Network
4. Conclusion
5. References

Evolutionary Deep Learning

- The performance of DL can be influenced by many factors such as the architecture and hyperparameters, which involves data quality, model configuration and robustness, learning techniques, and problem characteristics.

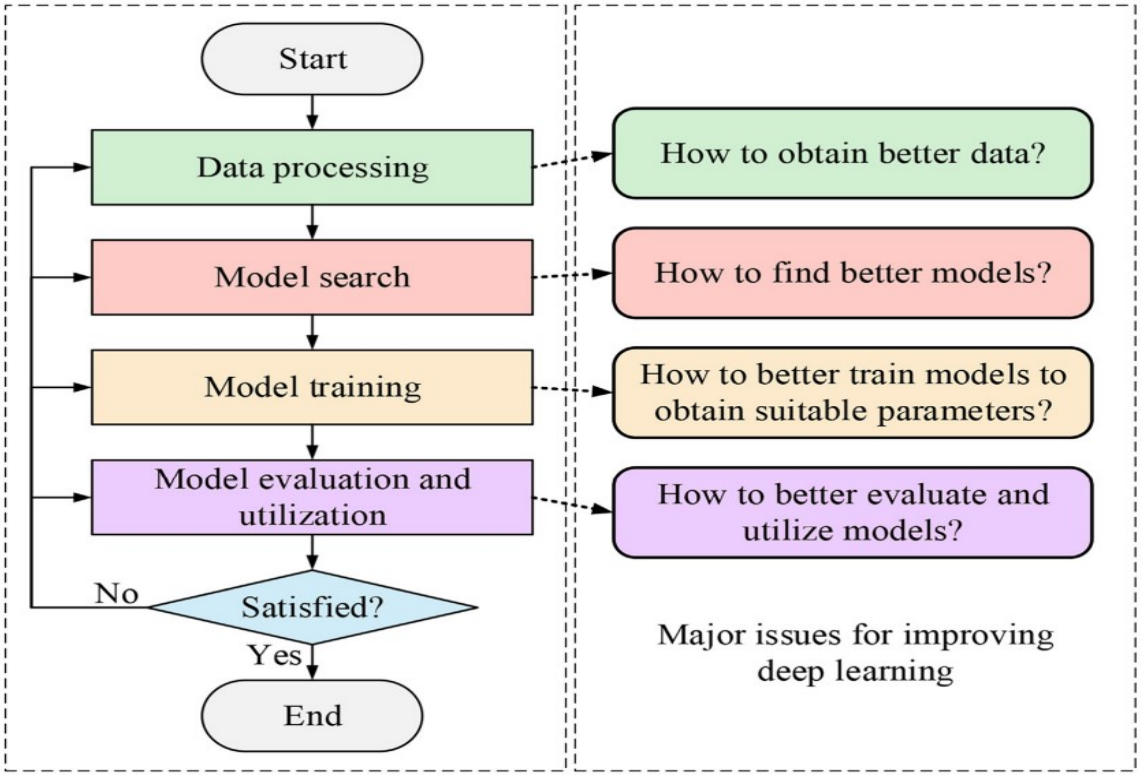


Figure: The flowchart of the whole DL lifetime with major issues for improving DL in different procedures

Evolutionary Deep Learning

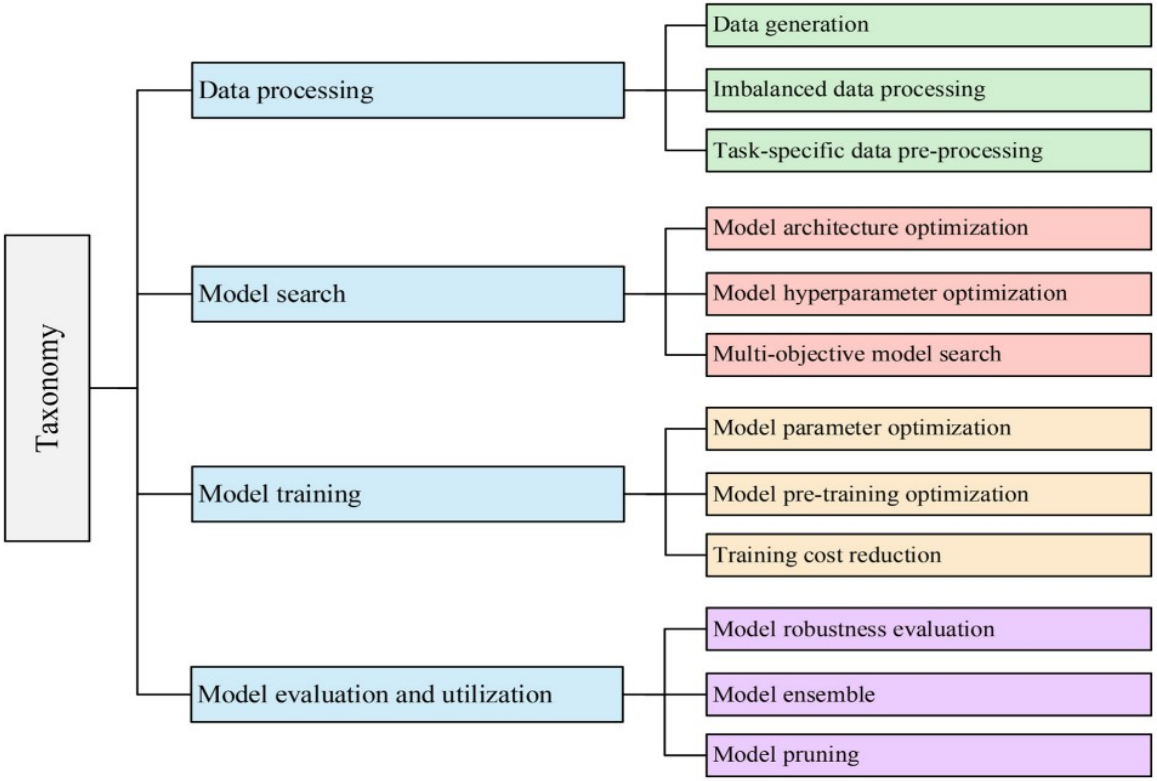


Figure: An illustration of the taxonomy.

Evolutionary Deep Learning

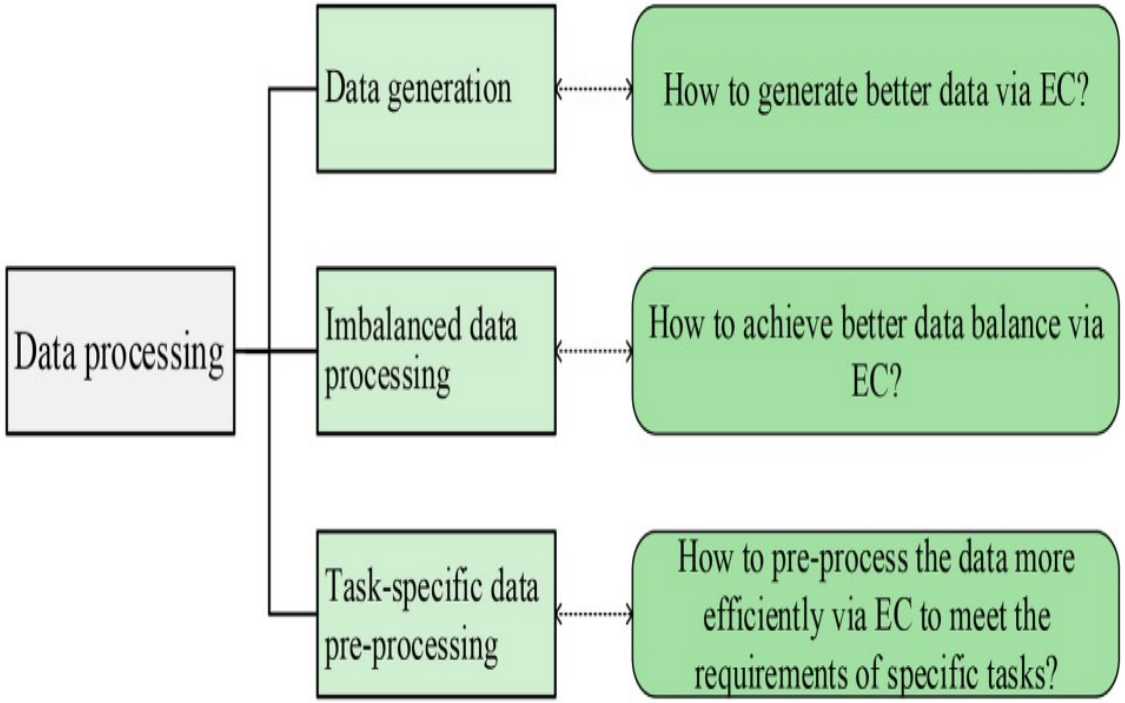


Figure: Categories and corresponding issues of EDL in the data processing procedure.

Evolutionary Deep Learning

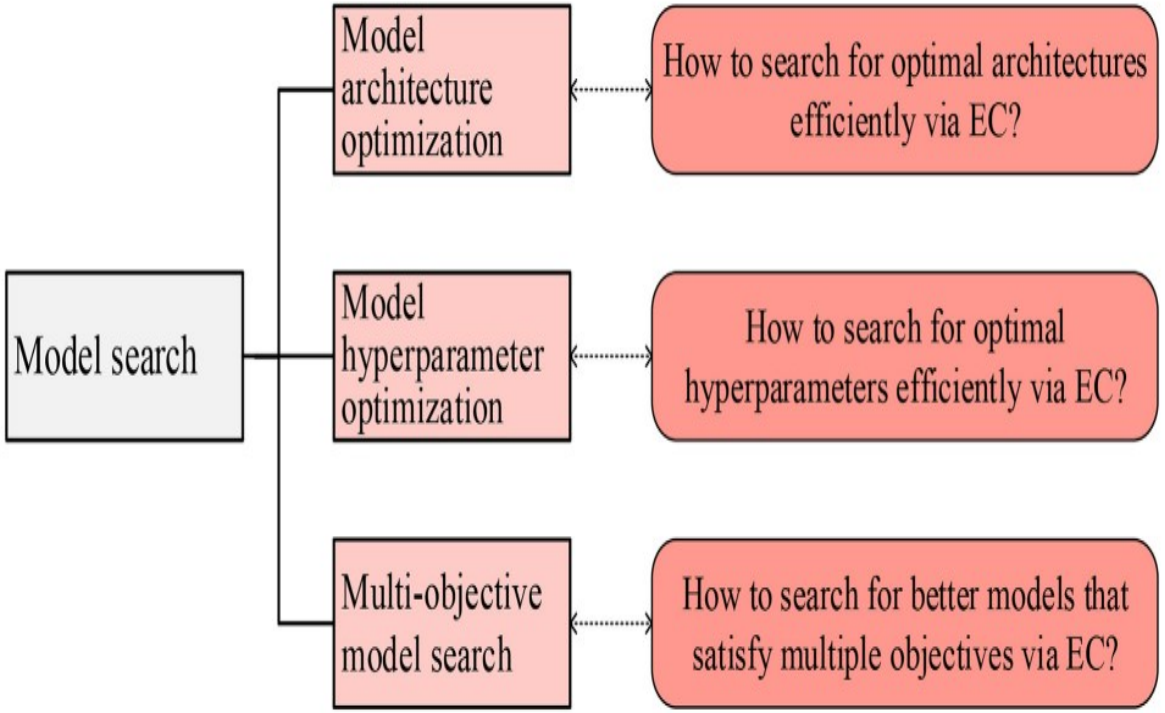


Figure: Categories and corresponding issues of EDL in the model search procedure.

Evolutionary Deep Learning

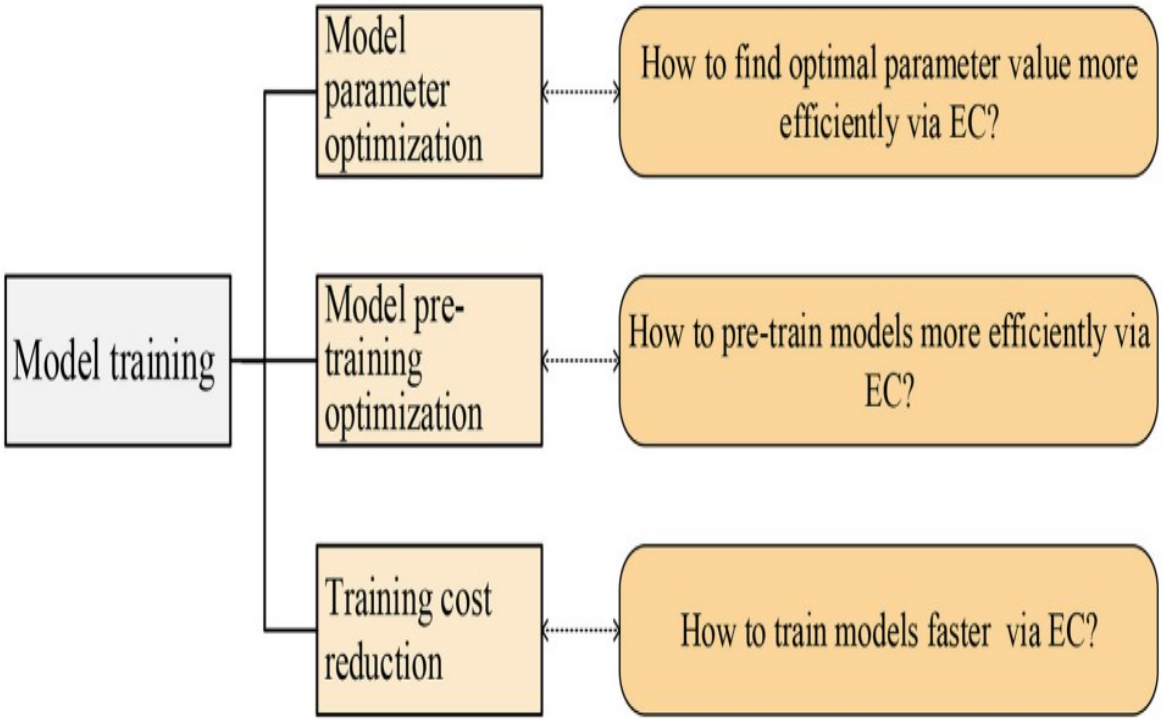


Figure: Categories and corresponding issues of EDL in the model training procedure.

Evolutionary Deep Learning

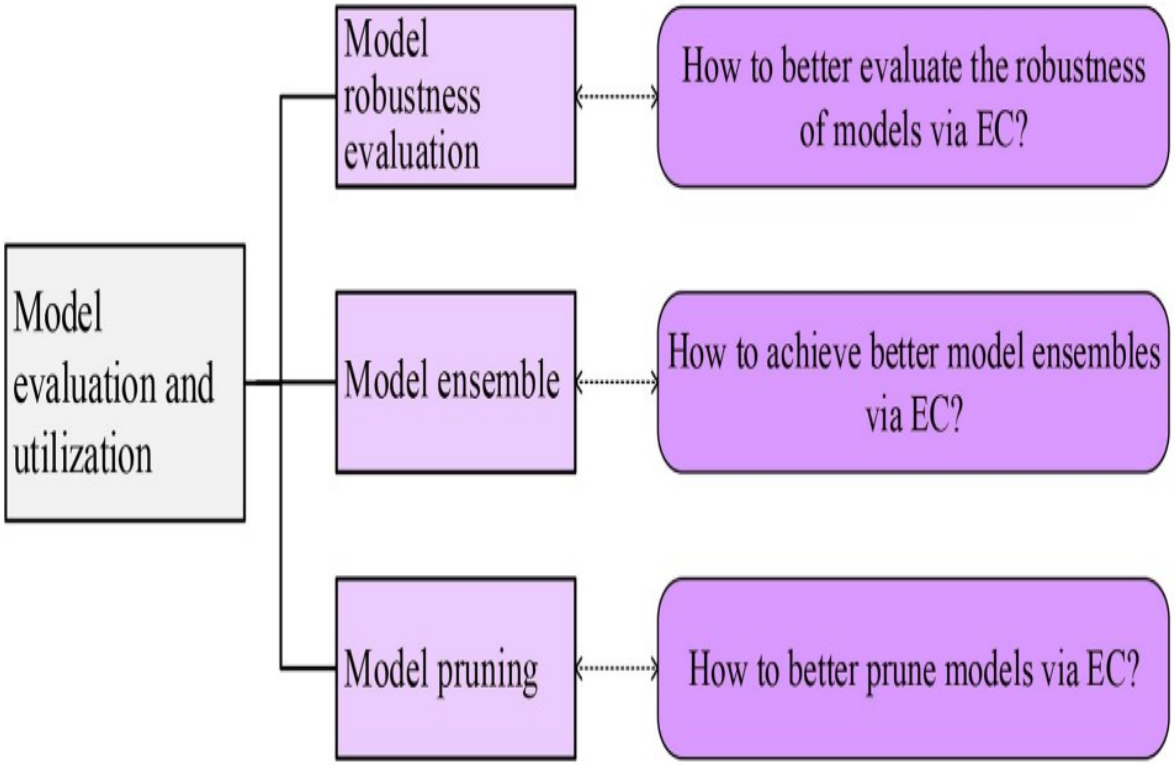


Figure: Categories and corresponding issues of using EDLs in the model evaluation and utilization procedure.

Evolutionary Deep Learning

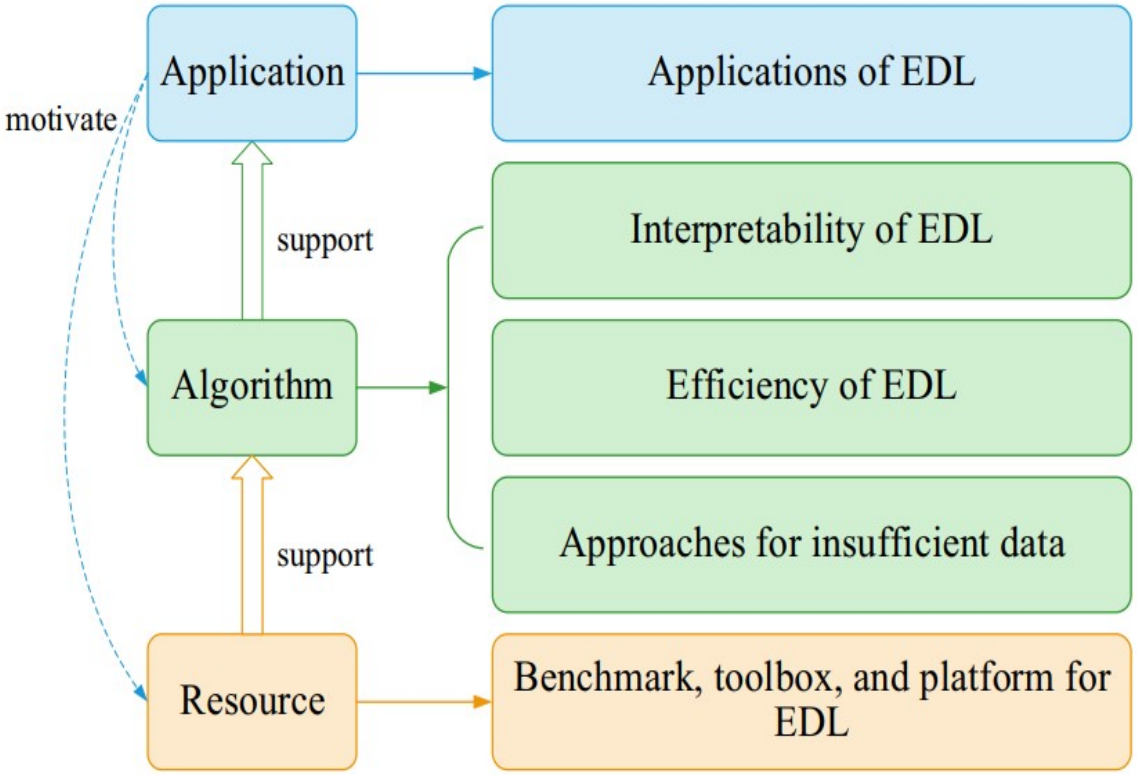


Figure: Some potential future research directions of EDL.

Evolutionary Deep Learning

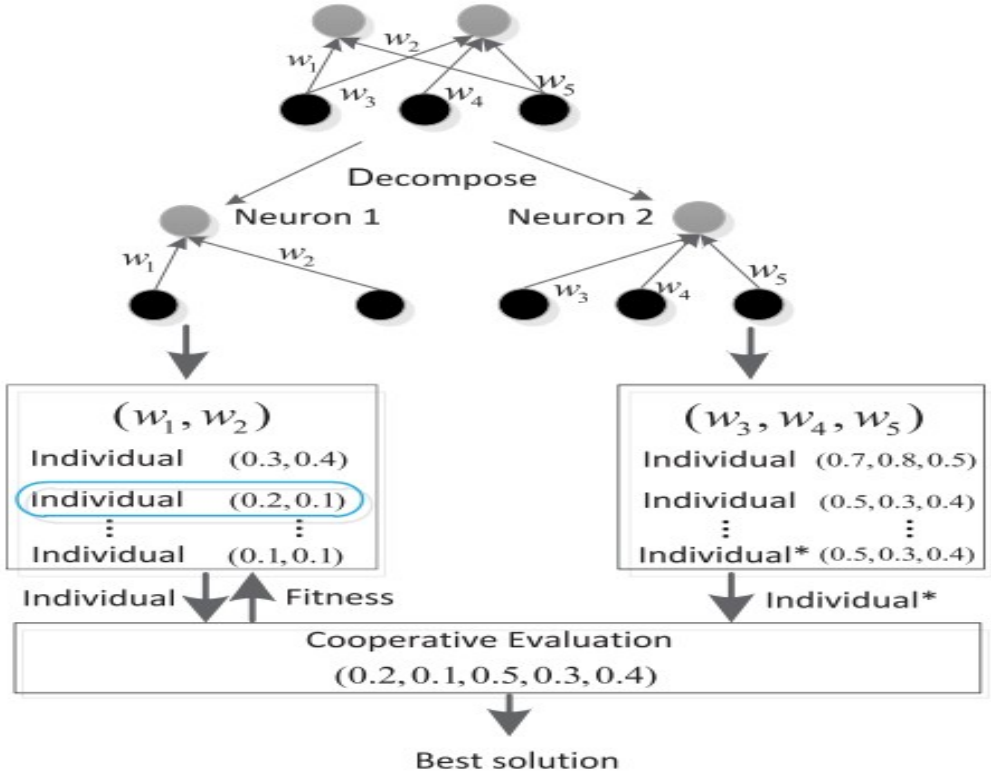


Figure: Neuron-based cooperative co-evolution, where the connection weights for neuron 1 and neuron 2 are partitioned into two separate groups (w1 and w2) and (w3, w4, and w5).

Presentation Outline

1. Machine Learning
2. Evolutionary Deep Learning
3. Pruning and Quantization for Deep Neural Network
4. Conclusion
5. References

Pruning and Quantization for Deep Neural Network

- Deep neural networks have been applied in many applications exhibiting extraordinary abilities in the field of computer vision.
- However, complex network architectures challenge efficient real-time deployment and require significant computation resources and energy costs.
- These challenges can be overcome through optimizations such as network compression.
- Two types of network compression: Pruning and Quantization.
- Pruning can be categorized as static if it is performed offline or dynamic if it is performed at run-time.
- The trade-offs in element-wise, channel-wise, shape-wise, filter-wise, layer-wise and even network-wise is trending pruning.
- Quantization reduces computations by reducing the precision of the datatype. Weights, biases, and activations may be quantized typically to 8-bit integers

Over-parameterization in Deep Neural Network

- Over-parameterization is the property of a neural network where redundant neurons do not improve the accuracy of results.
- However, This redundancy can often be removed with little or no accuracy loss.
- The three design considerations that may contribute to over-parameterization:
- Network structure,
- Network optimization, and
- Hardware accelerator design

Over-parameterization in Deep Neural Network

- Network structure encompasses three parts:
- Network components,
- Network architecture search, and
- Knowledge distillation

Network Components

- Network components is the design of efficient blocks such as separable convolution, inception blocks, and residual blocks.
- Network components also encompasses the types of connections within layers.
- Fully connected deep neural networks require N*N connections between neurons.
- Feed forward layers reduce connections by considering only connections in the forward path.This reduces the number of connections to N.
- Other types of components such as dropout layers can reduce the number of connections even further.

Network Architecture Search (NAS)

- Network Architecture Search (NAS), also known as network auto search, programmatically searches for a highly efficient network structure from a large predefined search space. An estimator is applied to each produced architecture. While time-consuming to compute, the final architecture often outperforms manually designed networks

Knowledge Distillation (KD)

- KD evolved from knowledge transfer. The goal is to generate a simpler compressed model that functions as well as a larger model.

Network Optimization

- Network optimization includes:
  1) computational convolution optimization,
  2) parameter factorization,
  3) network pruning, and
  4) network quantization.
- Convolution operations are more efficient than fully connected computations because they keep high dimensional information as a 3D tensor rather than flattening the tensors into vectors that removes the original spatial information.
- This feature helps CNNs to fit the underlying structure of image data in particular. Convolution layers also require significantly less coefficients compared to Fully Connected Layers.

Parameter factorization

- Parameter factorization is a technique that decomposes higher rank tensors into lower-rank tensors simplifying memory access and compressing model size.
- It works by breaking large layers into many smaller ones, thereby reducing the number of computations.
- It can be applied to both convolutional and fully connected layers.
- This technique can also be applied with pruning and quantization

Network Pruning

- Network pruning involves removing parameters that don't impact network accuracy.

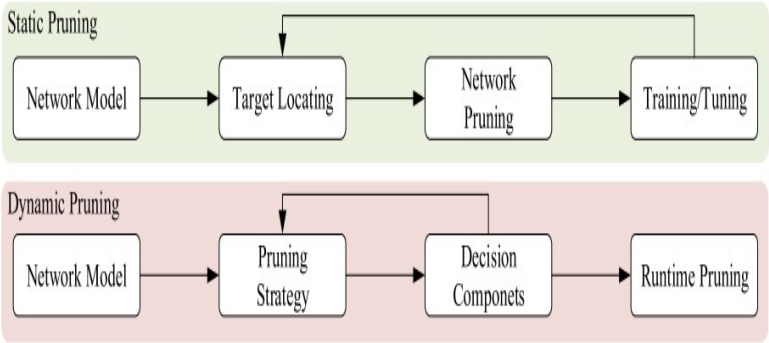


Figure: Pruning Categories: Static pruning is performed offline prior to inference while Dynamic pruning is performed at runtime:Tailin Liang et.al

Network Pruning and Network Quantization

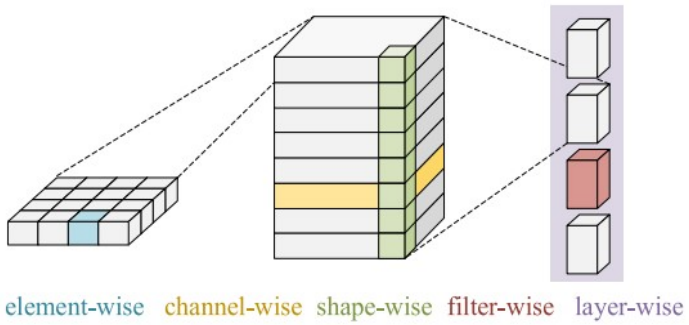


Figure: Pruning Opportunities: Different network sparsity results from the granularity of pruned structures. Shape-wise pruning was proposed by Wen et.al

- Quantization is known as the process of approximating a continuous signal by a set of discrete symbols or integer values.
- Network quantization involves replacing datatypes with reduced width datatypes.
- For example, replacing 32-bit Floating Point (FP32) with 8-bit Integers (INT8)

Hardware accelerator design

- Hardware accelerators are designed primarily for network acceleration.
- CPUs have been optimized with specialized Artificial Intelligence (AI) instructions usually within specialized Single Instruction Multiple Data (SIMD) units.
- GPUs have been used for both training and inference. nVidia has specialized tensor units incorporated into their GPUs that are optimized for neural network acceleration.
- Specialized ASICs,FPGAs have also been designed for neural network acceleration.

Presentation Outline

1. Machine Learning
2. Evolutionary Deep Learning
3. Pruning and Quantization for Deep Neural Network
4. Conclusion
5. References

Conclusion

- The increasing availability of unstructured data allows artificial intelligence applications to provide better insight into data and thus give companies a competitive advantage.
- Machine learning (ML) is rapidly changing the world, from diverse types of applications and research pursued in industry and academia.
- It is evident from the studies that, automated construction of Deep Neural Networks (DNNs) has become a research hot spot nowadays because DNN's performance is heavily influenced by its architecture and parameters, which are highly task-dependent, but it is notoriously difficult to find the most appropriate DNN in terms of architecture and parameters to best solve a given task.

Presentation Outline

1. Machine Learning
2. Evolutionary Deep Learning
3. Pruning and Quantization for Deep Neural Network
4. Conclusion
5. References

References I

Thank You!

Perceptual Video Processing

by
Dinesh Naik
(B.E, M.Tech, Ph.D)

Assistant Professor, Dept. of Information Technology
National Institute of Technology Karnataka, Surathkal

February, 2023

Efficient Convolutional Neural Networks

- The efficient neural-network design approach either searches for an efficient base model from a predefined search pool or modifies the given base model to obtain the final compressed model.
- Network pruning, data quantization, tensor decomposition, and knowledge distillation approaches use the given base model, whereas more recent approaches, such as NAS, programmatically search for highly efficient neural network structures.

CNN Acceleration Approaches

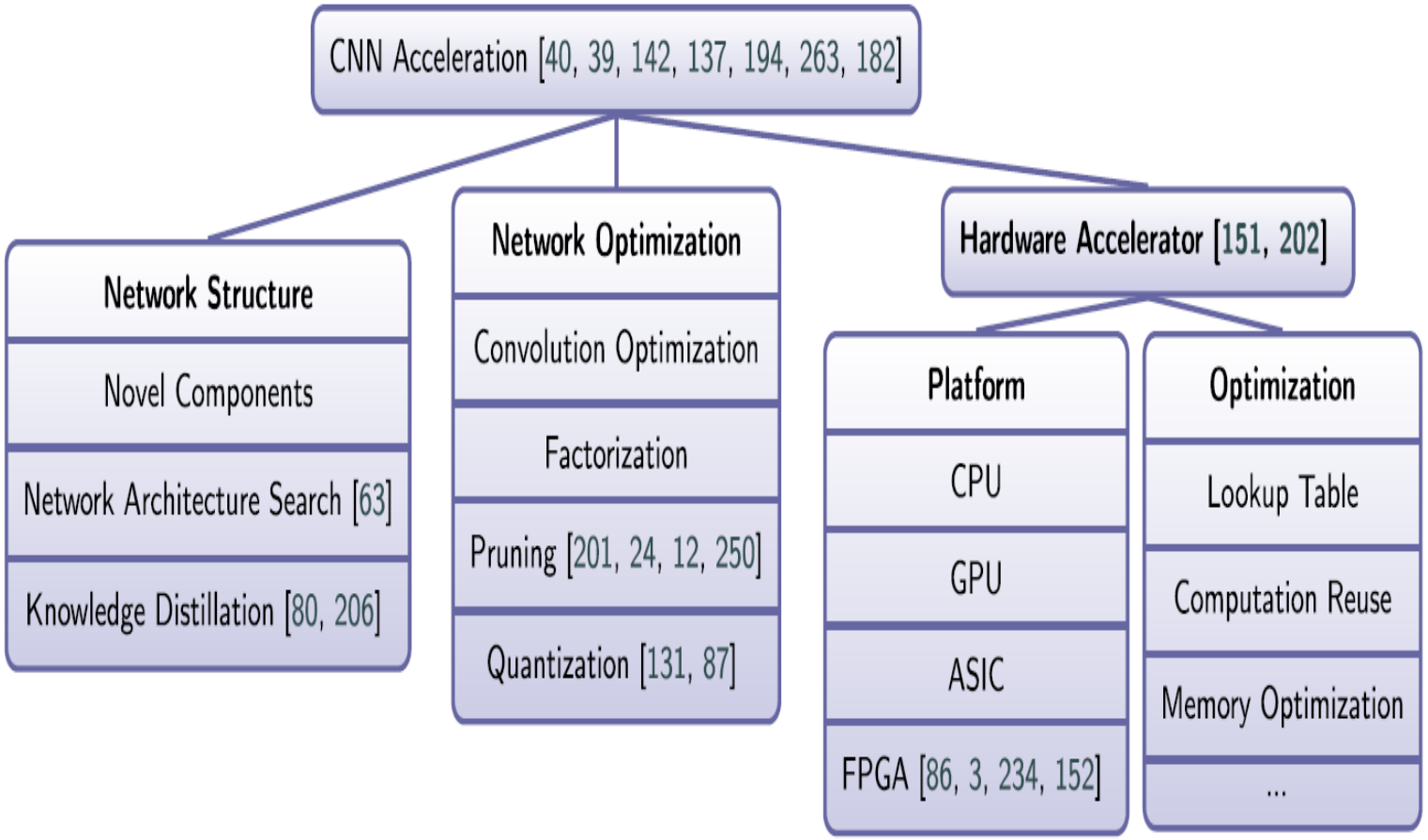


Figure: CNN Acceleration Approaches

Network Structure

- Novel components also encompasses the types of connections within layers. Fully connected deep neural networks require N2 connections between neurons.
- Feed forward layers reduce connections by considering only connections in the forward path. This reduces the number of connections to N.
- Other types of components such as dropout layers can reduce the number of connections even further.

Network Structure

- Network Architecture Search (NAS) [63], also known as network auto search, programmatically searches for a highly efficient network structure from a large predefined search space.

- An estimator is applied to each produced architecture. While time-consuming to compute, the final architecture often outperforms manually designed networks.

Network Structure

- Knowledge Distillation (KD) evolved from knowledge transfer [27]. The goal is to generate a simpler compressed model that functions as well as a larger model.
- KD trains a student network that tries to imitate a teacher network.
- The student network is usually but not always smaller and shallower than the teacher.
- The trained student model should be less computationally complex than the teacher.

Network Optimization

- Network optimization includes:
  1) computational convolution optimization,
  2) parameter factorization,
  3) network pruning, and
  4) network quantization.
- Convolution operations are more efficient than fully connected computations because they keep high dimensional information as a 3D tensor rather than flattening the tensors into vectors that removes the original spatial information.
- This feature helps CNNs to fit the underlying structure of image data in particular.
- Convolution layers also require significantly less coefficients compared to Fully Connected Layers.
- Computational convolution optimizations include Fast Fourier Transform (FFT) based convolution [168], Winograd convolution [135], and the popular image to column (im2col) [34] approach.

CNN Convolution Operations

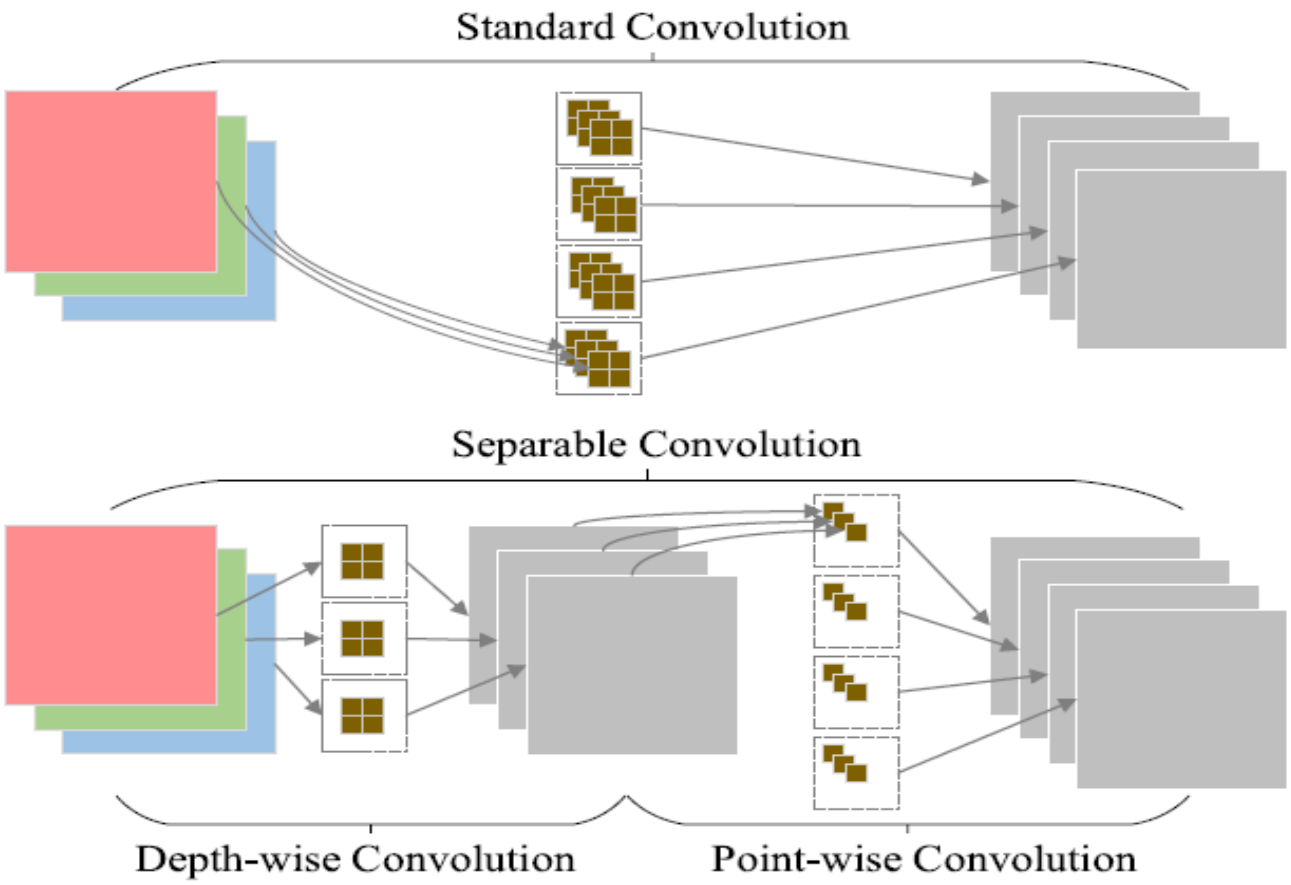


Figure: Separable Convolution: A standard convolution is decomposed into depthwise convolution and point-wise convolution to reduce both the model size and computations.

CNN Convolution Operations

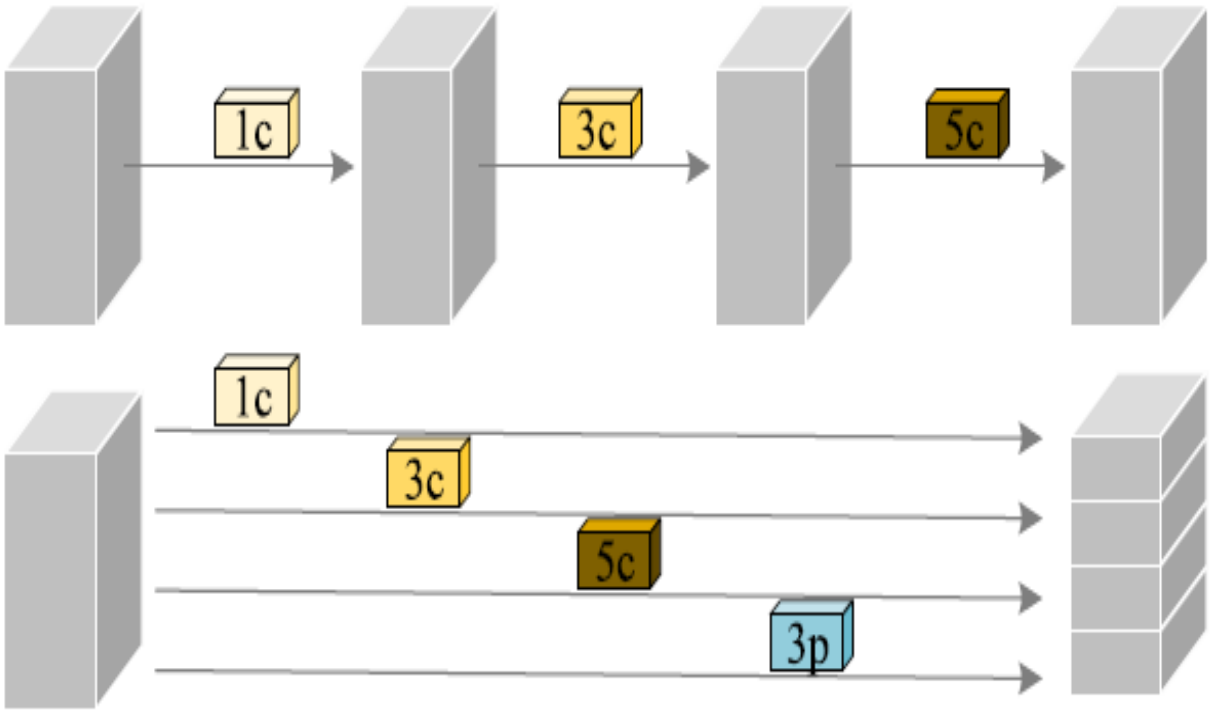


Figure: Inception Block: The inception block computes multiple convolutions with one input tensor in parallel, which extends the receptive field by mixing the size of kernels. The yellow - brown coloured cubes are convolutional kernels sized 1, 3, and 5. The blue cube corresponds to a 3 x 3 pooling operation.

CNN Convolution Operations

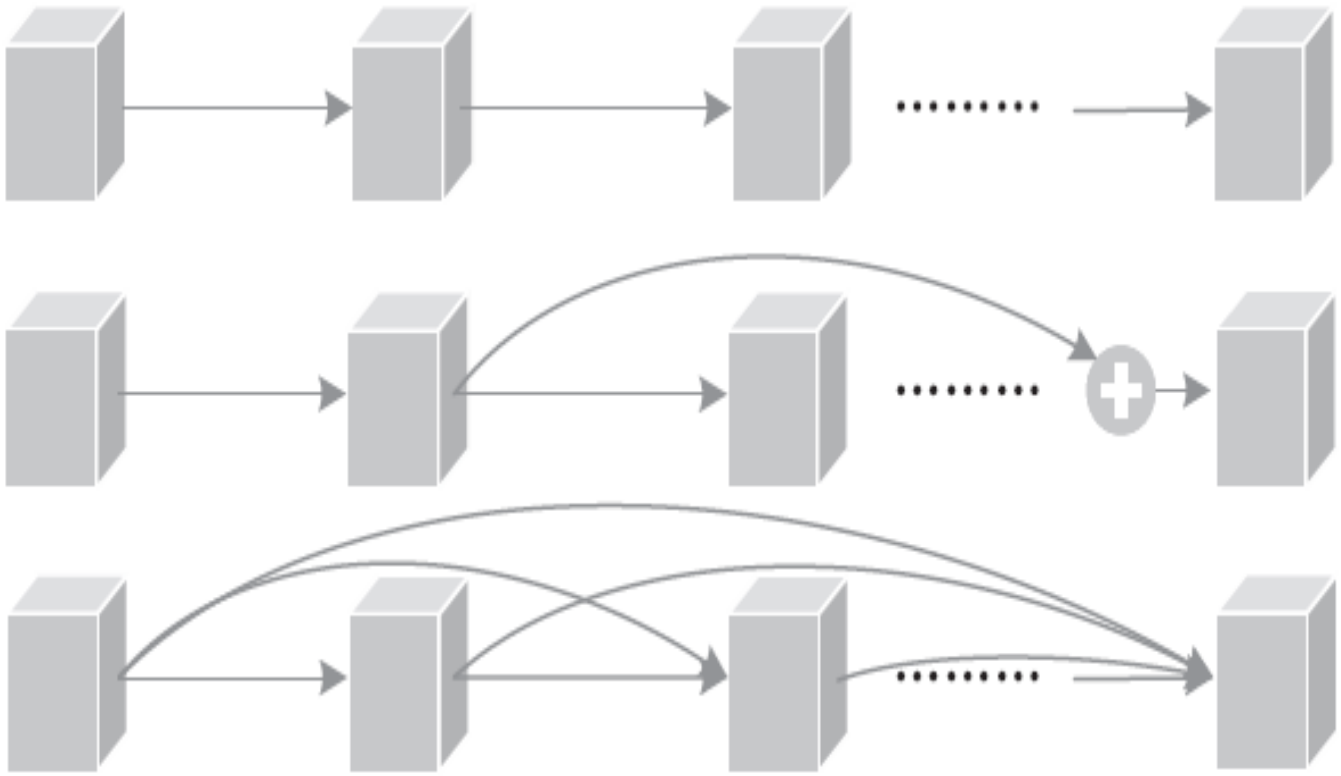


Figure: Conventional Network Block (top), Residual Network Block (middle), and Densely Connected Network Block (bottom).

Parameter factorization

- Parameter factorization is a technique that decomposes higher rank tensors into lower-rank tensors simplifying memory access and compressing model size.
- It works by breaking large layers into many smaller ones, thereby reducing the number of computations.
- It can be applied to both convolutional and fully connected layers.
- This technique can also be applied with pruning and quantization

Network Pruning

- Network pruning involves removing parameters that don't impact network accuracy.

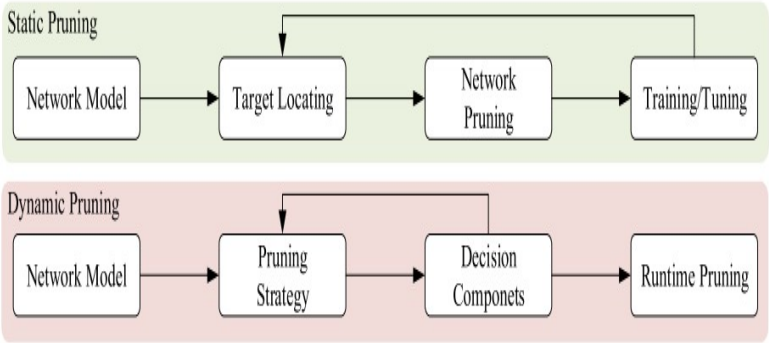


Figure: Pruning Categories: Static pruning is performed offline prior to inference while Dynamic pruning is performed at runtime:Tailin Liang et.al

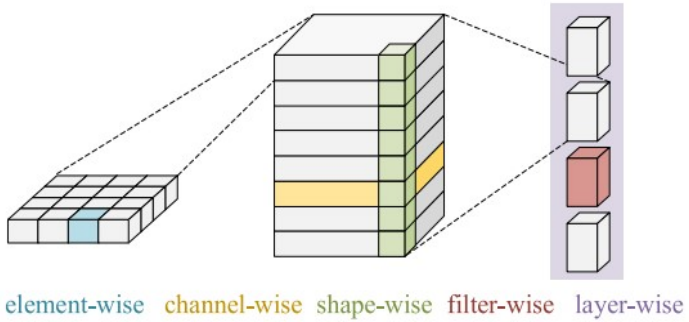


Figure: Pruning Opportunities: Different network sparsity results from the granularity of pruned structures. Shape-wise pruning was proposed by Wen et.al

- Quantization is known as the process of approximating a continuous signal by a set of discrete symbols or integer values.
- Network quantization involves replacing datatypes with reduced width datatypes.
- For example, replacing 32-bit Floating Point (FP32) with 8-bit Integers (INT8)

Pruning

- Network pruning is a well-known topic that can be traced back to the early 1990s.
- The key idea in pruning is to remove unimportant weights, filters, channels, or even layers from the original DNN, resulting in a reduced number of memory access and computation operations.
- Weight or connection pruning attempts to eliminate unimportant connections from the original model, whereas more recent techniques use large-scale pruning, such as filter-pruning or channel-pruning.
- In general, network pruning can be categorized as (1) connection (weight) pruning, also known as unstructured pruning, and (2) filter or channel pruning, also known as structural pruning.

Pruning Flow

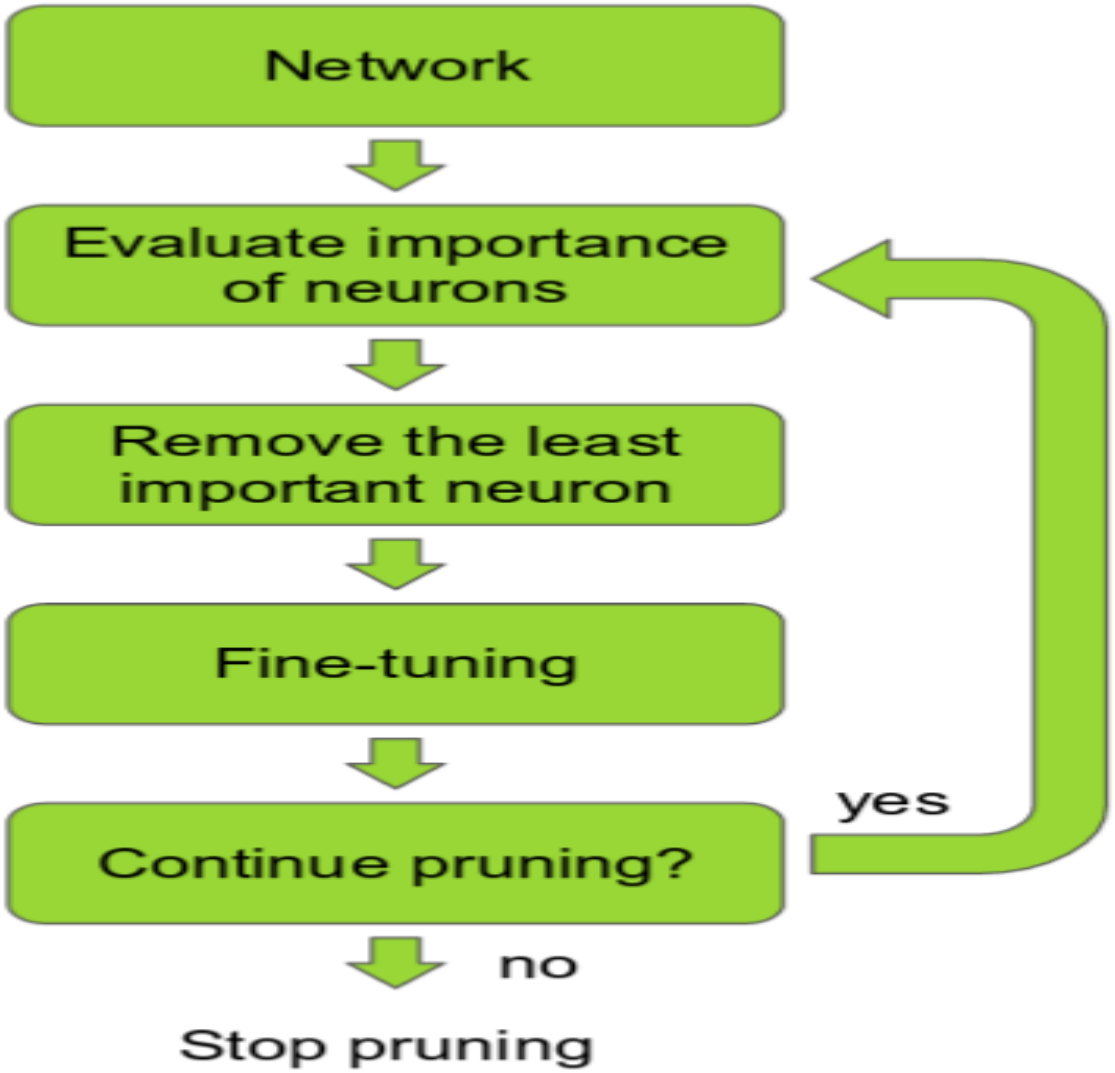


Figure: The image is taken from [1611.06440 Pruning Convolutional Neural Networks for Resource Efficient Inference]

Weight Pruning

- Some early works, i.e., Optimal Brain Damage (OBD) and Optimal Brain Surgeon (OBS), use the second derivative (Hessian matrix) of the loss function to prune each non-essential weight.
- The most popular early works in the deep learning era used a three-step method to prune the weights below the threshold and demonstrated that many connections can be removed from the DNN to achieve a significant compression ratio.
- First, the network is trained to find important connections, then, unimportant connections are pruned, and, finally, the network is retrained to fine-tune the remaining connections.
- Iterative connection pruning and fine-tuning gradually produce a compressed network.
- Dead neurons, for which all connected weights are pruned, can also be safely dropped.

Structural Pruning

- In this category, the methods aim to reduce the memory footprint by pruning entire filters, channels, or even layers.
- After pruning, the input and output shapes of the layers and weight matrices are changed, still permitting dense matrix operations without requiring any specialized hardware or software.
- Identifying the least important filter and removing it is the most popular scheme in filter pruning.
- Ranking the filters with certain criteria, minimizing the reconstruction errors, and finding the replaceable filters with similarity measurements are the three main branches under this category.

Structural Pruning- ThiNet-ThinNet

- Proposed a unified framework, namely ThiNet (stands for "Thin Net"), to prune the unimportant filters to simultaneously accelerate and compress CNN models in both training and test stages with minor performance degradation.

- With their pruned network, some important transfer tasks such as object detection or fine-grained recognition can run much faster (both training and inference), especially in small devices.

- A well-defined optimization problem, which shows that whether a filter can be pruned depends on the outputs of its next layer, not its own layer. This novel finding differentiates ThiNet from existing methods which prune filters using statistics calculated from their own layer.

Framework of ThiNet

- **Filter selection**: Unlike existing methods that use layer i's statistics to guide the pruning of layer i's filters, in this method layer i + 1 to guide the pruning in layer i.
- The key idea is: if we can use a subset of channels in layer (i + 1)'s input to approximate the output in layer i + 1,the other channels can be safely removed from the input of layer i + 1.
- Note that one channel in layer (i + 1)'s input is produced by one filter in layer i, hence we can safely prune the corresponding filter in layer i.
- **Pruning.** Weak channels in layer (i + 1)'s input and their corresponding filters in layer i would be pruned away, leading to a much smaller model. Note that, the pruned network has exactly the same structure but with fewer filters and channels. In other words, the original wide network is becoming much thinner. That is why we call this method "ThiNet".

Framework of ThiNet

- **Fine-tuning.** Fine-tuning is a necessary step to recover the generalization ability damaged by filter pruning.But it will take very long for large datasets and complex models.

- For time-saving considerations, we fine-tune one or two epochs after the pruning of one layer. In order to get an accurate model, more additional epochs would be carried out when all layers have been pruned.

- **Iterate to step 1 to prune the next layer**.

Structural Pruning

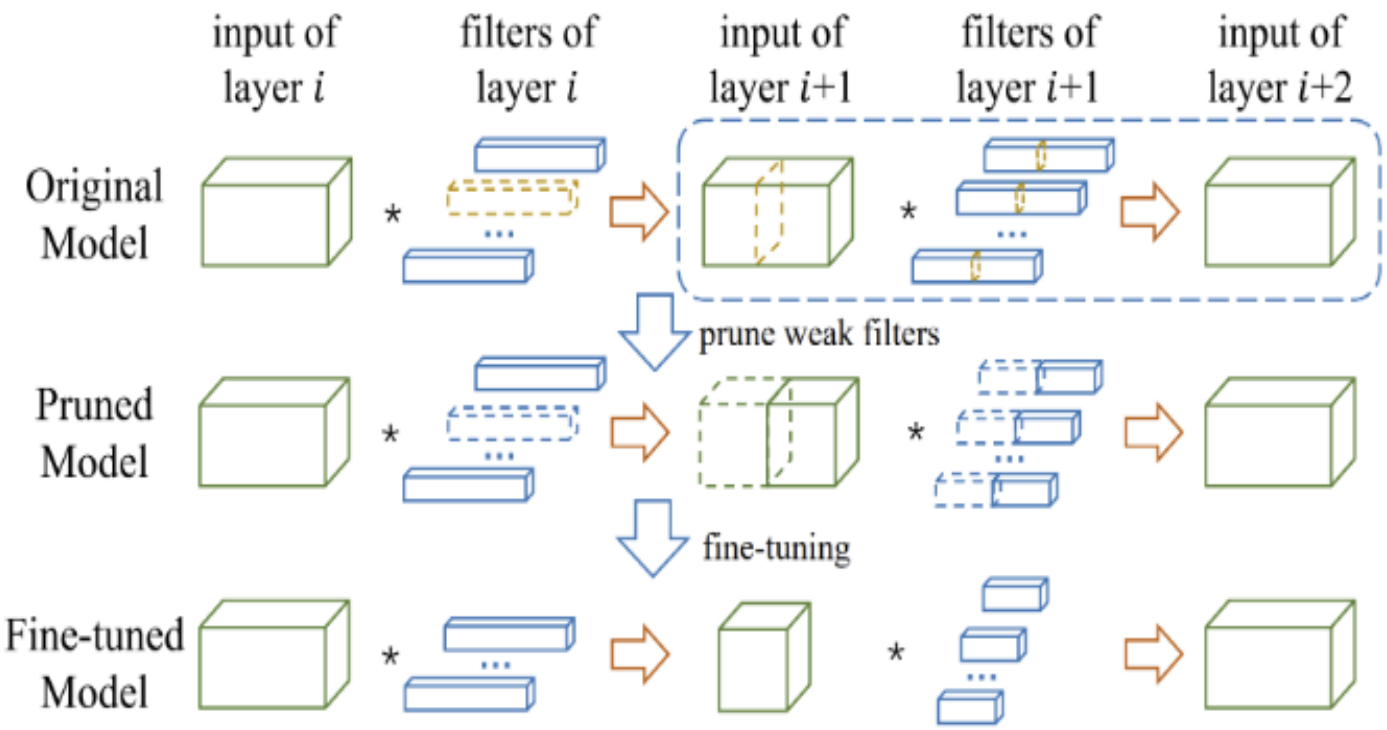


Figure: Example of Structural Pruning (ThiNet: statistics of layer i + 1 are used to guide the pruning in layer i).

Structural Pruning- Benefits

- The benefits of removing the whole unimportant filter have a great deal:
- The pruned model has no difference in network structure, thus it can be perfectly supported by any off-the-shelf deep learning libraries.
- Memory footprint would be reduced dramatically. Such memory reduction comes not only from model parameter itself, but also from the intermediate activation, which is rarely considered in previous studies.
- Since the pruned network structure has not be damaged, it can be further compressed and accelerated by other compression methods, e.g.,the parameter quantization approach.
- More vision tasks, such as object detection or semantic segmentation, can be accelerated greatly using the pruned model.

Quantization

- Quantization reduces computations by reducing the precision of the data type, instead of reducing the number of computations in the network.
- This minimizes the bitwidth of the data storage and flow through the deep neural network. The computation and storage of data at a lower bitwidth allows for speedy inference with reduced energy consumption.
- During quantization, the neural network data are restricted to a set of discrete levels that can have different distributions: uniform or non-uniform.
- A uniform distribution with even steps is a widely used quantization scheme. A logarithmic distribution is a commonly used non-uniform quantization method.
- Both deterministic and stochastic methods were used to project high-precision data into a discrete space. The former projects data to the nearest discrete distance level, whereas, in the latter, the projection to one of the two adjacent discrete levels is determined by the probability.

Quantization

- Quantizing neural networks from FP32 precision to low bitwidth is a popular compression technique that can be traced back to the early 1990s.
- There are two forms of quantization: post-training quantization and quantization-aware training.
- As the name suggests, post-training quantization uses the FP32 weights and activations to apply quantization after the model is fully trained.
- In quantization-aware training, the quantization error is considered as part of the training loss when training the model.
- Although post-training quantization is easy to use, quantization-aware training generally improves model's accuracy.

Binarization

- The most extreme form of quantization is binarization.
- In binarization, the data can have only two possible values 0, 1 or 1, 1. With binarization, heavy matrix multiplications can be replaced with simple XNOR and bitcount operations.
- Due to the fact that it saves significant storage and computation, binarization is a promising technique for deploying deep neural networks in resource-constrained devices.
- However, owing to extreme quantization, binarization inevitably suffers from heavy information loss. Simultaneously, owing to its discontinuity, optimizing a binary neural network becomes difficult.

Hardware accelerator design

- Hardware accelerators are designed primarily for network acceleration.
- CPUs have been optimized with specialized Artificial Intelligence (AI) instructions usually within specialized Single Instruction Multiple Data (SIMD) units.
- GPUs have been used for both training and inference. nVidia has specialized tensor units incorporated into their GPUs that are optimized for neural network acceleration.
- Specialized ASICs,FPGAs have also been designed for neural network acceleration.

Hardware accelerator design

- Specialized ASICs have also been designed for neural network acceleration. They typically target inference at the edge, in security cameras, or on mobile devices. Examples include: General ProcessorTechnologies (GPT) [179], ARM, nVidia.
- ASICs may also target both training and inference in datacenters.
- Tensor processing units (TPU) from Google [125], Habana from Intel [169], Kunlun from Baidu [191], Hanguang from Alibaba [124], and Intelligence Processing Unit (IPU) from Graphcore [121].

Evolutionary Computation

- EC is also an important branch in AI that is efficient in optimizing complex optimization problems.
- In general, EC imitates the evolution rule of "survival of the fittest" from nature to evolve candidate solutions, so as to obtain more satisfactory solutions for optimization problems.
- EC algorithms mainly include evolutionary algorithm (EA) and swarm intelligence (SI) algorithms.
- To be more specific, representative EAs include genetic algorithm (GA), evolutionary strategy (ES) , genetic programming (GP), evolutionary programming (EP), differential evolution (DE), and estimation of distribution algorithm (EDA), while representative SI algorithms include ant colony optimization (ACO) and particle swarm optimization (PSO).

Table 1
Comparisons of Different Surveys about EC for DL.

Survey	Perspective and focus	Taxonomy and category	Year
Liu et al.' survey [44]	EC for neural architecture search	Encoding space/encoding strategy/population updating/efficient evaluation/application	2021
Zhou et al.' survey [45]	EC for the construction of DNNs	Optimization on architecture/parameter/both architecture and parameter/miscellaneous issues like different tasks and objectives	2021
Zhou et al.' survey [46]	advances of evolutionary neural architecture search	shallow and deep networks/and innovation/booming/trends of evolutionary DNNs	2021
Darwish et al.' survey [47]	EC for both architecture search and hyperparameter optimization	Evolutionary algorithm/swarm intelligence for DNN/CNN/DBN/RNN	2020
Bharti et al.' survey [48]	EC for both architecture search and hyperparameter optimization	EC for LSTM/CNN/GAN/AE/ESN	2020
This survey	EC in the whole DL procedure	EC for data process/model search/model training/model evaluation and utilization in the DL procedure	–

Figure: Comparisons of Different Surveys about EC for DL.

Deep learning model

- In general, widely-used DL models can be roughly divided into five categories according to their basic model, i.e., convolutional neural network (CNN), deep belief network (DBN), stacked auto-encoder (SAE), recurrent neural network (RNN) , and generative adversarial network (GAN).
- CNN is a widely-used DNN structure consisting of convolutional layers, pooling layers, and fully connected layers.
- DBN can be developed by stacking restricted Boltzmann machines (RBMs). In general, DBNs can pre-train the network in an unsupervised fashion and then fine-tunes it in a supervised way according to the labeled data, so as to obtain a more satisfactory features representation.
- SAE is composed of multiple auto-encoders (AEs) in a stacking manner. Generally speaking, an AE consists of two symmetrical components (i.e., encoder and decoder).

Deep learning model(contd.

- RNN is a special neural network that contains recurrent connections.
- GAN is a deep learning model that contains two components, i.e., generator and discriminator.
- The generator learns how to generate data similar to real training data to fool the discriminator, while the discriminator learns how to figure out the data from a real training set or generated by the generator.
- After training, the generator will have a great probability for data generation, which is meaningful in many real-world applications.

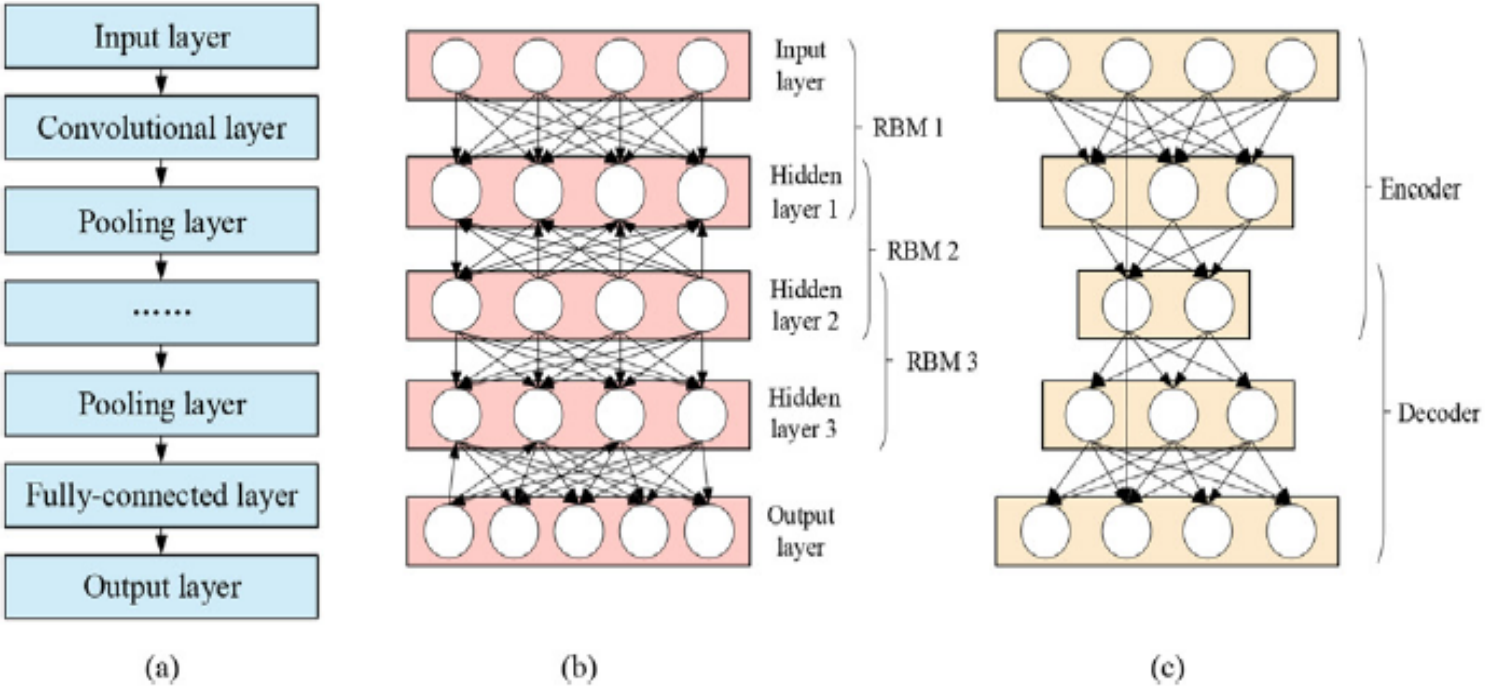


Figure: Examples of some different deep learning models: (a) CNN; (b) DBN; (c) AE.

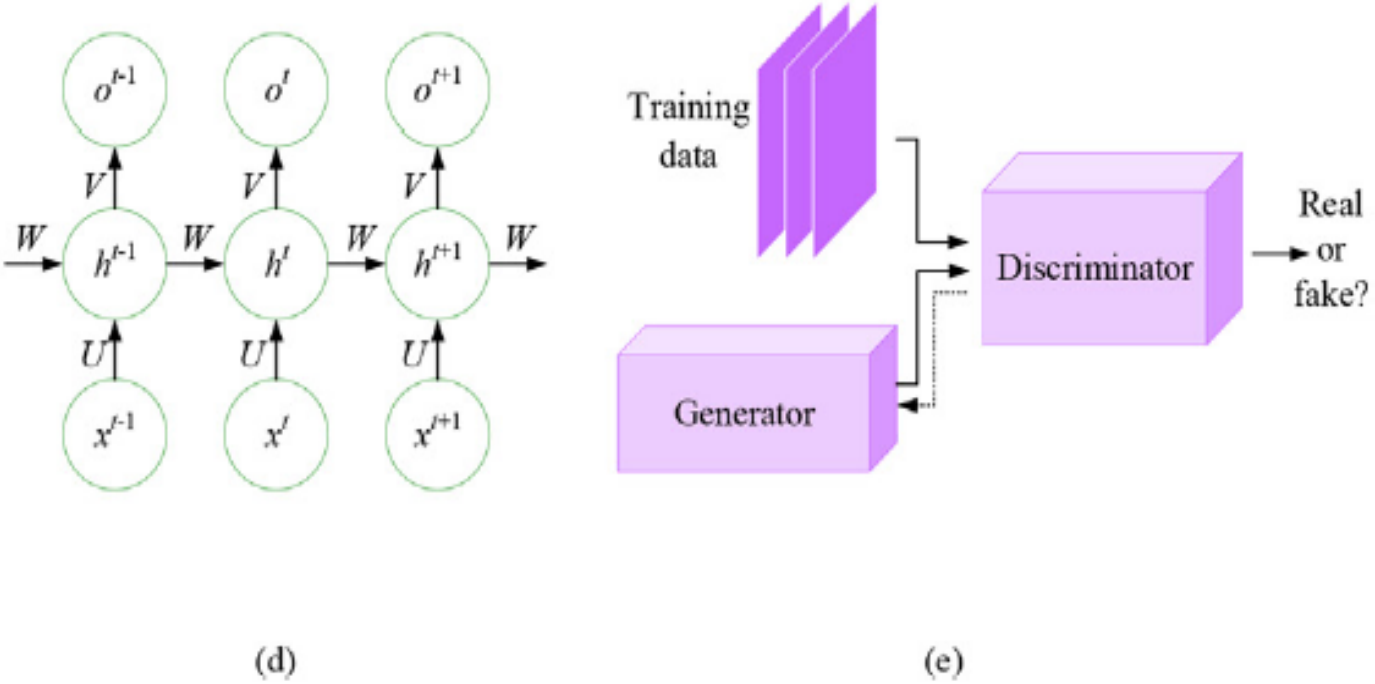


Figure: Examples of some different deep learning models:(d) RNN; and (e) GAN.

Presentation Outline

1. Evolutionary Deep Learning
2. Pruning and Quantization for Deep Neural Network
3. Conclusion
4. References

Evolutionary Deep Learning

- The performance of DL can be influenced by many factors such as the architecture and hyperparameters, which involves data quality, model configuration and robustness, learning techniques, and problem characteristics.

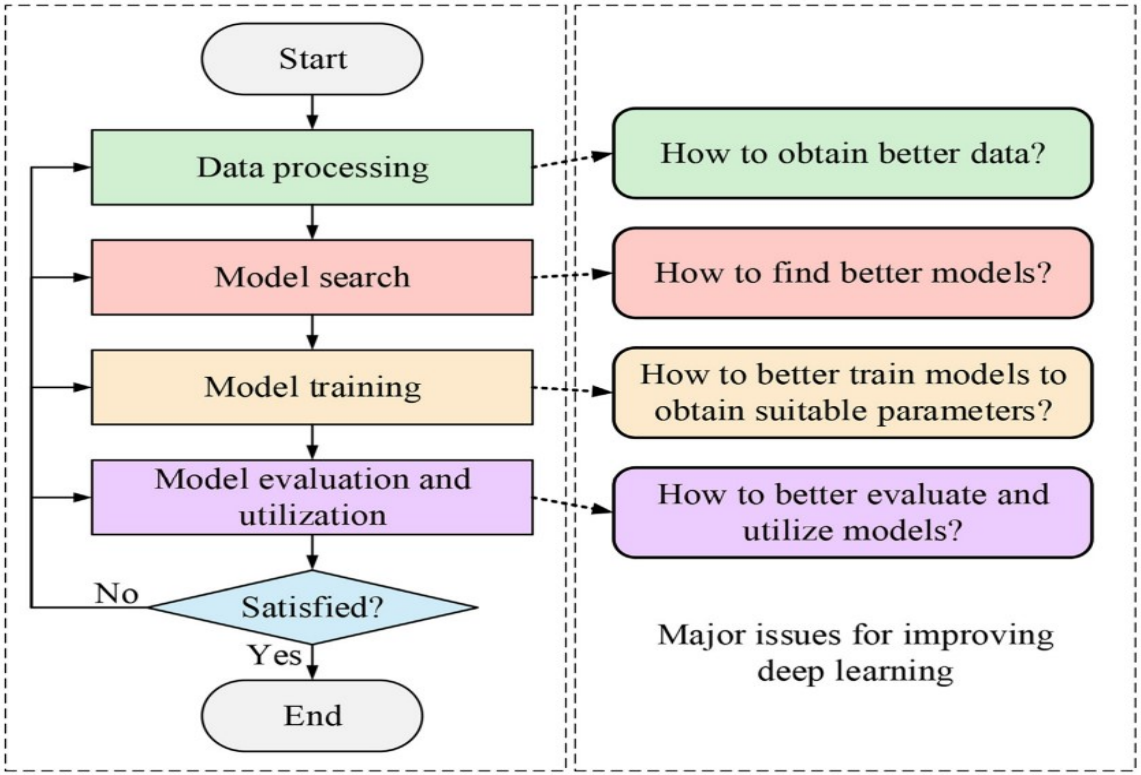


Figure: The flowchart of the whole DL lifetime with major issues for improving DL in different procedures

Evolutionary Deep Learning

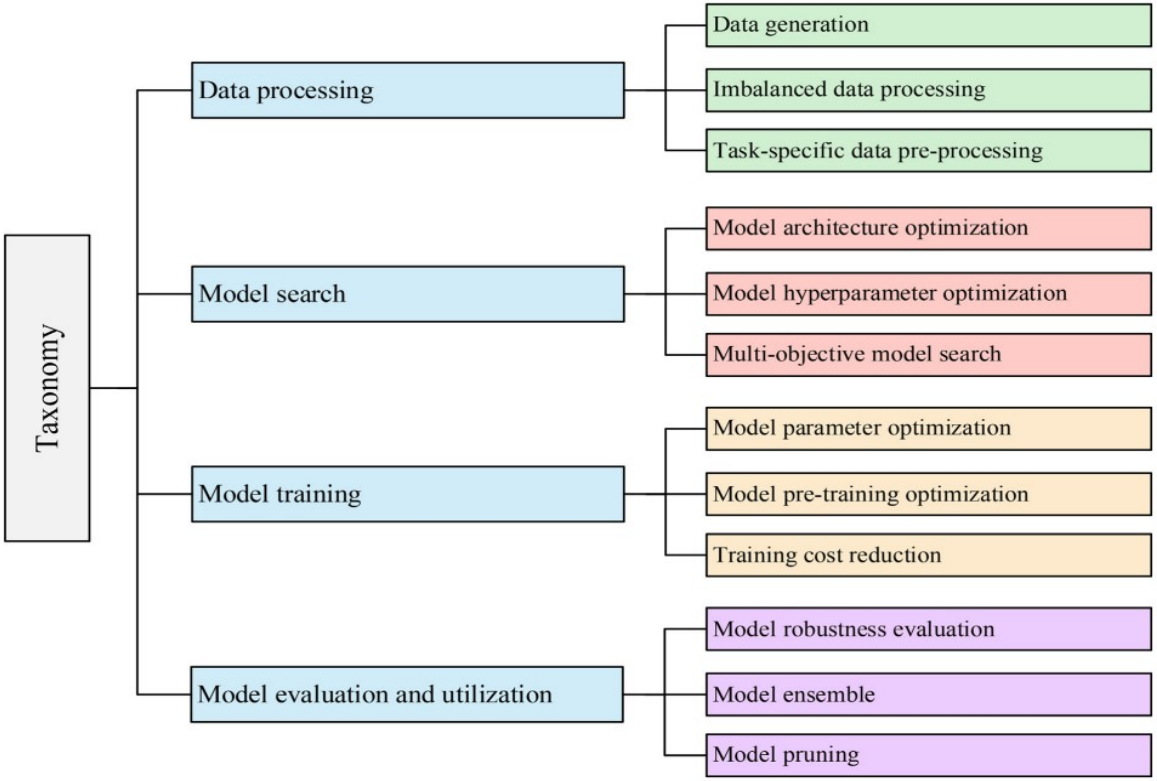


Figure: An illustration of the taxonomy.

Evolutionary Deep Learning

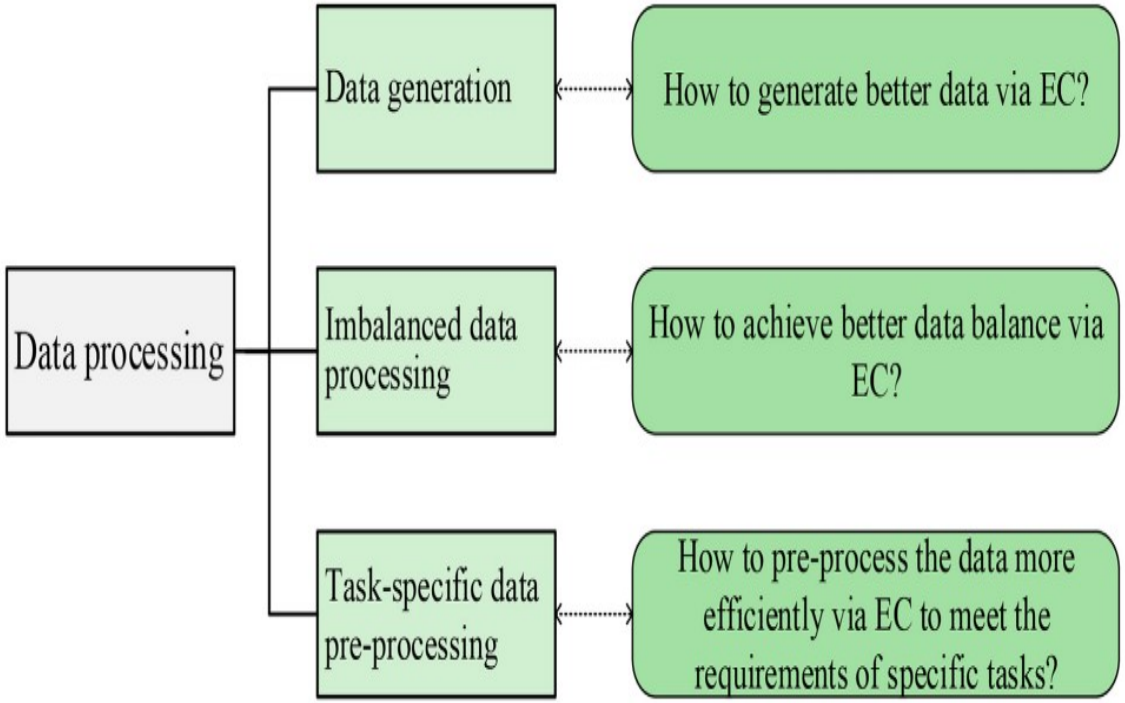


Figure: Categories and corresponding issues of EDL in the data processing procedure.

Evolutionary Deep Learning

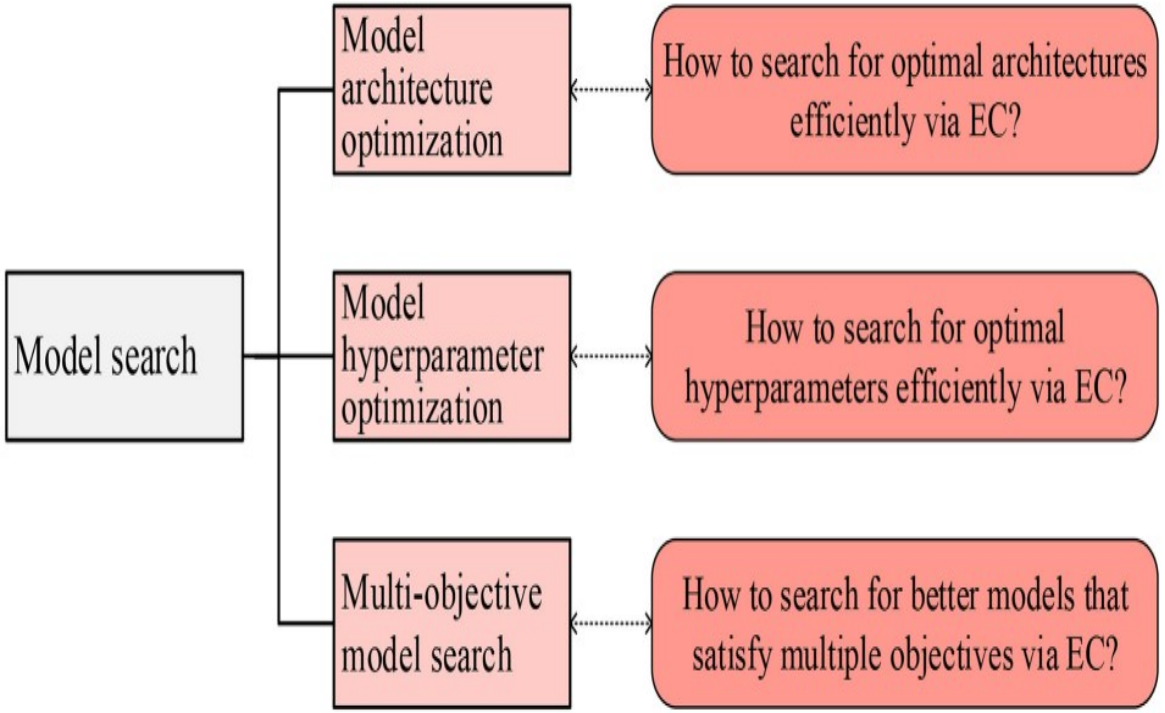


Figure: Categories and corresponding issues of EDL in the model search procedure.

Evolutionary Deep Learning

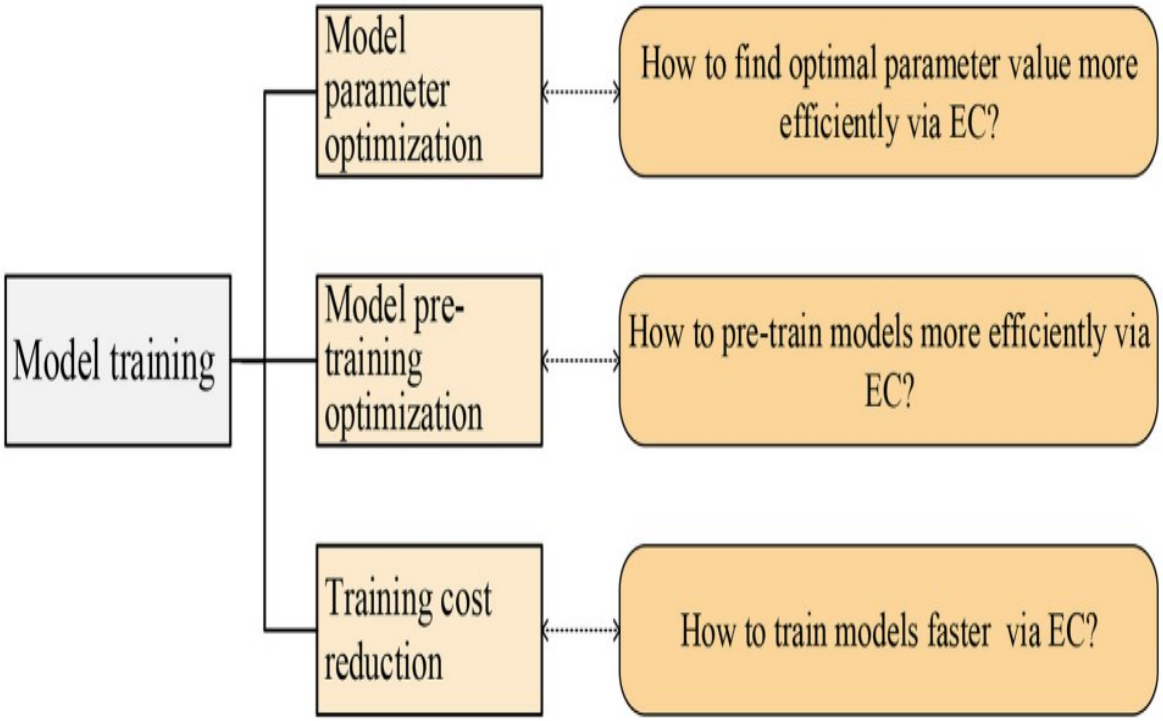


Figure: Categories and corresponding issues of EDL in the model training procedure.

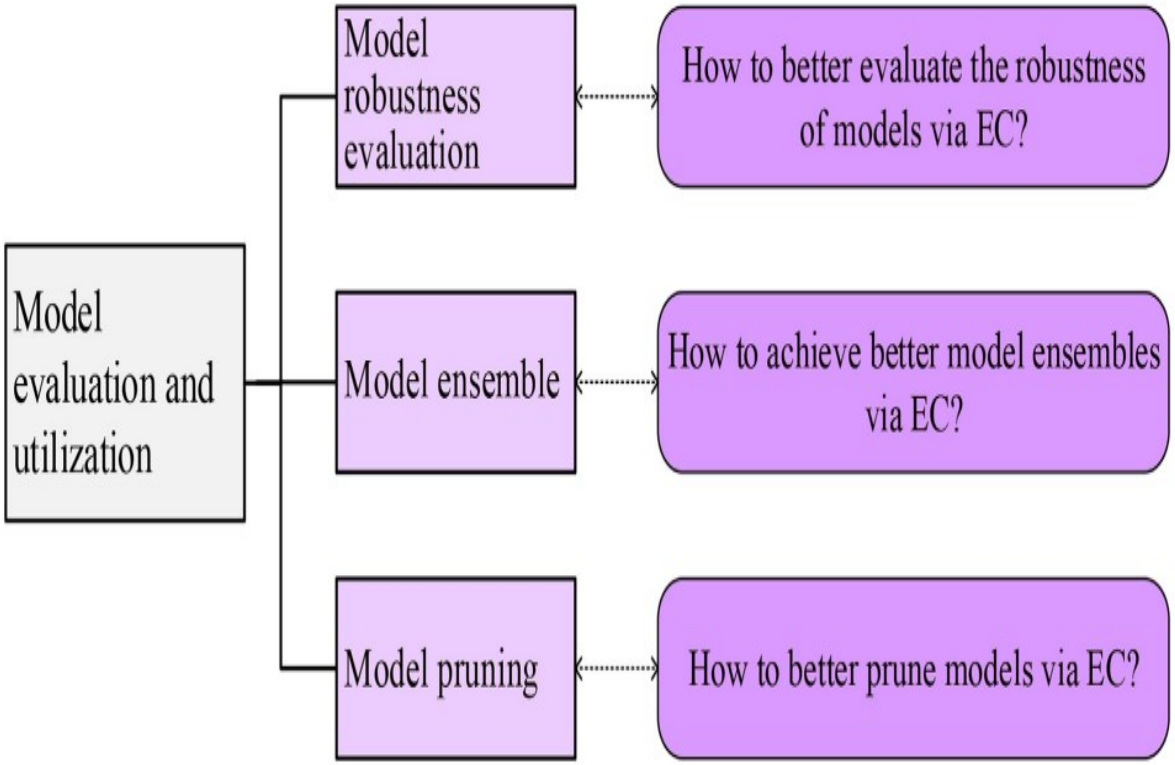


Figure: Categories and corresponding issues of using EDLs in the model evaluation and utilization procedure.

Evolutionary Deep Learning

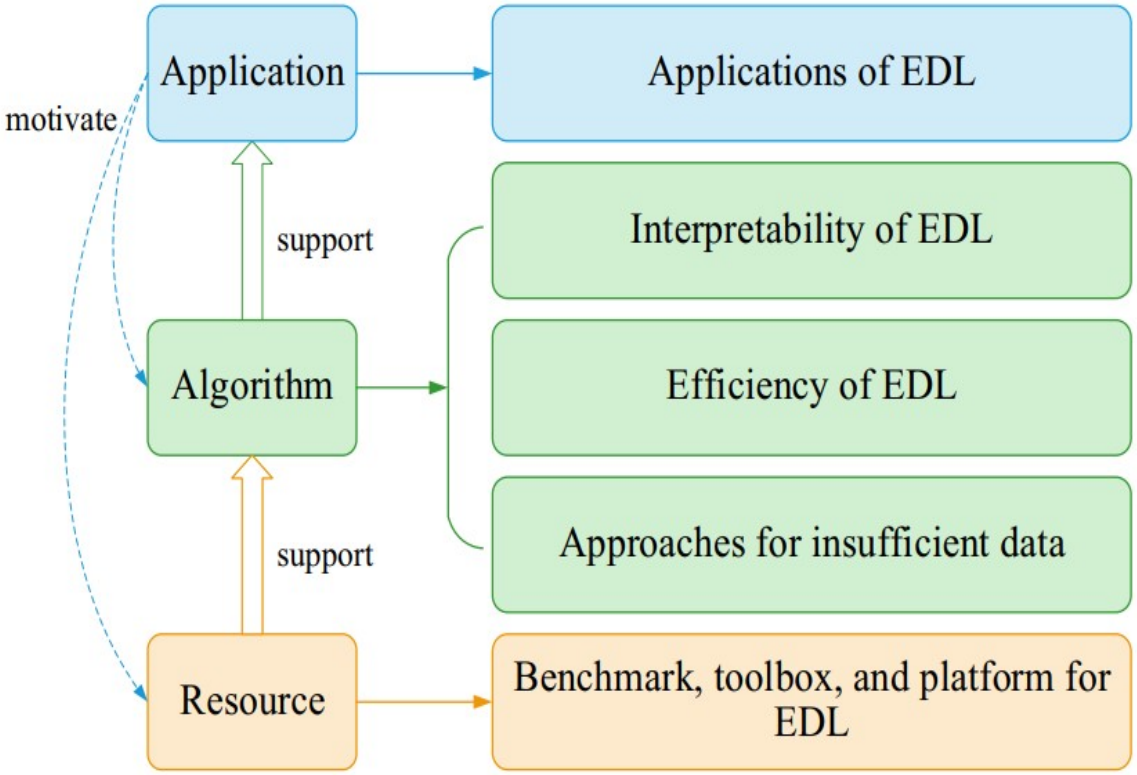


Figure: Some potential future research directions of EDL.

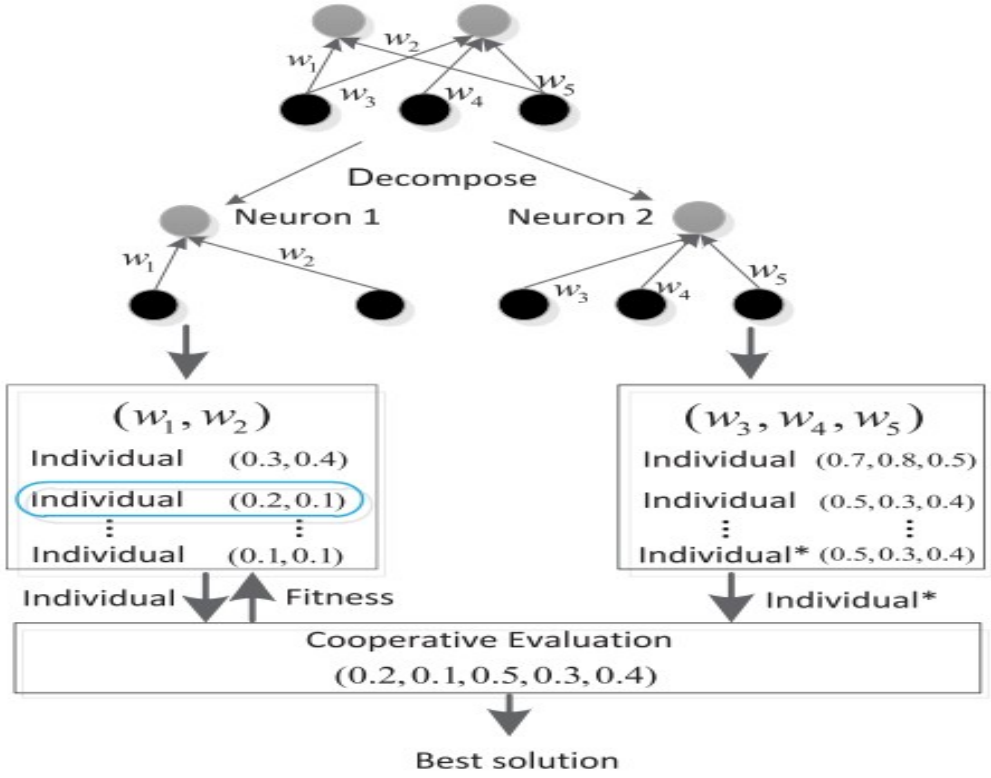


Figure: Neuron-based cooperative co-evolution, where the connection weights for neuron 1 and neuron 2 are partitioned into two separate groups (w1 and w2) and (w3, w4, and w5).

Presentation Outline

1. Evolutionary Deep Learning
2. Pruning and Quantization for Deep Neural Network
3. Conclusion
4. References

Pruning and Quantization for Deep Neural Network

- Deep neural networks have been applied in many applications exhibiting extraordinary abilities in the field of computer vision.
- However, complex network architectures challenge efficient real-time deployment and require significant computation resources and energy costs.
- These challenges can be overcome through optimizations such as network compression.
- Two types of network compression: Pruning and Quantization.
- Pruning can be categorized as static if it is performed offline or dynamic if it is performed at run-time.
- The trade-offs in element-wise, channel-wise, shape-wise, filter-wise, layer-wise and even network-wise is trending pruning.
- Quantization reduces computations by reducing the precision of the datatype. Weights, biases, and activations may be quantized typically to 8-bit integers

Over-parameterization in Deep Neural Network

- Over-parameterization is the property of a neural network where redundant neurons do not improve the accuracy of results.
- However, This redundancy can often be removed with little or no accuracy loss.
- The three design considerations that may contribute to over-parameterization:
- Network structure,
- Network optimization, and
- Hardware accelerator design

Over-parameterization in Deep Neural Network

- Network structure encompasses three parts:
- Network components,
- Network architecture search, and
- Knowledge distillation

Network Components

- Network components is the design of efficient blocks such as separable convolution, inception blocks, and residual blocks.
- Network components also encompasses the types of connections within layers.
- Fully connected deep neural networks require N*N connections between neurons.
- Feed forward layers reduce connections by considering only connections in the forward path.This reduces the number of connections to N.
- Other types of components such as dropout layers can reduce the number of connections even further.

Network Architecture Search (NAS)

- Network Architecture Search (NAS), also known as network auto search, programmatically searches for a highly efficient network structure from a large predefined search space. An estimator is applied to each produced architecture. While time-consuming to compute, the final architecture often outperforms manually designed networks

Knowledge Distillation (KD)

- KD evolved from knowledge transfer. The goal is to generate a simpler compressed model that functions as well as a larger model.

Presentation Outline

1. Evolutionary Deep Learning
2. Pruning and Quantization for Deep Neural Network
3. Conclusion
4. References

Conclusion

- The increasing availability of unstructured data allows artificial intelligence applications to provide better insight into data and thus give companies a competitive advantage.
- Machine learning (ML) is rapidly changing the world, from diverse types of applications and research pursued in industry and academia.
- It is evident from the studies that, automated construction of Deep Neural Networks (DNNs) has become a research hot spot nowadays because DNN's performance is heavily influenced by its architecture and parameters, which are highly task-dependent, but it is notoriously difficult to find the most appropriate DNN in terms of architecture and parameters to best solve a given task.

Presentation Outline

1. Evolutionary Deep Learning
2. Pruning and Quantization for Deep Neural Network
3. Conclusion
4. References

References I

Thank You!

Perceptual Video Processing

by
Dinesh Naik
(B.E, M.Tech, Ph.D)

Assistant Professor, Dept. of Information Technology
National Institute of Technology Karnataka, Surathkal

April 9, 2023

Acknowledgement

- I would like to express my sincere gratitude to Prof. Dr. Abdulmotaleb El Saddik Multimedia Communications Research Laboratory
- I also thank Maaruf Ali, Department of Electronic Engineering, School of Technology, Oxford Brookes University

IMAGE COMPRESSION – AN INTRODUCTION

- to reduce storage space;
- to reduce the time of transmission of data files.
- to enable the use of lower data transmission rates;
- compressed images cost less: to transmit, archive, process than uncompressed images;
- to allow faster processing of image data – important in Knowledge Based Systems;
- to overcome the electromagnetic spectrum overcrowding – e.g. satellite television transmission. An example being the transmission of nine television channels from a single 25 MHz satellite transponder – previously, only a single analogue television channel would have been broadcast using such a transponder.

How is Image Compression Achieved

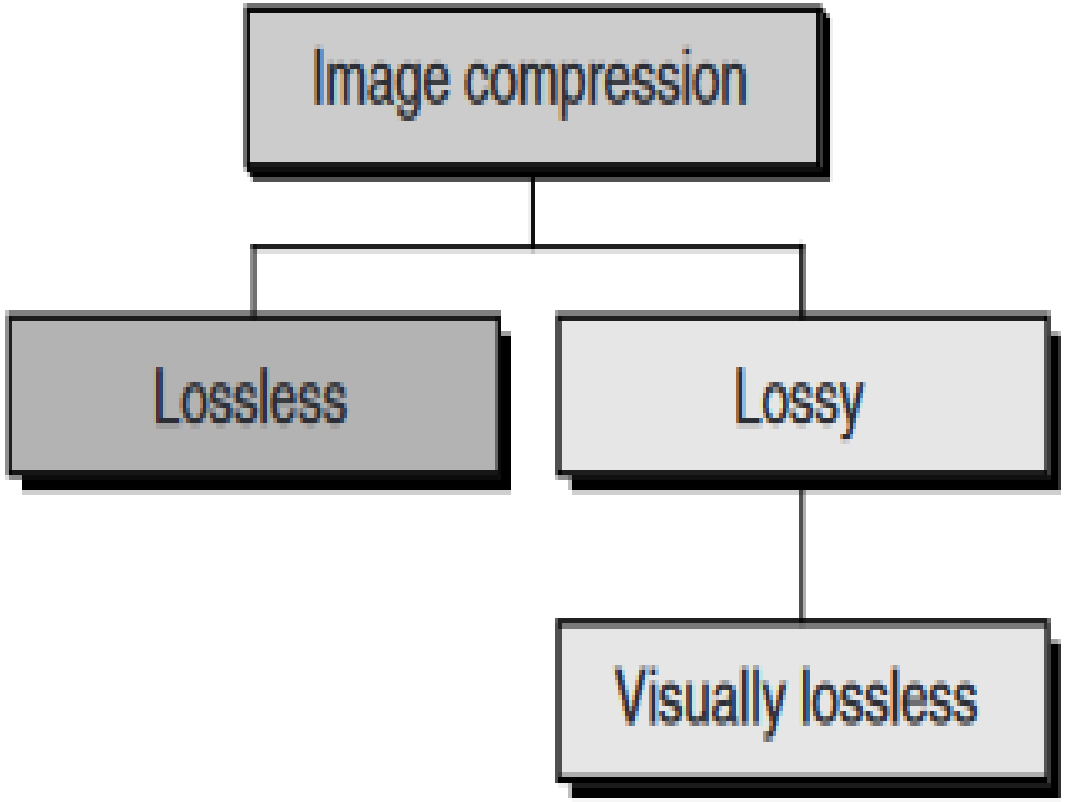


Figure: Classification of image compression algorithms

How is Image Compression Achieved

- Image compression is achieved by exploiting the redundancy in the image data and the non-linearities of the human visual system which is less sensitive to certain spatial and temporal frequencies than others.
- Redundancy in images exists in two main forms.
- Statistical redundancy – this arises from the probability distribution in the image data itself, being the spatial correlation of the pixels in a single image data and temporal correlation between frames for motion video. There is also spectral redundancy – in images composed of more than one spectral band, the spectral values for the same pixel location are often correlated.
- Psychovisual redundancy – this utilises the non-linearities of the human visual system to remove detail from moving and still images to which the eye is the least sensitive to and thus the perceived image quality is not affected.

Categorisation of the various types of redundancies present in images

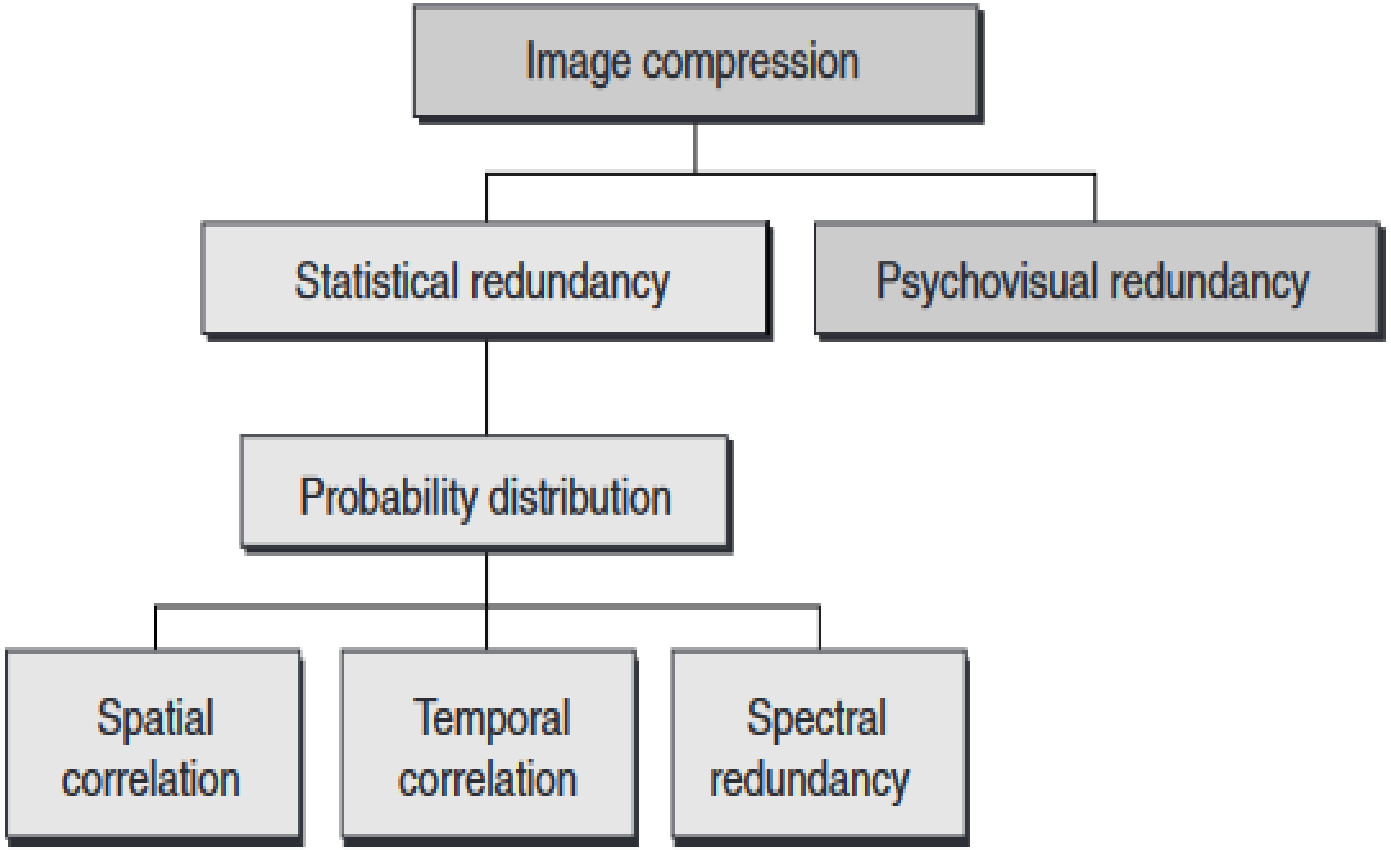


Figure: The types of redundancies present in image(s)

Motivation

Digital video in computing means for

- **Text:**
  - 1 page with 80 char/line and 64 lines/page and 2 Byte/Char
    - 80 x 64 x 2 x 8 = 80 kBit/page
- **Image:**
  - 24 Bit/Pixel, 512 x 512 Pixel/image
    - 512 x 512 x 24 = 6 MBit/Image
- **Audio:**
  - CD-quality, samplerate44,1 kHz, 16 Bit/sample
    - Mono: 44,1 x 16 = 706 kBit/s
    - Stereo: 706 x 2 = 1.412 MBit/s
- **Video:**
  - full frames with 1024 x 1024 Pixel/frame, 24 Bit/Pixel, 30 frames/s
    - 1024 x 1024 x 24 x 30 = 720 MBit/s
  - more realistic
    - 360 x 240 Pixel/frame = 60 MBit/s

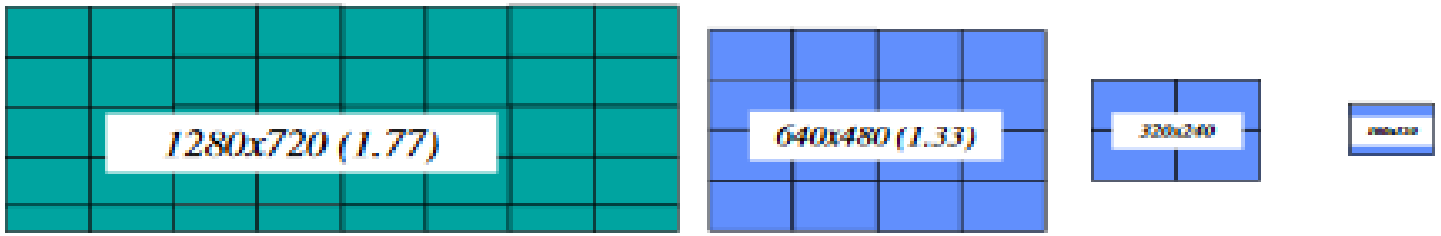
Hence compression is *necessary*

Video Data Size

size of uncompressed video in gigabytes

	1920x1080	1280x720	640x480	320x240	160x120
1 sec	0.19	0.08	0.03	0.01	0.00
1 min	11.20	4.98	1.66	0.41	0.10
1 hour	671.85	298.60	99.53	24.88	6.22
1000 hours	671,846.40	298,598.40	99,532.80	24,883.20	6,220.80

image size of video

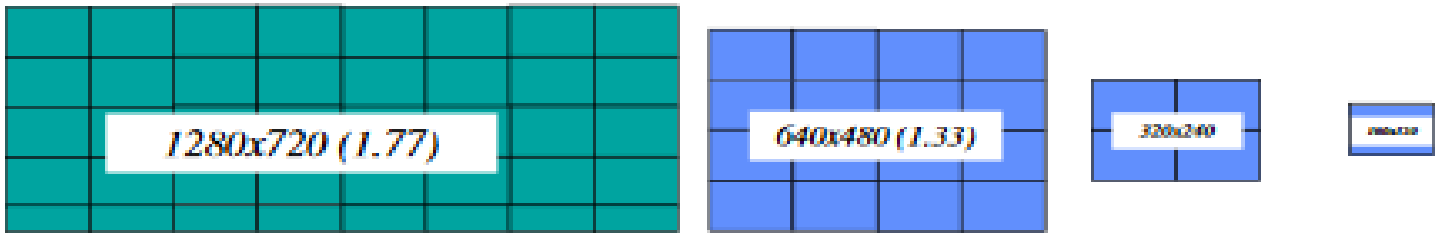


Video Data Size

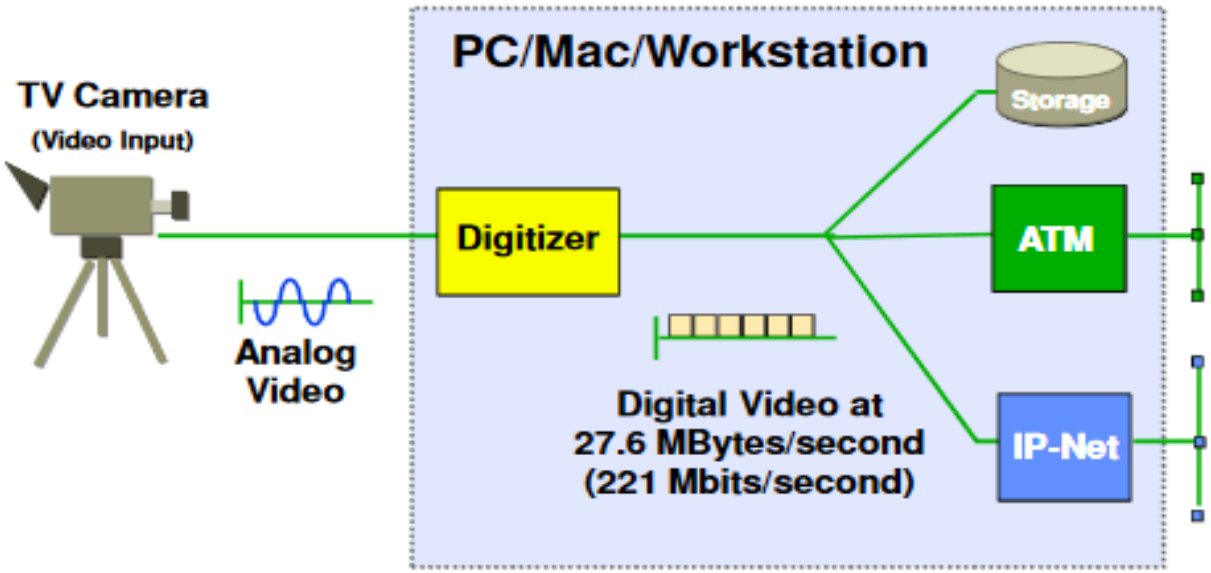
size of uncompressed video in gigabytes

	1920x1080	1280x720	640x480	320x240	160x120
1 sec	0.19	0.08	0.03	0.01	0.00
1 min	11.20	4.98	1.66	0.41	0.10
1 hour	671.85	298.60	99.53	24.88	6.22
1000 hours	671,846.40	298,598.40	99,532.80	24,883.20	6,220.80

image size of video



Digitizing Video

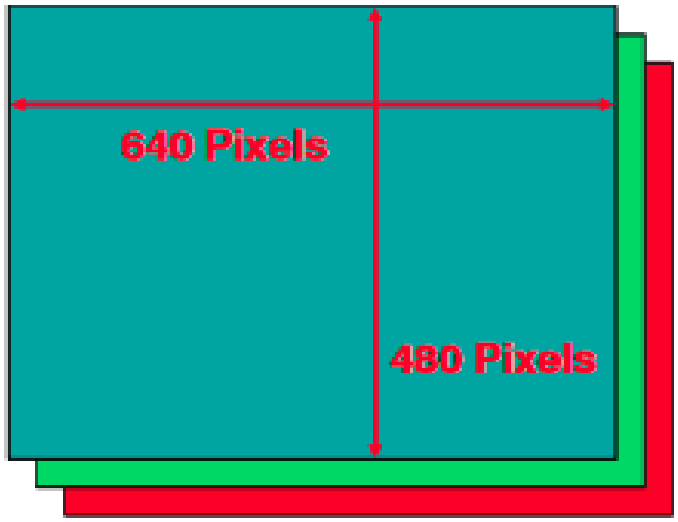


Transforming from Time domain into Frequency domain

Data in the transformed domain is easier to compress

Video Digitization Parameters

"Full Screen Video"



(Pixel Area) x (Color Depth) x (Frame Rate) = Bandwidth

(640*480) * 3 * 30 = 27,648,000 Bytes/s

Or

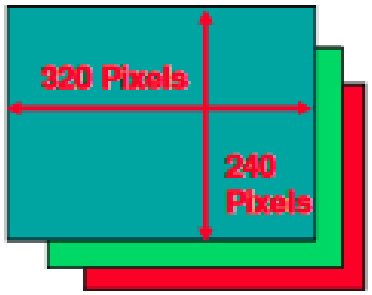
221,184,000 bits per second (221Mbps)

3-bytes (24-bit) Color

NTSC = 30 frames per second
PAL/SECAM = 25 frames per second

Modify Digitization Parameters

1/4 Screen Video



1-byte (8-bit) Color

10 frames per second

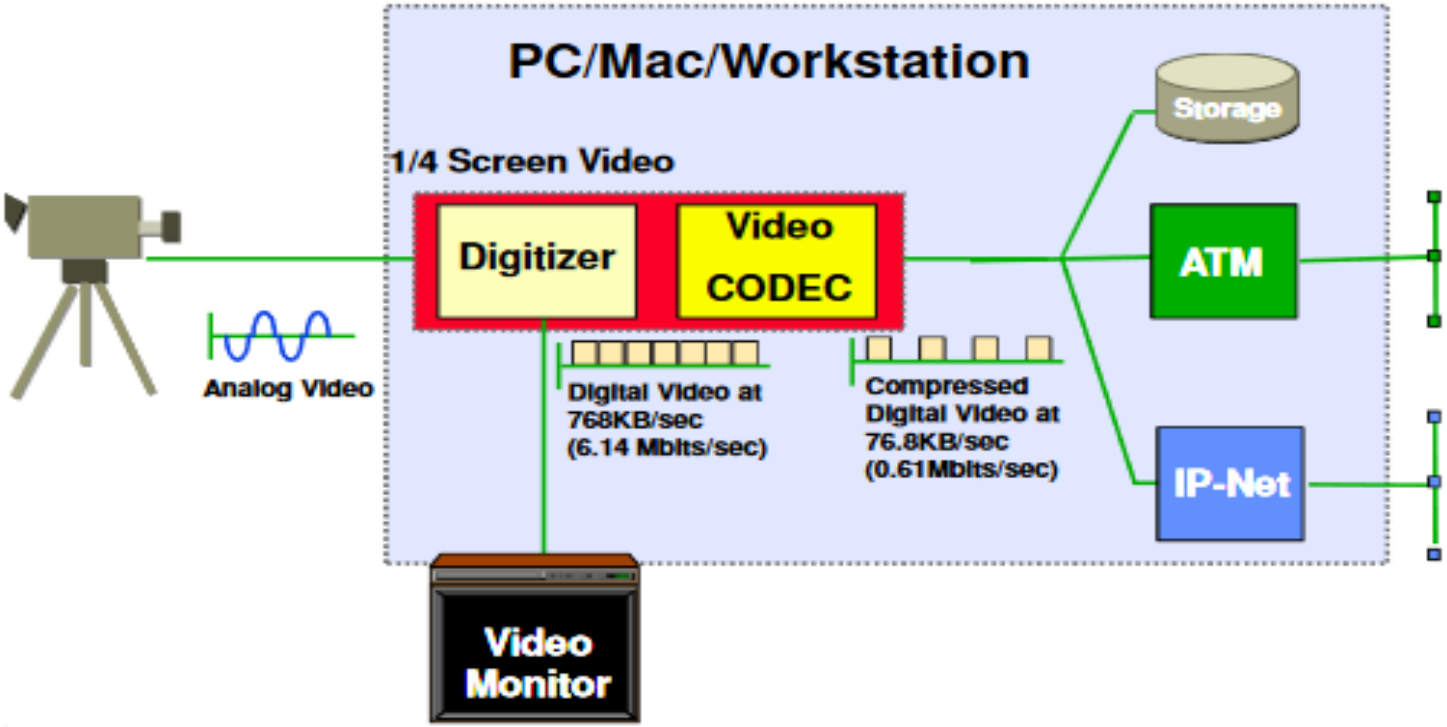
(Pixel Area) x (Color Depth) x (Frame Rate) = Bandwidth

(320*240) * 1 * 10 = 768,000 Bytes/sec

or

6,144,000 bits/sec (6.14Mbps)

Video Compression



Video Bit Rate Calculation

$$\left[\frac{\text{width} * \text{height} * \text{depth} * \text{fps}}{\text{compression factor}}\right] = \text{bits/sec}$$

width = {160, 320, 640, 720, 1280, 1920, ...} pixels

height = {120, 240, 480, 485, 720, 1080, ...} pixels

depth = {1, 4, 8, 15, 16, 24, ...} bits

fps = {5, 15, 20, 24, 30, ...} frames per second

compression factor = {1, 6, 24, ...}

Effects of Compression

storage for 1 hour of compressed video in megabytes

1920x1080	1280x720	640x480	320x240	160x120
671,846	298,598	99,533	24,883	6,221
223,949	99,533	33,178	8,294	2,074
111,974	49,766	16,589	4,147	1,037
26,874	11,944	3,981	995	249
6,718	2,986	995	249	62

3 bytes/pixel, 30 frames/sec

Constraints for Multimedia Systems

- Compression/Decompression
  - Uncompressed video is rather tolerant to transmission errors, but compressed one is very sensitive to network transmission impairments
  - Decompression of images, video or audio should be possible without the knowledge of other data units. Media synchronization will take care of all presentational needs
- In conversational multimedia applications (Dialogue-based)
  - The overall end-to-end delay should not exceed 150 ms. However, the compression-decompression delay should be reduced to about 50 ms, to leave room for other network introduced delay components
- For presentational multimedia (Retrieval mode)
  - fast-forward,-backward, with simultaneous display, should be possible
  - random access to single images and audio frames of a data stream, with access time less than 0.5 sec

Coding Categories

Source coding

- based on data semantics
- often lossy

Entropy coding

- ignoring data semantics
- often lossless

Channel coding

- adaptation to comm. channel
- introduction of redundancy

Hybrid coding

- entropy
- source coding

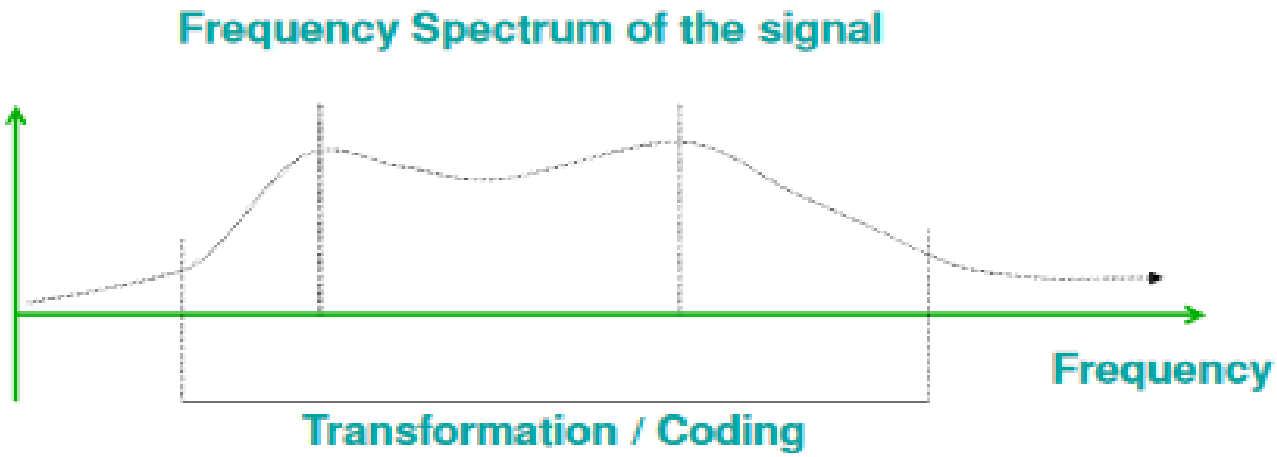
Coding Categories and Techniques

Entropy coding	Run-Length coding	
	Huffman coding	
	Arithmetic coding	
Source coding	Prediction	DPCM
		DM
	Transformation	FFT
		DCT
	Layered coding	Bit Position
		Sub-sampling
		Sub-band coding
	Vector Quantization	
Hybrid coding	JPEG	
	MPEG	
	H.261	
	Proprietary format: Quicktime, DVI RTV, DVI PLV,...	

Source Coding: Sub-Band

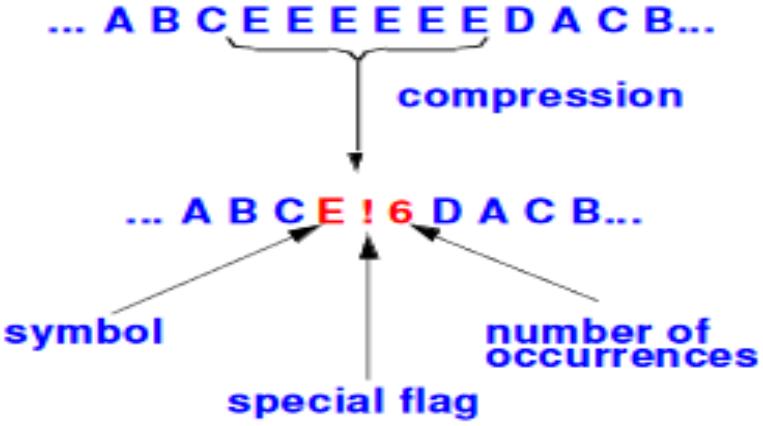
Assumption:

- Some frequency ranges are more important than others



Entropy Coding: Run-Length

- **Assumption:**
  - Long sequences of identical symbols
- **Example:**
  - Special variant: zero-length encoding
  - only repetition of zeroes count
  - in red, "symbol" not needed (i.e. "pays" for >3 repetitions)



Huffman Coding

Assumption:

- Some symbols occur more often than other
- E.g., characters in an English text

Principle:

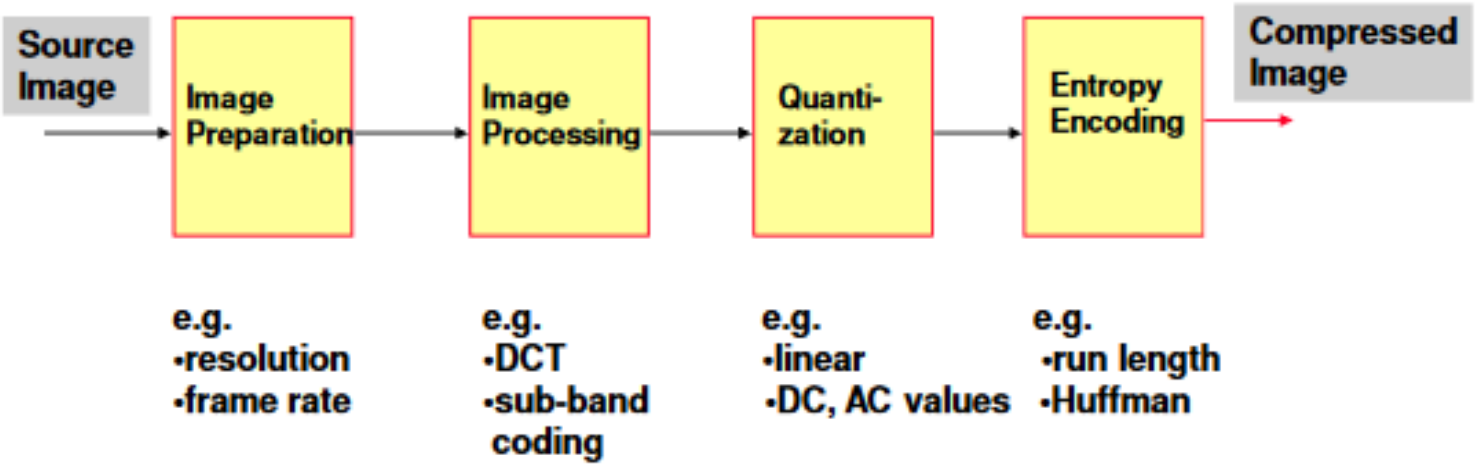
- Frequently occuring symbols are coded with shorter strings

Table:

Symbol	Code
A	001
B	1
C	011
D	000
E	010

A	B	D	C	A	A	E
001	1	000	011	001	001	010

Basic Encoding Steps



JPEG International Standard since 1992

- JPEG (Joint Photographic Expert Group) is a standard jointly developed by ISO/EIC JTC1/SC2/WG10 and ITU-TS
- Can compress still images 10:1 and 50:1, without visibly affecting image quality
- Uses the Discrete Cosine Transform (DCT) for coding and decoding
- Aimed at photographic material, thus it does not have embedded encoded/compressed audio signal
- Standard supports four modes
  - Sequential encoding
  - Progressive encoding
  - Lossless encoding
  - Hierarchical encoding
- 10 MHz JPEG chip can compress full-page, 300 dpi, 24-bit colour image in 1 sec or a 640 x 480 24-bit-colour image in 0.1 sec
- 25 MHz chip can do the same in 0.03 sec (Real-time video compr.)

IMAGE COMPRESSION BY DISCRETE COSINE TRANSFORM

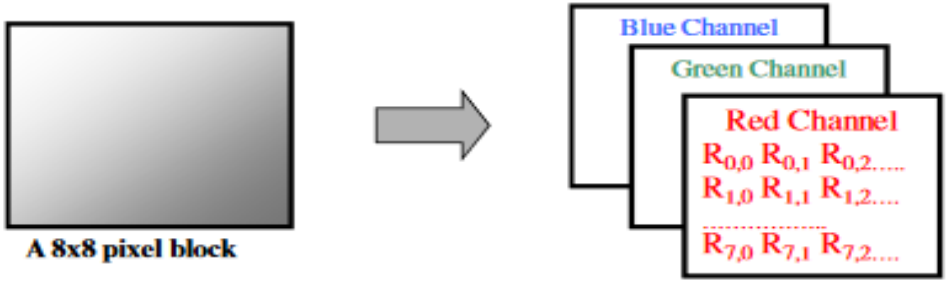
- The DCT is a core compression technology used in the industry standards JPEG (Joint Photographic Experts Group) for still-image compression and MPEG (Motion Picture Experts Group) for video compression, achieving a compression ratio of 20:1 without noticeable quality degradation.

- The principle of the DCT is to transform the original image pixels to an identical number of DCT coefficients, where the DCT coefficients have a nonuniform distribution of direct-current (DC) terms representing the average values, and alternate-current (AC) terms representing fluctuations.

- The compression is achieved by applying the advantages of encoding DC terms (with a large dynamic range) with a large number of bits and low-frequency AC terms (a few, with a reduced dynamic range) with a reduced number of bits, and neglecting some high-frequency AC terms that have small dynamic ranges (most of them do not affect the visual quality of the picture).

Image Preparation

- Before performing DCT, divide image into blocks (usually 8x8 pixels)
- Each pixel in an image is represented by image components, usually 3 components (e.g. the RGB components)
- These are normally converted to a different format, such as the YUV, where the luminance and chrominance are separated.



Discrete Cosine Transform (DCT)

- The blocks of components are then transformed into frequency components.
- Each block is represented by 64 frequency components, one of which (the DC component) is the average colour of the 64 pixels in the block

- Applying FDCT to change the blocks from spatial domain to frequency domain
- Roughly speaking, F(0,0) is the average value within the block (called DC component).
- Other values stores the variation within a block (AC components)

Two-Dimensional Discrete Cosine Transform

- Image compression uses 2D-DCT, whose transform pairs are defined as follows:

Forward DCT:

$$F(u,v) = \frac{2C(u)C(v)}{\sqrt{MN}} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} p(i,j) \cos\left(\frac{(2i+1)u\pi}{2M}\right) \cos\left(\frac{(2j+1)v\pi}{2N}\right)$$

Inverse DCT:

$$p(i,j) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} \frac{2C(u)C(v)}{\sqrt{MN}} F(u,v) \cos\left(\frac{(2i+1)u\pi}{2M}\right) \cos\left(\frac{(2j+1)v\pi}{2N}\right)$$

$$C(m) = \begin{cases} \frac{\sqrt{2}}{2} & if\ m = 0 \\ 1 & otherwise \end{cases}$$

$p(i, j) =$ pixel level at the location (i, j)

$F(u, v) =$ DCT coefficient at the frequency indices (u, v)

Two-Dimensional Discrete Cosine Transform

- JPEG divides an image into 8 x 8 image subblocks and applies DCT for each subblock individually. Hence, we simplify the general 2D-DCT in terms of 8 x8 size. The equation for 2D 8 x 8 DCT is modified as

$$F(u,v) = \frac{C(u)C(v)}{4} \sum_{i=0}^{7} \sum_{j=0}^{7} p(i,j) \cos\left(\frac{(2i+1)u\pi}{16}\right) \cos\left(\frac{(2j+1)v\pi}{16}\right)$$

The inverse of 2D 8 x 8 DCT is expressed as

$$p(i,j) = \sum_{u=0}^{7} \sum_{v=0}^{7} \frac{C(u)C(v)}{4} F(u,v) \cos\left(\frac{(2i+1)u\pi}{16}\right) \cos\left(\frac{(2j+1)v\pi}{16}\right)$$

Determine the 2D-DCT coefficients for the following image:

$$\begin{bmatrix} 100 & 50 \\ 100 & -10 \end{bmatrix}$$

Solution:

Applying $N = 2$ and $M = 2$ to Equation (14.18) yields

$$F(u,v) = \frac{2C(u)C(v)}{\sqrt{2 \times 2}} \sum_{i=0}^{1} \sum_{j=0}^{1} p(i,j) \cos\left(\frac{(2i+1)u\pi}{4}\right) \cos\left(\frac{(2j+1)v\pi}{4}\right)$$

For $u = 0$ and $v = 0$, we achieve

$$\begin{aligned} F(0,0) &= c(0)c(0) \sum_{i=0}^{1} \sum_{j=0}^{1} p(i,j) \cos(0) \cos(0) \\ &= \left(\frac{\sqrt{2}}{2}\right)^2 [p(0,0) + p(0,1) + p(1,0) + p(1,1)] \\ &= \frac{1}{2}(100 + 50 + 100 - 10) = 120 \end{aligned}$$

For $u = 0$ and $v = 1$, we achieve

$$F(0,1) = c(0)c(1)\sum_{i=0}^{1}\sum_{j=0}^{1} p(i,j)\cos(0)\cos\left(\frac{(2j+1)\pi}{4}\right)$$

$$= \left(\frac{\sqrt{2}}{2}\right) \times 1 \times \left(p(0,0)\cos\frac{\pi}{4} + p(0,1)\cos\frac{3\pi}{4} + p(1,0)\cos\frac{\pi}{4} + p(1,1)\cos\frac{3\pi}{4}\right)$$

$$= \frac{\sqrt{2}}{2}\left(100 \times \frac{\sqrt{2}}{2} + 50\left(-\frac{\sqrt{2}}{2}\right) + 100 \times \frac{\sqrt{2}}{2} - 10\left(-\frac{\sqrt{2}}{2}\right)\right) = 80$$

Similarly,

$$F(1,0) = 30, \text{ and } F(1,1) = -30$$

Finally, we get

$$F(u,v) = \begin{bmatrix} 120 & 80 \\ 30 & -30 \end{bmatrix}$$

Applying the MATLAB function **dct2()** verifies the DCT coefficients as follows:

```
>> F = dct2([100 50;100 -10])
F =
    120.0000   -80.0000
    30.0000    -30.0000
```

Given the following DCT coefficients from a 2×2 image, determine the pixel $p(0,0)$:

$$F\left(u,v\right) = \begin{bmatrix} 120 & 80 \\ 30 & -30 \end{bmatrix}$$

Solution:

Applying Equation (14.19) of the inverse 2D-DCT with $N = M = 2$, $i = 0$, and $j = 0$, it follows that

$$\begin{aligned} p(0,0) &= \sum_{u=0}^{1}\sum_{v=0}^{1} c(u)c(v)F(u,v)\cos\left(\frac{u\pi}{4}\right)\cos\left(\frac{v\pi}{4}\right) \\ &= \left(\frac{\sqrt{2}}{2}\right)\times\left(\frac{\sqrt{2}}{2}\right)\times F(0,0) + \left(\frac{\sqrt{2}}{2}\right)\times F(0,1)\times\left(\frac{\sqrt{2}}{2}\right) \\ &+\left(\frac{\sqrt{2}}{2}\right)\times F(1,0)\times\left(\frac{\sqrt{2}}{2}\right) + F(0,1)\left(\frac{\sqrt{2}}{2}\right)\times\left(\frac{\sqrt{2}}{2}\right) \\ &= \frac{1}{2}\times 120 + \frac{1}{2}\times 80 + \frac{1}{2}\times 30 + \frac{1}{2}(-30) = 100 \end{aligned}$$

We apply the MATLAB function **idct2()** to verify the inverse DCT and get the following pixel values:

```
>> p = idct2([120 80; 30 -30])
p =
   100.0000    50.0000
   100.0000   -10.0000
```

Pixel block

-127	-128	-123	-119	-115	-111	-107	-103
-128	-123	-119	-115	-111	-107	-103	-100
-123	-119	0	-111	-107	-2	-100	-95
-119	-115	-111	-107	-103	-100	-95	-91
2	-111	-107	-103	-100	-95	-91	15
-111	-15	-23	-30	-21	-20	-2	-83
-107	-103	-100	-95	-91	-87	-83	-79
-103	-100	-95	-91	-87	-83	-79	-75

FDCT

-711.25	-66.59	-0.02	-8.63	-22.00	-1.24	36.54	3.57
-105.04	0.21	-4.02	-0.92	-10.18	1.68	35.04	0.67
-81.72	-4.87	-26.86	0.47	-7.80	-0.50	-30.26	-2.16
58.90	3.08	19.46	4.13	27.20	-1.00	-56.33	-0.36
-58.00	-0.03	73.05	6.19	78.25	3.76	-5.90	-2.49
-42.73	-2.63	-39.20	-2.30	-32.02	-1.07	-3.90	-0.20
97.98	3.13	-66.26	-5.85	-79.57	-2.19	20.86	4.63
0.00	3.38	58.37	2.11	33.65	4.57	78.96	0.73

Block of DCT coefficients

Quantization Table

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

- DCT coefficients are divided by Quantization values and rounded off
- Q-values are chosen based on DCT co-eff. importance

Example

Quantization Table

$$\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

-711.25 /16

= -44

-711.25	-66.59	-0.02	-8.63	-22.00	-1.24	36.54	3.57
-105.04	0.21	-4.02	-0.92	-10.18	1.68	35.04	0.67
-81.72	-4.87	-26.86	0.47	-7.80	-0.50	-30.26	-2.16
58.90	3.08	19.46	4.13	27.20	-1.00	-56.33	-0.36
-58.00	-0.03	73.05	6.19	78.25	3.76	-5.90	-2.49
-42.73	-2.63	-39.20	-2.30	-32.02	-1.07	-3.90	-0.20
97.98	3.13	-66.26	-5.85	-79.57	-2.19	20.86	4.63
0.00	3.38	58.37	2.11	33.65	4.57	78.96	0.73

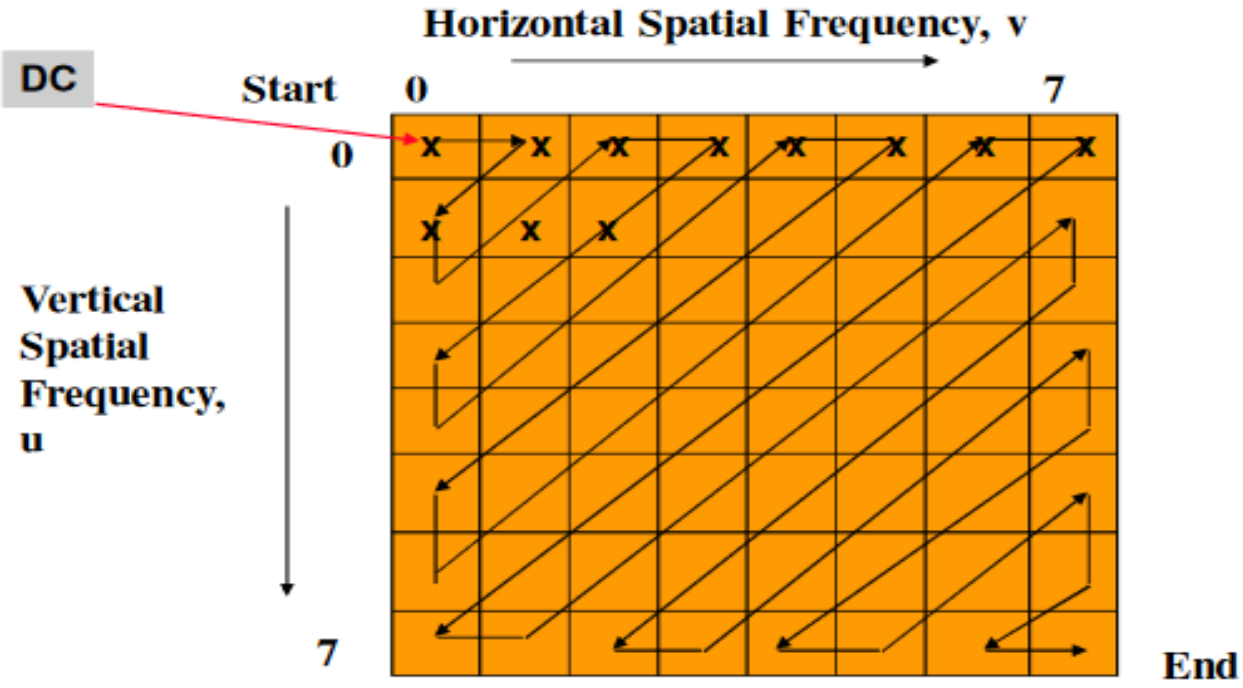
Block of DCT coefficients

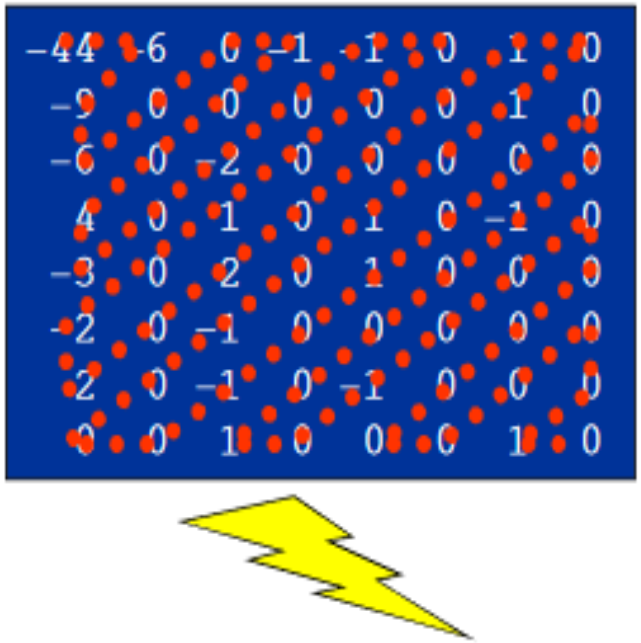
-44	-6	0	-1	-1	0	1	0
-9	0	0	0	0	0	1	0
-6	0	-2	0	0	0	0	0
4	0	1	0	1	0	-1	0
-3	0	2	0	1	0	0	0
-2	0	-1	0	0	0	0	0
2	0	-1	0	-1	0	0	0
0	0	1	0	0	0	1	0

Quantized Values

Zig-Zag sequence

- Use Zig-Zag walks to order the coefficients sequentially






```
-44, -6, -9, -6, 0, 0, -1, 0, 0, 4, -3, 0, -2, 0, -1, 0, 0, 0, 1, 0, -2, 2,
0, 2, 0, 0, 0, 1, 0, 1, 0, 1, 0, -1, 0, 0, 0, -1, 0, 1, 0, 0, 0, 0, -1, 0,
0, 0, 1, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0
```

Run-Length Encoding

- Coefficients are represented by [Repeats-1] [coefficient+1024]

```
-44, -6, -9, -6, 0, 0, -1, 0, 0, 4, -3, 0, -2, 0, -1, 0, 0, 0, 1, 0, -2, 2,
0, 2, 0, 0, 0, 1, 0, 1, 0, 1, 0, -1, 0, 0, 0, -1, 0, 1, 0, 0, 0, 0, -1, 0,
0, 0, 1, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0
```

```
[1,-44], [1,-6], [1,-9], [1,-6], [2,0], [1,-1], [2,0], [1,4], [1,-3], [1,0],
[1,-2], [1,0], [1,-1], [3,0], [1,1], [1,0], [1,-2], [1,2], [1,0], [1,2], [3,0],
[1,1], [1,0], [1,1], [1,0], [1,1], [1,0], [1,-1], [3,0], [1,-1], [1,0], [1,1],
[4,0], [1,-1], [3,0], [1,1], [1,0], [1,-1], [11,0], [1,1], [1,0]
```

```
[0,980], [0,1018], [0,1015], [0,1018], [1,1024], [0,1023], [1,1024],
[0,1028], [0,1021], [0,1024], [0,1022], [0,1024], [0,1023], [2,1024],
[0,1025], [0,1024], [0,1022], [0,1026], [0,1024], [0,1026], [2,1024],
[0,1025], [0,1024], [0,1025], [0,1024], [0,1025], [0,1024], [0,1023],
[2,1024], [0,1023], [0,1024], [0,1025], [3,1024], [0,1023], [2,1024],
[0,1025], [0,1024], [0,1023], [10,1024], [0,1025], [0,1024]
```

Run-Length Encoding

- ***Repeats-1*** **has maximum value of 63, need 6 bits.**
- ***Quantized coefficient+1024*** **has range [0,2047], need 11 bits.**

[0,980]	00000 0	01111 01010 0
[0,1018]	00000 0	01111 11101 0
[0,1015]	00000 0	01111 11011 1
[0,1018]	00000 0	01111 11101 0
. . .	. . .	
[0,1024]	00000 0	10000 00000 0

Thank You!

Perceptual Video Processing

by
Dinesh Naik
(B.E, M.Tech, Ph.D)

Assistant Professor, Dept. of Information Technology
National Institute of Technology Karnataka, Surathkal

April 9, 2023

Acknowledgement

- I would like to express my sincere gratitude to Prof. Dr. Abdulmotaleb El Saddik Multimedia Communications Research Laboratory
- I also thank Maaruf Ali, Department of Electronic Engineering, School of Technology, Oxford Brookes University

Two-Dimensional JPEG Grayscale Image Compression Example

- We examine an 8×8 grayscale subblock
- Applying 2D-DCT leads to DCT coefficients shown in figure.
- These DCT coefficients have a big DC component of 1198 but small AC component values.
- These coefficients are further normalized (quantized) with a quality factor Q as shown in figures .
- After normalization, as shown in figures, the DC coefficient is reduced to 75, and a few small AC coefficients exist, while most are zero. We can encode and transmit only nonzero DCT coefficients and omit transmitting zeros, since they do not carry any information.
- They can be easily recovered by resetting coefficients to zero during decoding. By this principle we achieve data compression.

How is Image Compression Achieved

Figure: Original grayscale image

150	148	140	132	150	155	155	151
155	152	143	136	152	155	155	153
154	149	141	135	150	150	150	150
156	150	143	139	154	152	152	155
156	151	145	140	154	152	152	155
154	152	146	139	151	149	150	151
156	156	151	142	154	154	154	154
151	154	149	139	151	153	154	153

Figure: 8 x 8 grayscale subblock

DCT Coefficients for the Subblock Image

1198	−10	26	24	−5	−16	0	12
−8	−6	3	8	0	0	0	0
0	−3	0	0	−8	0	0	0
0	0	0	0	0	0	0	0
0	−4	0	0	0	0	0	0
0	0	−1	0	0	0	0	0
−10	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0

The Quality Factor

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Normalized DCT Coefficients

75	−1	3	2	0	0	0	0
−1	−1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

- As shown in above figure, most nonzero coefficients reside in the upper left corner. Hence, the order of encoding for each value is based on the zigzag path in which the order is numbered, as in next Figure.
- According to the order, we record the nonzero DCT coefficients as a JPEG vector, shown as
- JPEG vector : [75 -1 -1 0 -1 3 2 EOB].
- where "EOB" is end of block coding. The JPEG vector can further be compressed by encoding the difference of DC values between subblocks, using differential pulse code modulation (DPCM), as well as by run-length coding of AC values and Huffman coding, which both belong to lossless compression techniques.

DCT Coefficient Scan Order

0	1	5	6	14	15	27	28
2	4	7	13	16	26	29	42
3	8	12	17	25	30	41	43
9	11	18	24	31	40	44	53
10	19	23	32	39	45	52	54
20	22	33	38	46	51	55	60
21	34	37	47	50	56	59	61
35	36	48	49	57	58	62	63

- During the decoding stage, the JPEG vector is recovered first. Then the quantized DCT coefficients are recovered according to the zigzag path.
- Next, the recovered DCT coefficients are multiplied by a quality factor to obtain the estimate of the original DCT coefficients.
- Finally, we apply the inverse DCT to achieve the recovered image subblock, which is shown in next slide.
- For comparison, the errors between the recovered image and original image are calculated.

The Recovered Image Subblock

153	145	138	139	147	154	155	153
153	145	138	139	147	154	155	153
155	147	139	140	148	154	155	153
157	148	141	141	148	155	155	152
159	150	142	142	149	155	155	152
161	152	143	143	149	155	155	152
162	153	144	144	150	155	154	151
163	154	145	144	150	155	154	151

The Coding Error of the Image Subblock

3	−3	−2	7	−3	−1	0	2
−2	−7	−5	3	−5	−1	0	0
1	−2	−2	5	−2	4	5	3
1	−2	−2	2	−6	3	3	−3
3	−1	−3	2	−5	3	3	−3
7	0	−3	4	−2	6	5	1
6	−3	−7	2	−4	1	0	−3
12	0	−4	5	−1	2	0	−2

JPEG Color Image Compression

- Initially we focus on the encoder, since the decoder is just the reverse process of the encoding. The block diagram for the JPEG encoder is shown in Figure.
- The JPEG encoder has the following main steps.
- 1. Transform RGB to YIQ or YUV (U and V = chrominance components).
- 2. Perform DCT on blocks.
- 3. Perform quantization.
- 4. Perform zigzag ordering, DPCM, and run-length encoding.
- 5. Perform entropy encoding (Huffman coding)

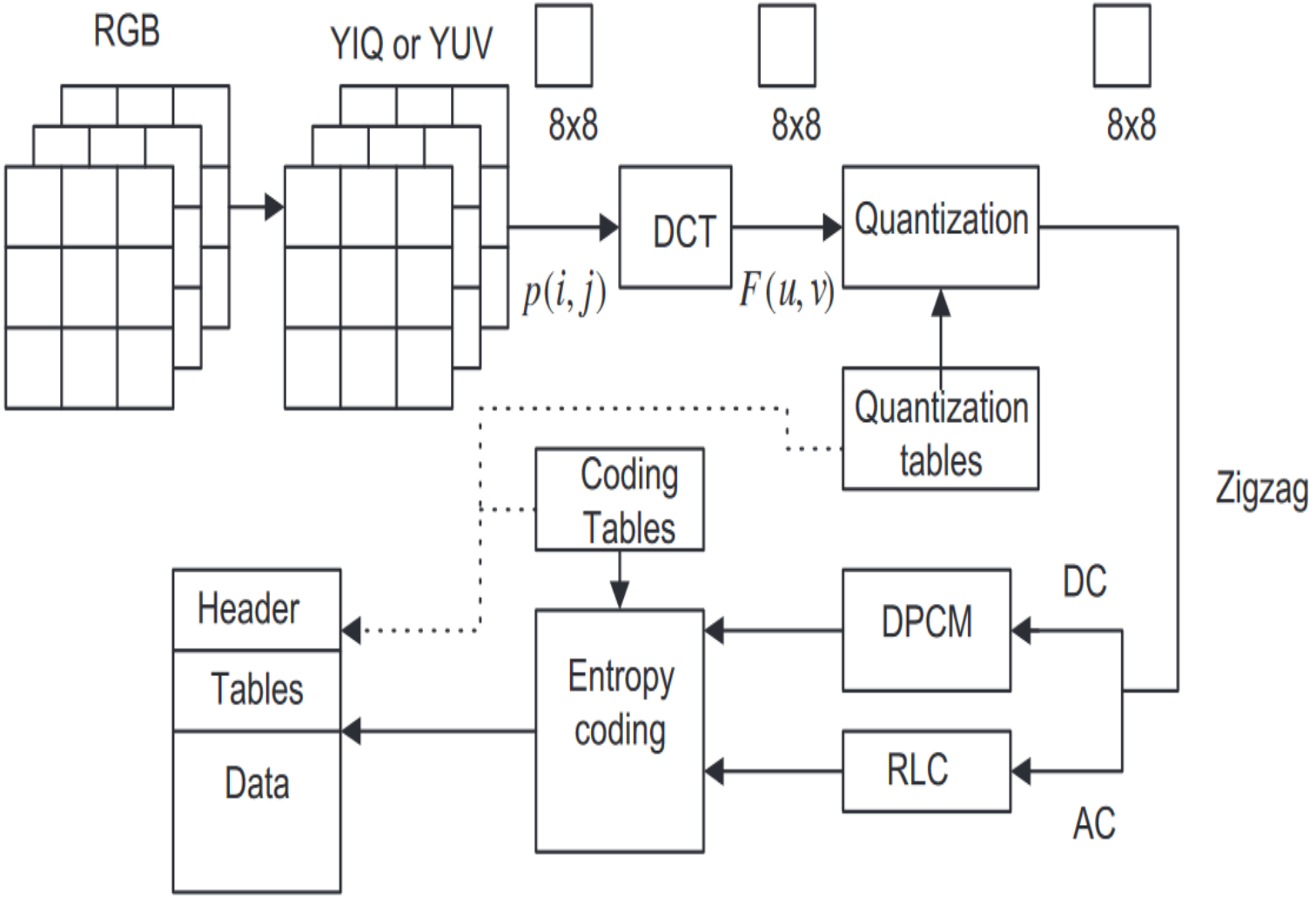


Figure: Block diagram for JPEG encoder.

RGB to YIQ Transformation

- The first transformation is of the RGB image to a YIQ or YUV image.
- The principle of transformation from RGB to YIQ, luminance channel carries more signal energy, up to 93%, while the chrominance channels carry up to only 7% of signal energy.
- After transformation, more effort can be spent on coding the luminance channel.

DCT on Image Blocks

- Each image is divided into 8 x 8 blocks.
- 2D-DCT is applied to each block to obtain the 8 x 8 DCT coefficient block.
- Note that there are three blocks, Y, I, and Q.

Quantization

- The quantization is operated using the 8 x 8 quantization matrix.
- Each DCT coefficient is quantized, that is, divided by the corresponding value given in the quantization matrix. In this way, a smaller number of bits can be used for encoding the DCT coefficients.
- There are two different quantization tables, one for luminance and the other for chrominance.
- We can see that the chrominance table has numbers with larger values, so that small values of DCT coefficients will result and hence fewer bits are required for encoding each DCT coefficient.
- Zigzag ordering to produce the JPEG vector is similar to the grayscale case, except that there are three JPEG vectors.

The Quality Factor for Luminance

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Figure: Block diagram for JPEG encoder.

The Quality Factor for Chrominance

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

Figure: Block diagram for JPEG encoder.

Differential Pulse Code Modulation on Direct-Current Coefficients

- Since each 8 x 8 image block has only one DC coefficient, which can be a very large number and varies slowly, we make use of DPCM for coding DC coefficients.
- As an example for the first five image blocks, the DC coefficients are 200, 190, 180, 160, and 170. DPCM with a coding rule of d(n) = DC(n)-DC(n-1)
- with initial condition d(0) = DC(0) produces a DPCM sequence of 200; -10; -10; -20; 10
- Hence, the reduced signal range of these values is feasible for entropy coding.

Run-Length Coding on Alternating-Current Coefficients

- The run-length method encodes the pair of

- the number of zeros to skip and
- the next nonzero value.

- The zigzag scan of the 8 x 8 matrix makes up a vector of 64 values with a long runs of zeros. For example, the quantized DCT coefficients are scanned
- as [75; -1; 0; -1; 0; 0; -1; 3; 0; 0; 0; 2; 0; 0; ...; 0]
- with one run, two runs, and three runs of zeros in the middle and 52 extra zeros towards the end.
- The run-length method encodes AC coefficients by producing a pair (run length, value), where the run-length is the number of zeros in the run, while the value is the next nonzero coefficient.
- A special pair (0, 0) indicates EOB.
- The result from a run-length encoding of AC coefficients: (0; -1), (1; -1), (2; -1), (0; 3), (3; 2), (0; 0).

Lossless Entropy Coding

- The DC and AC coefficients are further compressed using entropy coding.
- JPEG allows Huffman coding and arithmetic coding.
- We focus on the Huffman coding here

Coding DC Coefficients

- Each DPCM-coded DC coefficient is encoded by a pair of symbols (size, amplitude) with the size (4-bit code) designating the number of bits for the coefficient as shown in next slide, while the amplitude is encoded by the actual bits.
- For the negative number of the amplitude, 1's complement is used.
- For example, we can code the DPCM-coded DC coefficients 200, -10, -10, -20, 10 as
- (8; 11001000); (4; 0101); (4; 0101); (5; 01011); (4; 1010).
- Since there needs to be 4 bits for encoding each size, we can use 45 bits in total for encoding the DC coefficients for these five subblocks.

Huffman Coding Table	
Size	**Amplitude**
1	−1,1
2	−3,−2,2,3
3	−7,..,−4,4,...,7
4	−15,...,−8,8,...,15
5	−31,...,−16,16,...,31
.	.
.	.
.	.
10	−1023,...,−512,512,...,1023

Thank You!

Perceptual Video Processing

by

Dinesh Naik

(B.E, M.Tech, Ph.D)

Assistant Professor, Dept. of Information Technology
National Institute of Technology Karnataka, Surathkal

April 9, 2023

Acknowledgement

- I would like to express my sincere gratitude to Prof. Dr. Abdulmotaleb El Saddik Multimedia Communications Research Laboratory
- I also thank Maaruf Ali, Department of Electronic Engineering, School of Technology, Oxford Brookes University
- I also thank Professors: Li Tan, Purdue University North Central and Jean Jiang, Purdue University North Central.

WAVELET BASICS AND FAMILIES OF WAVELETS

- Wavelet transform has become a powerful tool for signal processing.
- It offers time–frequency analysis to decompose the signal in terms of a family of wavelets or a set of basic functions, which have a fixed shape but can be shifted and dilated in time.
- The wavelet transform can present a signal with a good time resolution or a good frequency resolution.
- There are two types of wavelet transforms: the continuous wavelet transform (CWT) and the discrete wavelet transform (DWT). Specifically, the DWT provides an efficient tool for signal coding. It operates on discrete samples of the signal.
- The DWT resembles other discrete transforms, such as the discrete Fourier transform (DFT) or the discrete cosine transform (DCT).

Let us examine a signal sampled at 1 kHz with 1024×32 (32,678) samples given by

$$x(t) = 0.5\cos(2\pi \times 80t)[u(t) - u(t-8)] + \sin(2\pi \times 180t)[u(t-8) - u(t-16)]$$
$$+\sin(2\pi \times 250t)[u(t-16) - u(t-32)] + 0.1\sin(2\pi \times 0.8t)[u(t-8) - u(t-24)]$$

The signal contains four sinusoids: 80 Hz for $0 \leq t < 8$ seconds, 180 Hz for $8 \leq t < 16$ seconds, 350 Hz for $16 \leq t \leq 32$ seconds, and finally 0.8 Hz for $8 \leq t \leq 24$ seconds. All the signals are plotted

Figure: signal sampled at 1 kHz with 1024×32 (32,678) samples

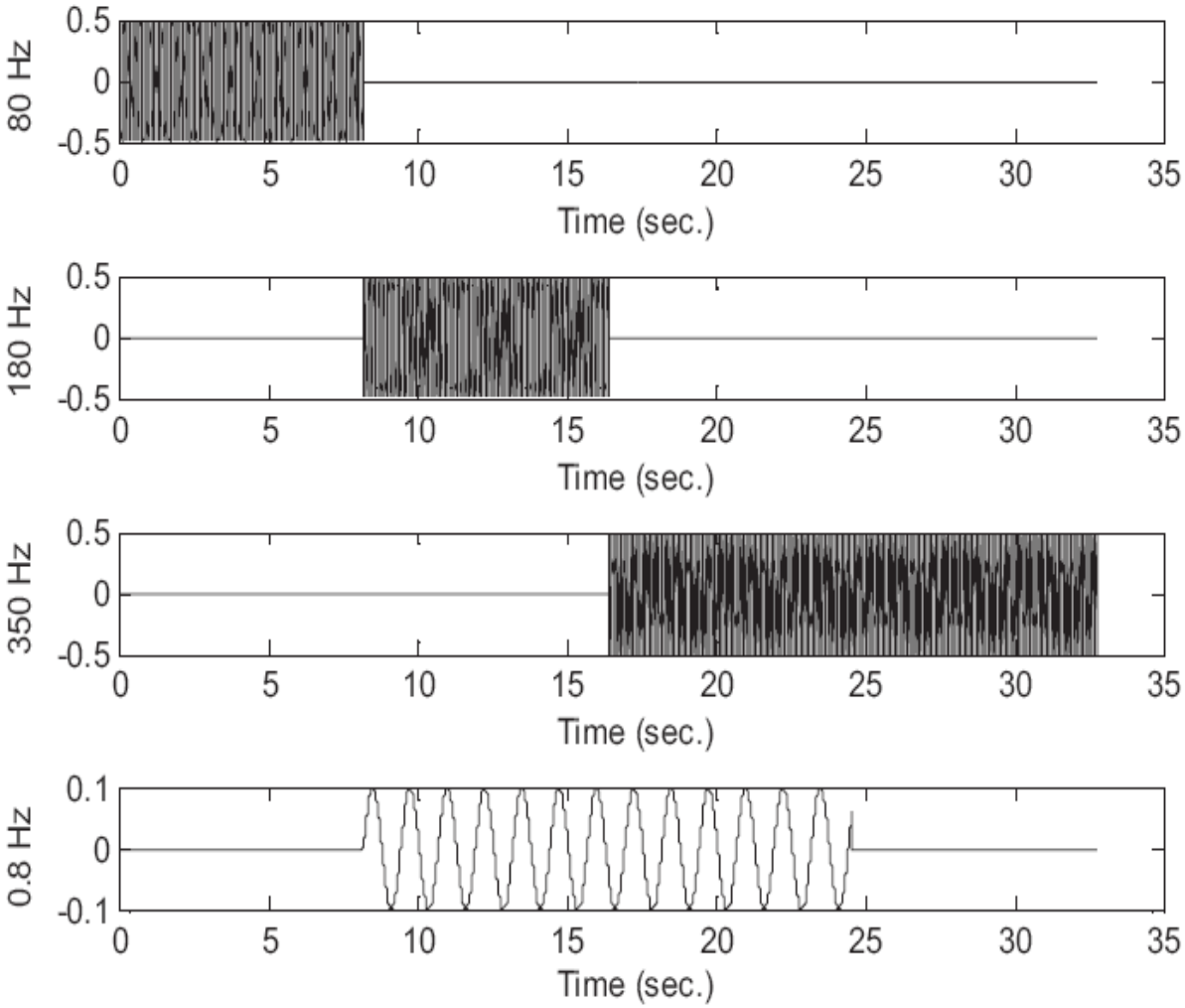


Figure: Individual signal components.

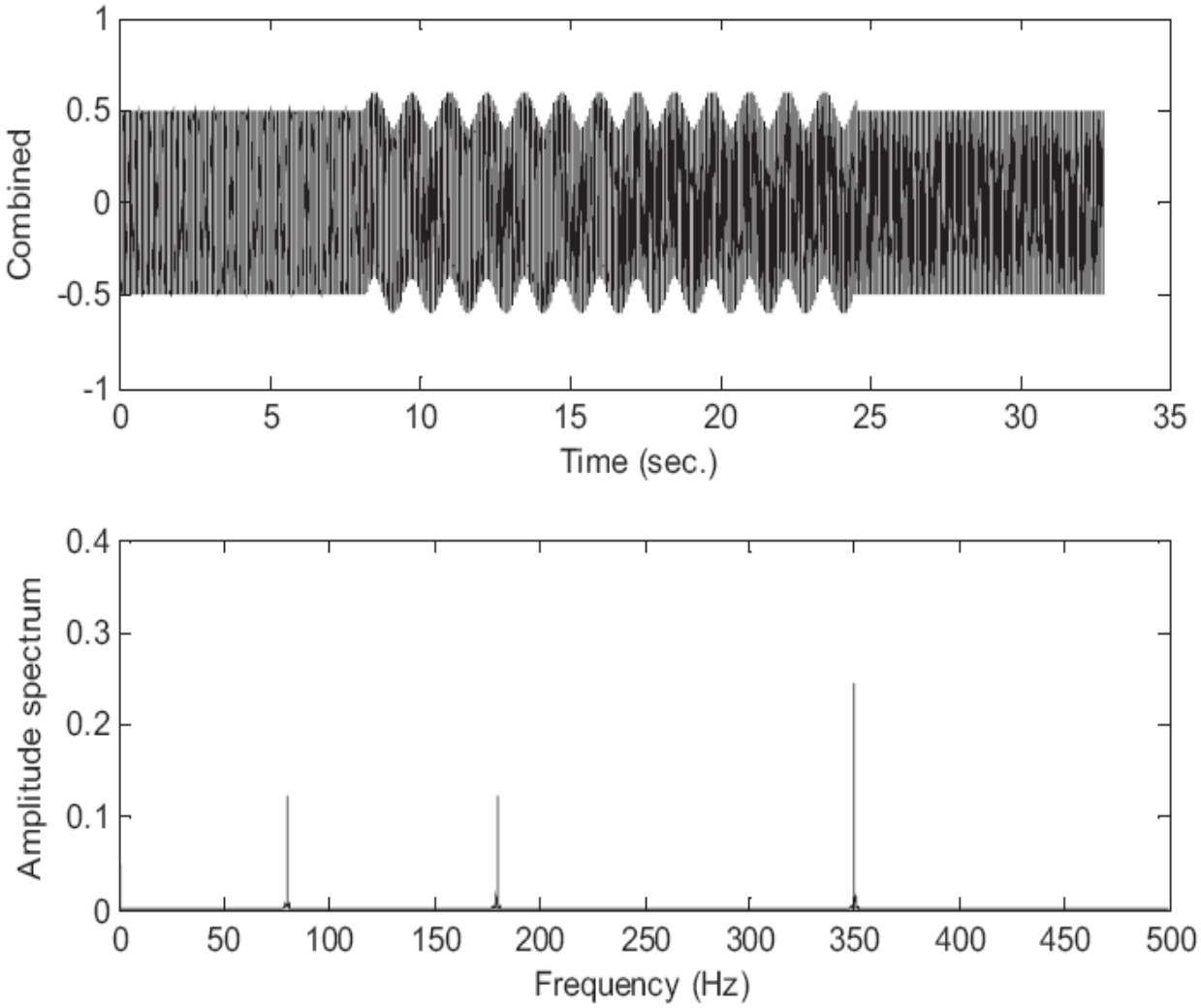


Figure: Combined signal and its spectrum.

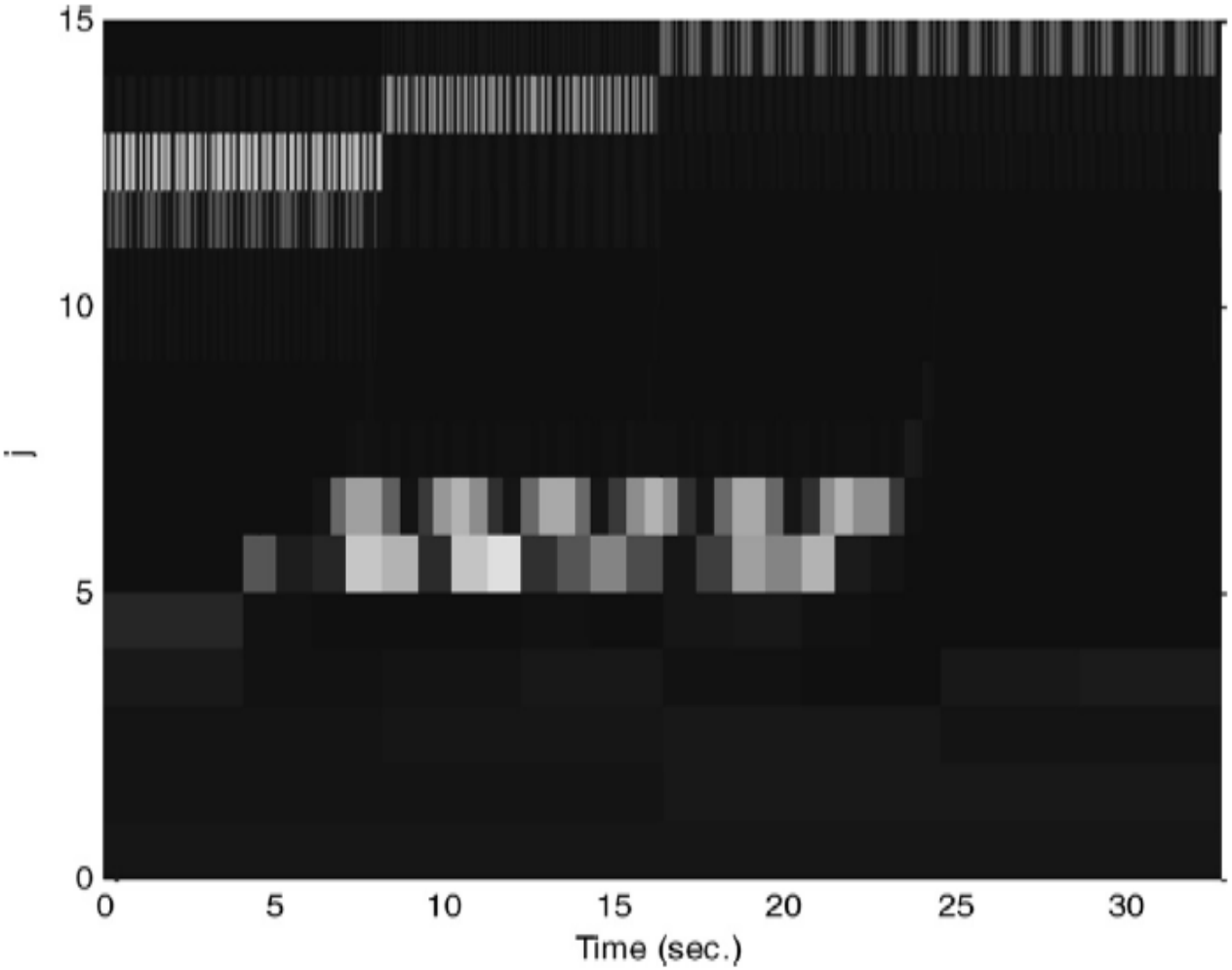


Figure: Wavelet transform amplitudes.

- The horizontal axis is time in seconds while the vertical axis is index j, which is inversely proportional to the scale factor ($a = 2^{-j}$).
- As will be discussed, the larger the scale factor (the smaller the index j), the smaller the frequency value.
- The amplitudes of the wavelet transform are displayed according to the intensity.
- The brighter the intensity, the larger the amplitude. The areas with brighter intensities indicate the strongest resonances between the signal and the wavelets of various frequency scales and time shifts.
- In Figure, the four different frequency components and the discontinuities of the sinusoids are displayed as well.
- We can further observe the fact that the finer the frequency resolution, the coarser the time resolution.
- For example, we can clearly identify the start and stop times for 80-, 180-, and 350-Hz frequency components, but frequency resolution is coarse, since index j has larger frequency spacing. However, for the 0.8-Hz sinusoid, we have fine frequency resolution (small frequency spacing so we can see the 0.8-Hz sinusoid) and coarse time resolution as evidenced by the way in which the start and stop times are blurred.

SUBBAND CODING BASICS

- In many applications such as speech and audio analysis, synthesis, and compression, digital filter banks are often used.
- The filter bank system consists of two stages. The first stage, called the analysis stage, is in the form of filter bank decomposition, in which the signal is filtered into subbands along with a sampling rate decimation.
- The second stage interpolates the decimated subband signals to reconstruct the original signal.
- For the purpose of data compression, spectral information from each subband channel can be used to quantize the subband signal efficiently to achieve efficient coding.

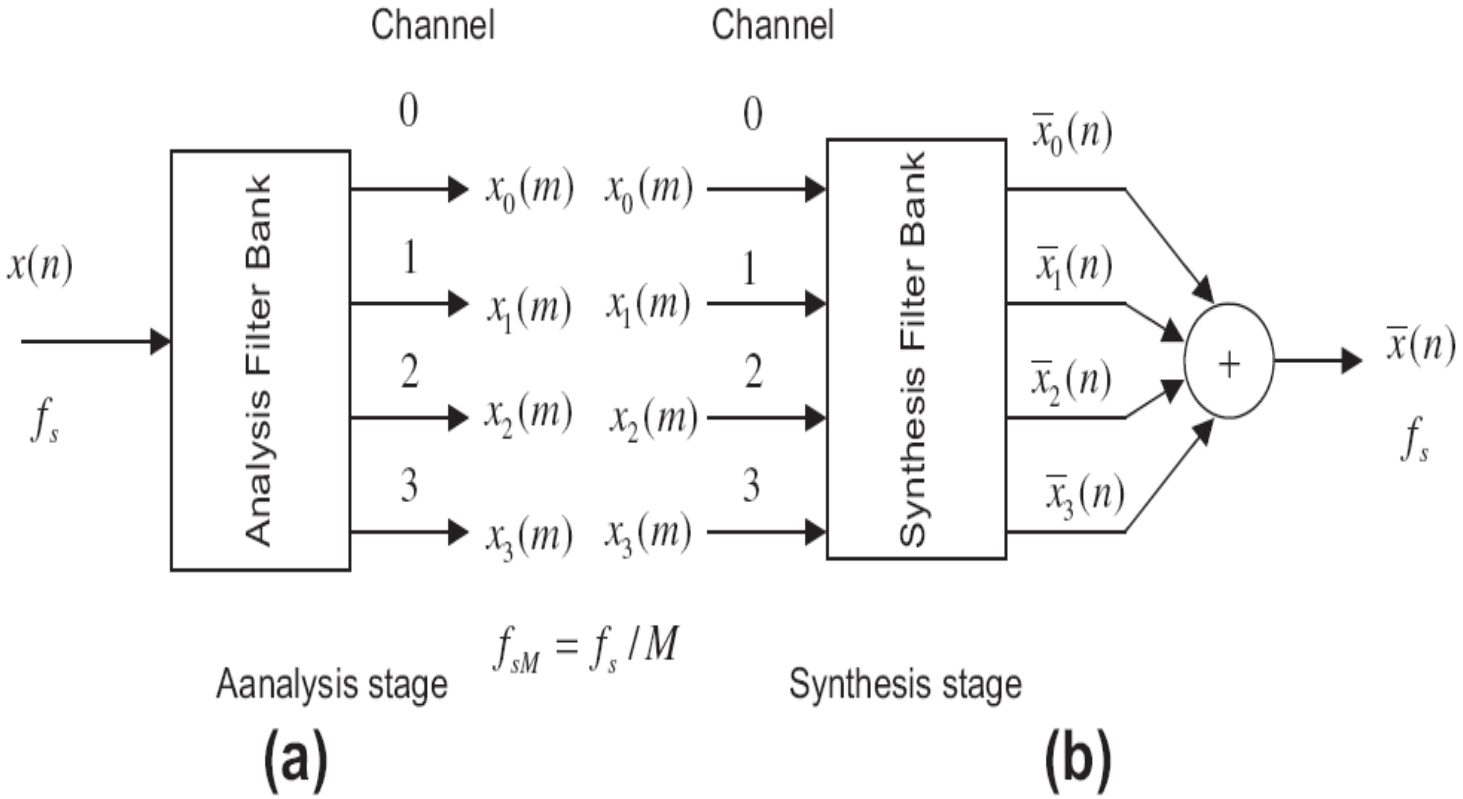

Channel
Channel
0
0
1
1
2
2
3
3
$x(n)$
f_s
Analysis Filter Bank
$x_0(m)$
$x_1(m)$
$x_2(m)$
$x_3(m)$
$x_0(m)$
$x_1(m)$
$x_2(m)$
$x_3(m)$
Synthesis Filter Bank
$\bar{x}_0(n)$
$\bar{x}_1(n)$
$\bar{x}_2(n)$
$\bar{x}_3(n)$
+
$\bar{x}(n)$
f_s
$f_{sM} = f_s / M$
Aanalysis stage
Synthesis stage
(a)
(b)

Figure:

SUBBAND CODING BASICS

- Figure illustrates the basic framework for a four-channel filter bank analyzer and synthesizer.
- At the analysis stage, the input signal x(n) at the original sampling rate f_s is divided via the analysis filter bank into four channels, $x_0(m)$, $x_1(m)$, $x_2(m)$, and $x_3(m)$, each at the decimated sampling rate f_s=M, where M = 4.
- For the synthesizer, these four decimated signals are interpolated via a synthesis filter bank. The outputs from all four channels ($\bar{x}_0(m)$, $\bar{x}_1(m)$, $\bar{x}_2(m)$, and $\bar{x}_0(m)$) of the synthesis filter bank are then combined to reconstruct the original signal $\bar{x}(n)$ at the original sampling rate f_s.

The CWT is defined as

$$W(a,b) = \int_{-\infty}^{\infty} f(t)\psi_{ab}(t)dt$$

where $W(a,b)$ is the wavelet transform and $\psi_{ab}(t)$ is called the mother wavelet, which is defined as

$$\psi_{ab}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right)$$

The wavelet function consists of two important parameters: scaling a and translation b. A scaled version of the function $\psi(t)$ with a scale factor of a is defined as $\psi(t/a)$. Consider a base function $\psi(t) = \cos(\omega t)$ when $a = 1$. When $a > 1$, $\psi(t) = \cos(\omega t/a)$ is a scaled function with a frequency less than ω rad/s. When $a < 1$, $\psi(t) = \cos(\omega t/a)$ has a frequency larger than ω. Figure 1 shows the scaled wavelet functions.

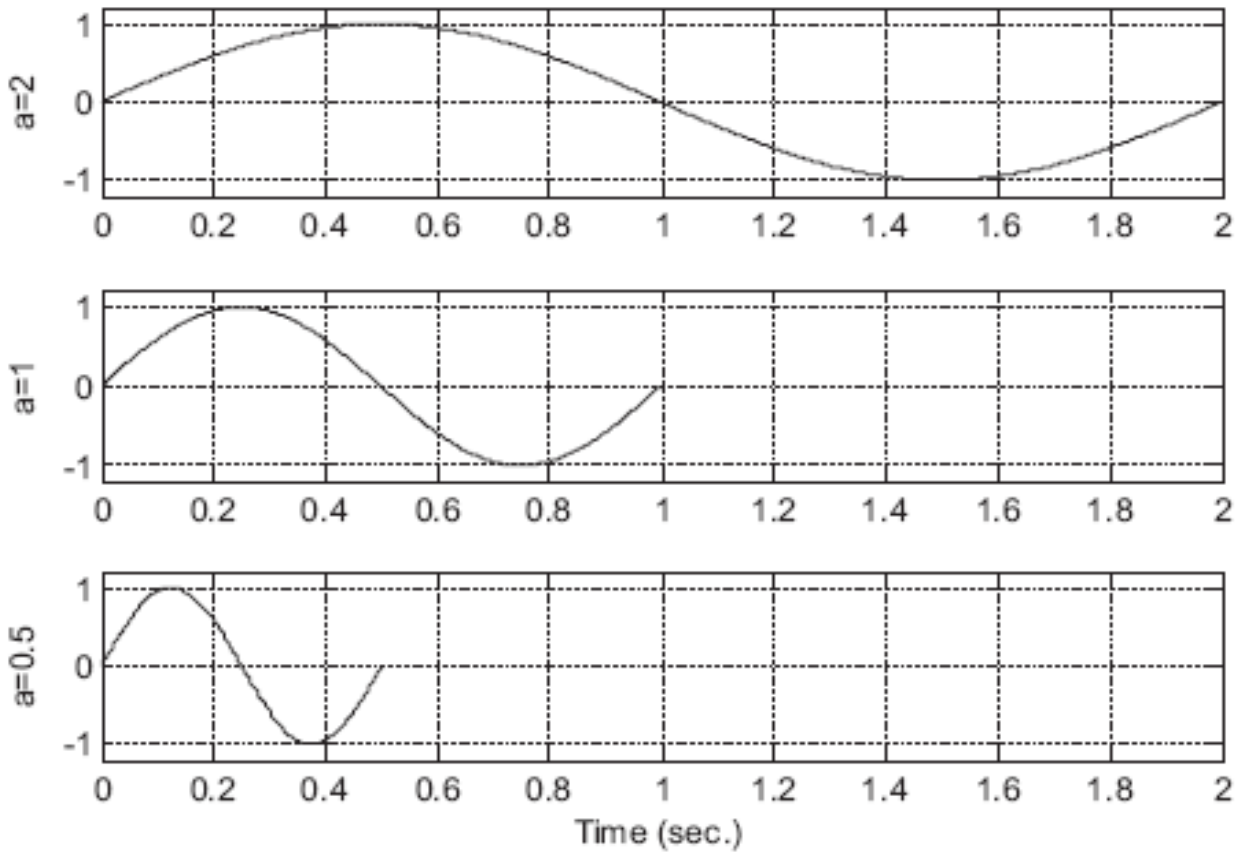


Figure: Scaled wavelet functions.

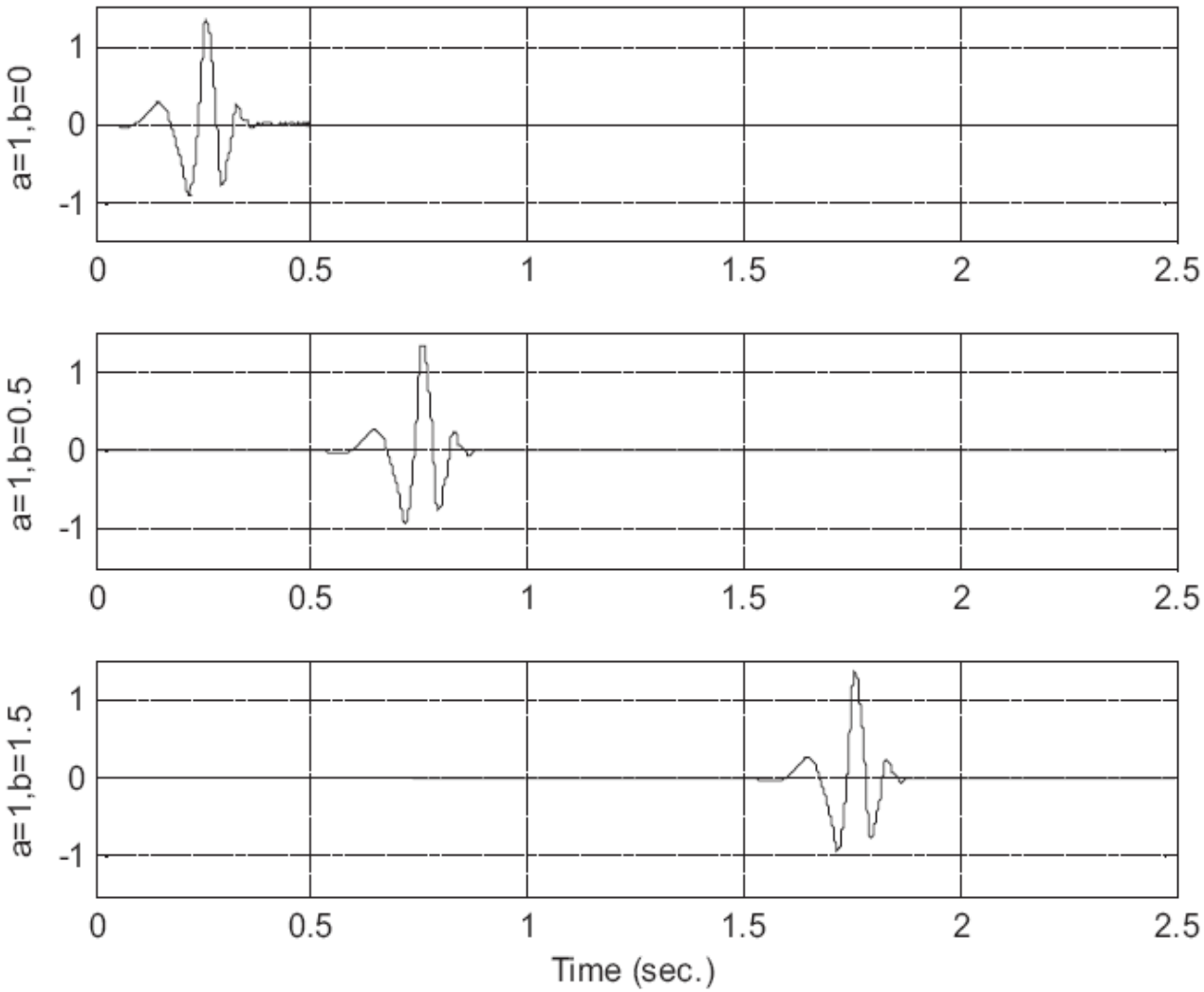


Figure: Translated wavelet functions.

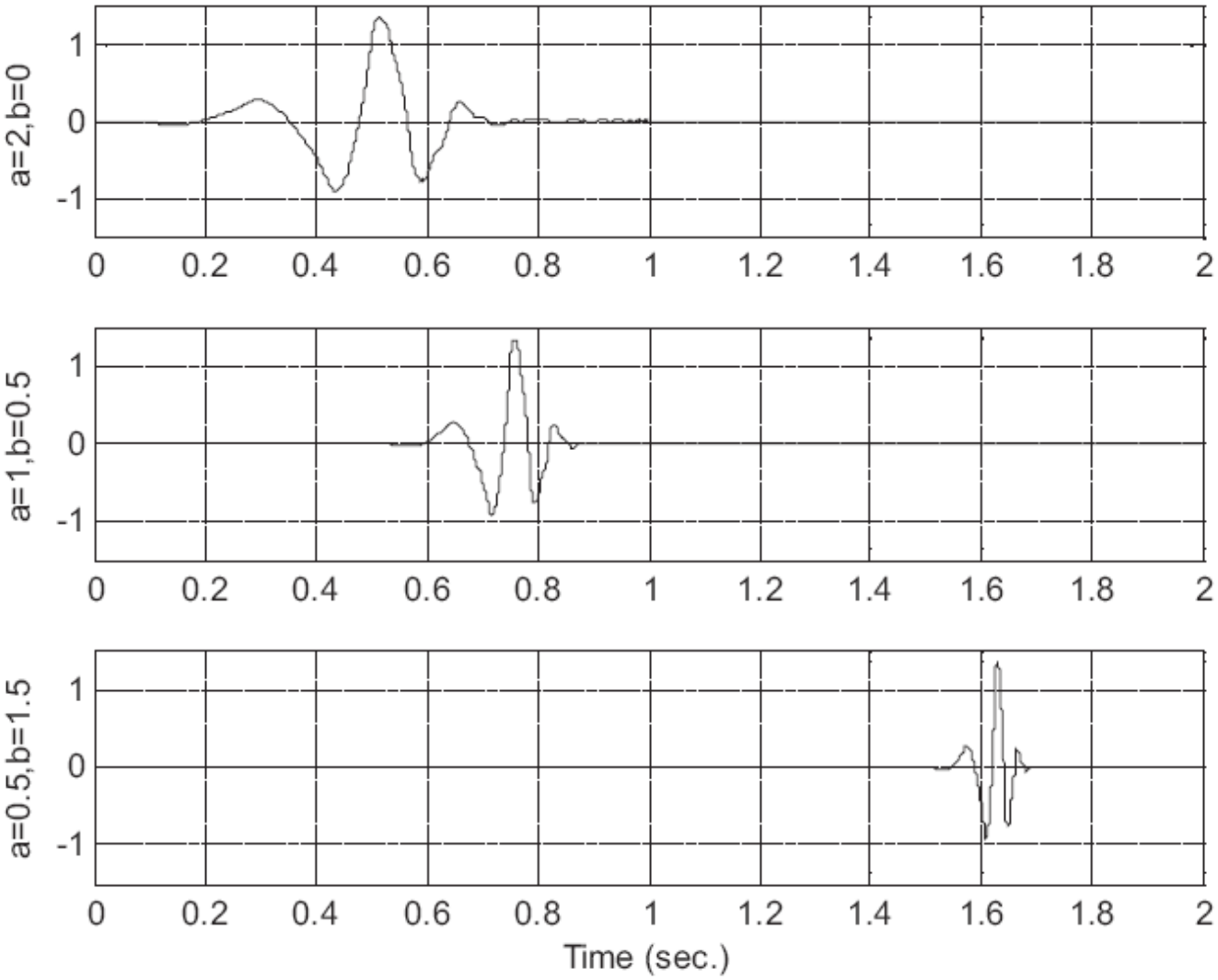


Figure: Scaled and Translated wavelet functions.

A translated version of the function $\psi(t)$ with a shifted time constant b is defined as $\psi(t-b)$. Figure shows several translated versions of the wavelet. A scaled and translated function $\psi(t)$ is given by $\psi((t-b)/a)$. This means that $\psi((t-b)/a)$ changes frequency and time shift. Several combined scaling and translated wavelets are displayed in Figure

To devise an efficient wavelet transform algorithm, we let the scale factor be a power of two, that is,

$$a = 2^{-j} \tag{13.33}$$

Note that the larger the index j, the smaller the scale factor $a = 2^{-j}$. The time shift becomes

$$b = k2^{-j} = ka \tag{13.34}$$

Substituting Equations (13.33) and (13.34) into the base function gives

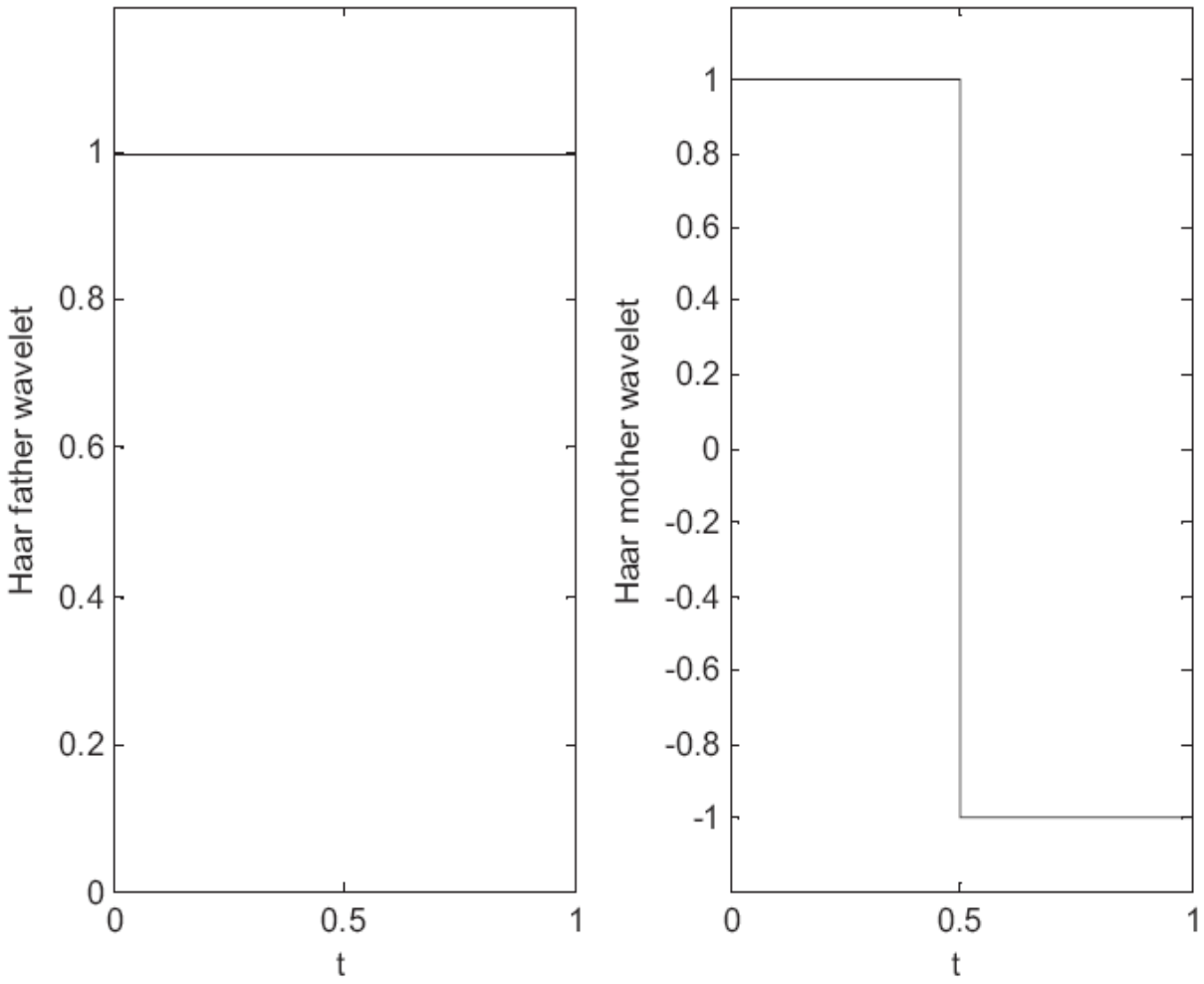
$$\psi\left(\frac{t-b}{a}\right) = \psi\left(\frac{t-kb}{a}\right) = \psi(a^{-1}t-k) = \psi(2^j t-k) \tag{13.35}$$

We can define a mother wavelet at scale j and translation k as

$$\psi_{jk}(t) = 2^{j/2}\psi(2^j t-k) \tag{13.36}$$

Similarly, a father wavelet (scaling function) at scale j and translation k is defined as

$$\phi_{jk}(t) = 2^{j/2}\phi(2^j t-k) \tag{13.37}$$



(a) Father wavelet $\phi(t)$ **(b)** Mother wavelet $\psi(t)$

Thank You!

Perceptual Video Processing

by
Dinesh Naik
(B.E, M.Tech, Ph.D)

Assistant Professor, Dept. of Information Technology
National Institute of Technology Karnataka, Surathkal

April 9, 2023

Acknowledgement

- I would like to express my sincere gratitude to Prof. Dr. Abdulmotaleb El Saddik Multimedia Communications Research Laboratory
- I also thank Maaruf Ali, Department of Electronic Engineering, School of Technology, Oxford Brookes University
- I also thank Professors: Li Tan, Purdue University North Central and Jean Jiang, Purdue University North Central.
- I thank V. Sanchez A. Basu, University of Alberta

Part I: Image Compression

Compression techniques are used to reduce the redundant information in the image data in order to facilitate the storage, transmission and distribution of images (e.g. GIF, TIFF, PNG, JPEG)

Limitations of JPEG Standard

- **Low bit-rate compression**: JPEG offers an excellent quality at high and mid bit-rates. However, the quality is unacceptable at low bit-rates (e.g. below 0.25 bpp)

- **Lossless and lossy compression**: JPEG cannot provide a superior performance at lossless and lossy compression in a single code-stream.

- **Transmission in noisy environments**: the current JPEG standard provides some resynchronization markers, but the quality still degrades when bit-errors are encountered.

- **Different types of still images**: JPEG was optimized for natural images. Its performance on computer generated images and bi-level (text) images is poor.

What is JPEG2000?

- JPEG2000 is a new compression standard for still images intended to overcome the shortcomings of the existing JPEG standard.

- The standardization process is coordinated by the Joint Technical Committee on Information technology of the International Organization for Standardization (ISO)/ International Electrotechnical Commission (IEC).

- JPEG2000 makes use of the wavelet and sub-band technologies. Some of the markets targeted by the JPEG2000 standard are Internet, printing, digital photography, remote sensing, mobile, digital libraries and E-commerce.

- The core compression algorithm is primarily based on the Embedded Block Coding with Optimized Truncation (EBCOT) of the bit-stream. The EBCOT algorithm provides a superior compression performance and produces a bit-stream with features such as resolution and SNR scalability and random access.

Features of JPEG2000

- **Lossless and lossy compression**: the standard provides lossy compression with a superior performance at low bit-rates. It also provides lossless compression with progressive decoding. Applications such as digital libraries/databases and medical imagery can benefit from this feature.

- **Protective image security**: the open architecture of the JPEG2000 standard makes easy the use of protection techniques of digital images such as watermarking, labeling, stamping or encryption

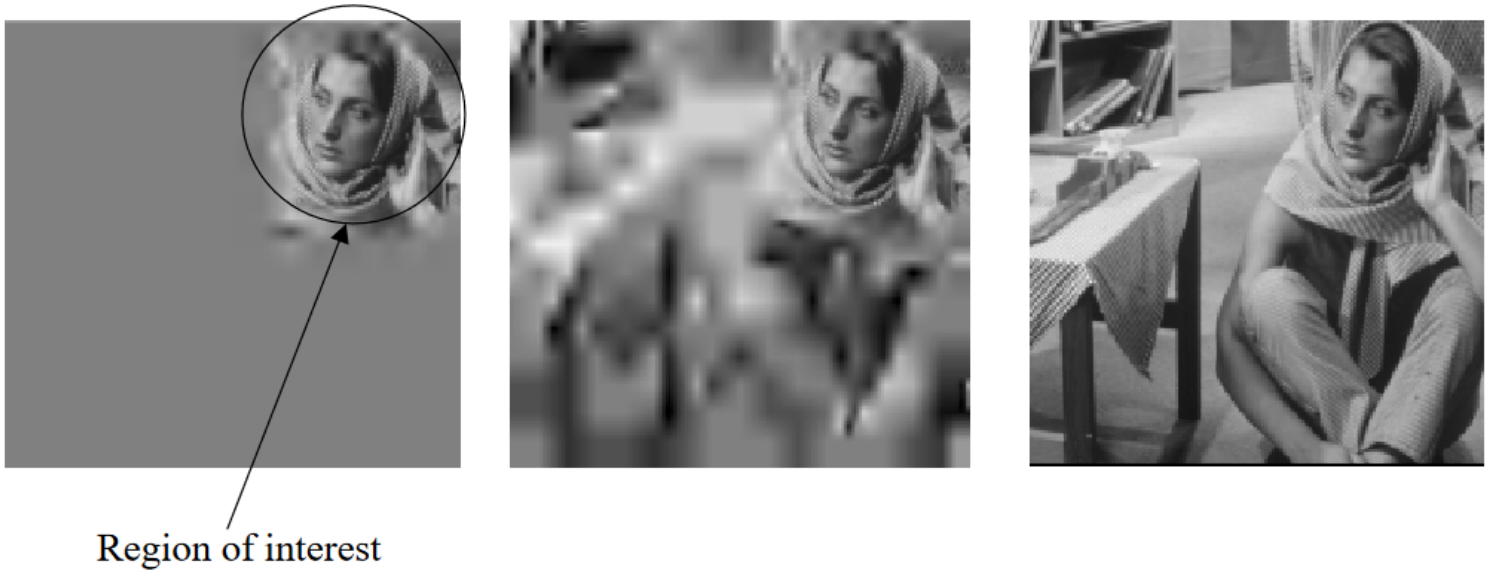
- **Region-of-interest coding**: in this mode, regions of interest (ROI's) can be defined. These ROI's can be encoded and transmitted with better quality than the rest of the image .

- **Robustness to bit errors**: the standard incorporate a set of error resilient tools to make the bit-stream more robust to transmission errors.

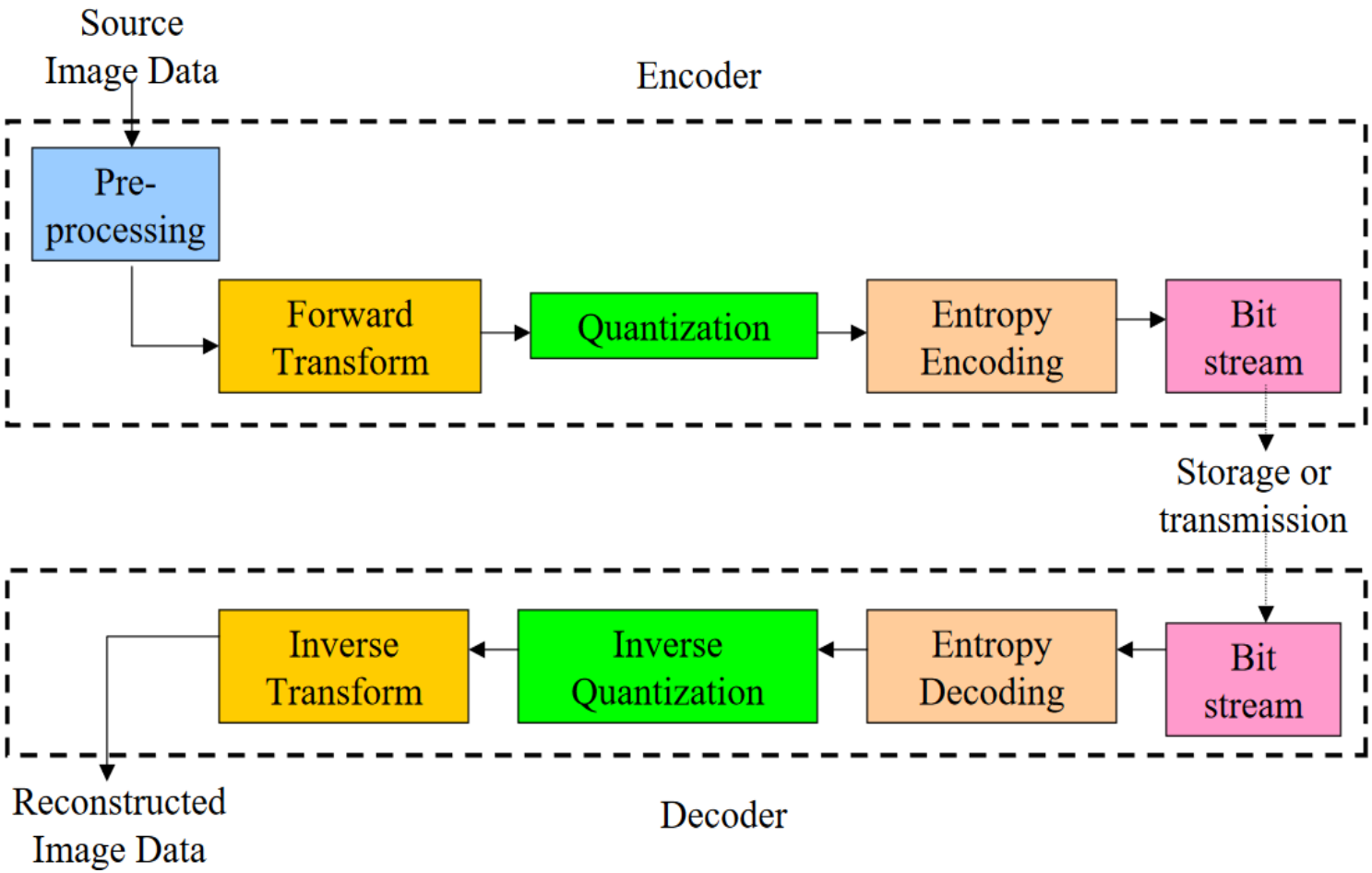
Features of JPEG2000 (cont'd)

Example of region of interest coding.

A **region of interest** in the *Barbara* image is reconstructed with quality scalability. The region of interest is decoded first before any background information.



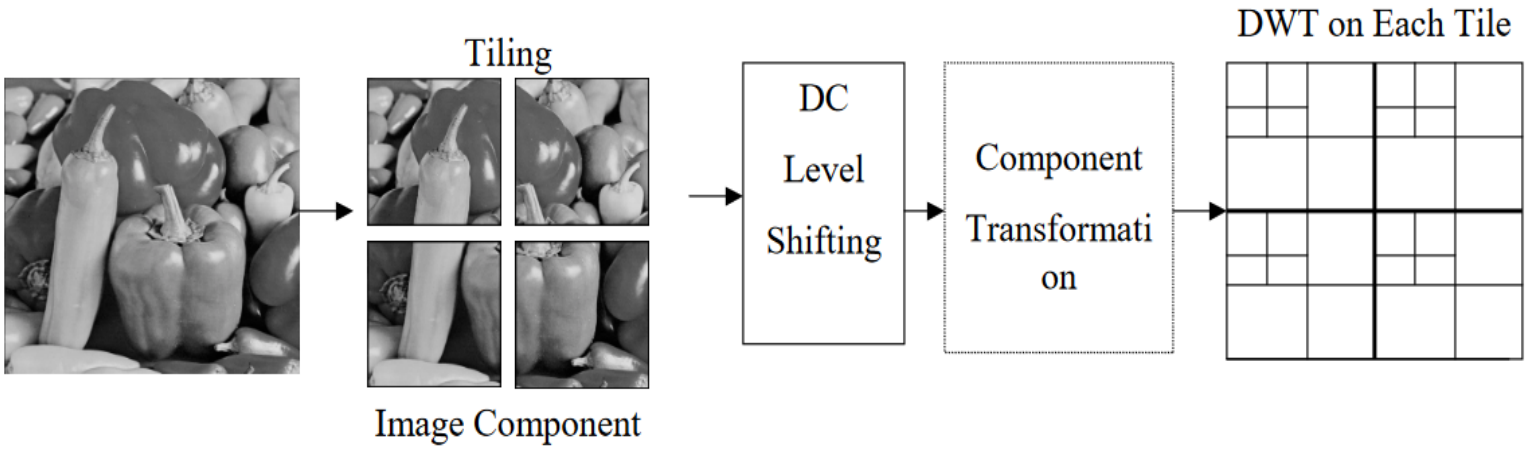
The JPEG 2000 Compression Engine



Pre-processing

Three steps:

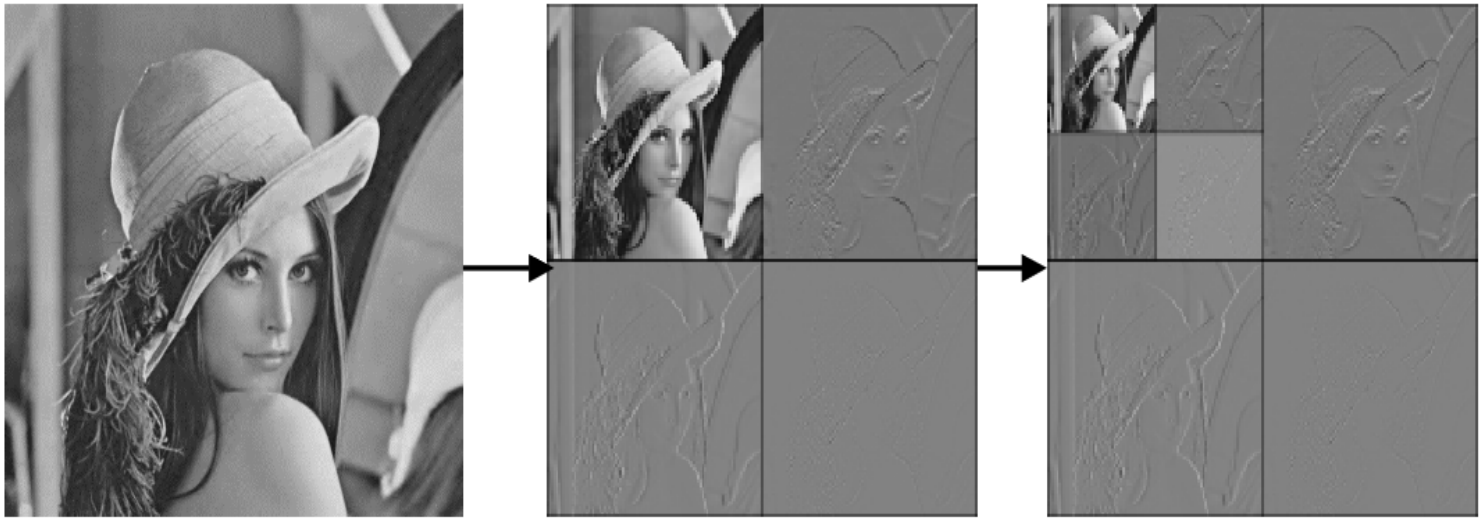
1. Image tiling (optional)→ for each image component.



- DC level shifting → samples of each tile are subtracted the same quantity (i.e. component depth).
- Color transformation (optional) → from RGB to Y Cb Cr

Forward Transform

- Discrete Wavelet Transform (DWT) is used to decompose each tile component into different sub-bands.
- The transform is in the form of dyadic decomposition and use bi-orthogonal wavelets.

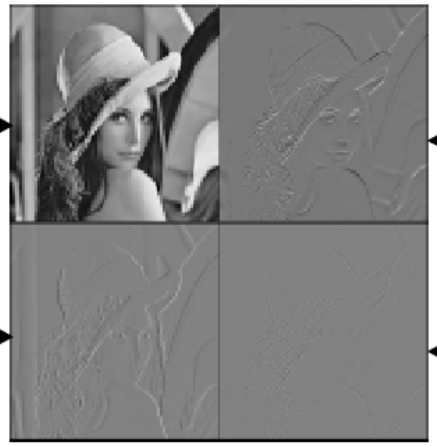


Forward Transform (cont'd)

- DWT can be irreversible or reversible.
  - Irreversible transform →Daubechies 9-tap/7-tap filter
  - Reversible transform → Le Gall 5-tap/3-tap filter
- Two filtering modes are supported:
  - Convolution based
  - Lifting based

2D-Forward Transform

- 1-D sets of samples are decomposed into low-pass and high-pass samples.
- Low-pass samples represent a down-sampled, low resolution version of the original set.
- High pass samples represent a down-sampled residual version of the original set (details).



Quantization

- After transformation, all coefficients are quantized using scalar quatization.
- Quantization reduces coefficients in precision. The operation is lossy unless the quatization step is 1 and the coefficients integers (e.g. reversible integer 5/3 wavelet).
- The process follows the formula:

$$q_b(u,v) = sign(a_b(u,v))\left\lfloor \frac{|a_b(u,v)|}{\Delta_b} \right\rfloor$$

Largest integer not exceeding a_b

Quantized value

Transform coefficient of sub-band b

Quantization step

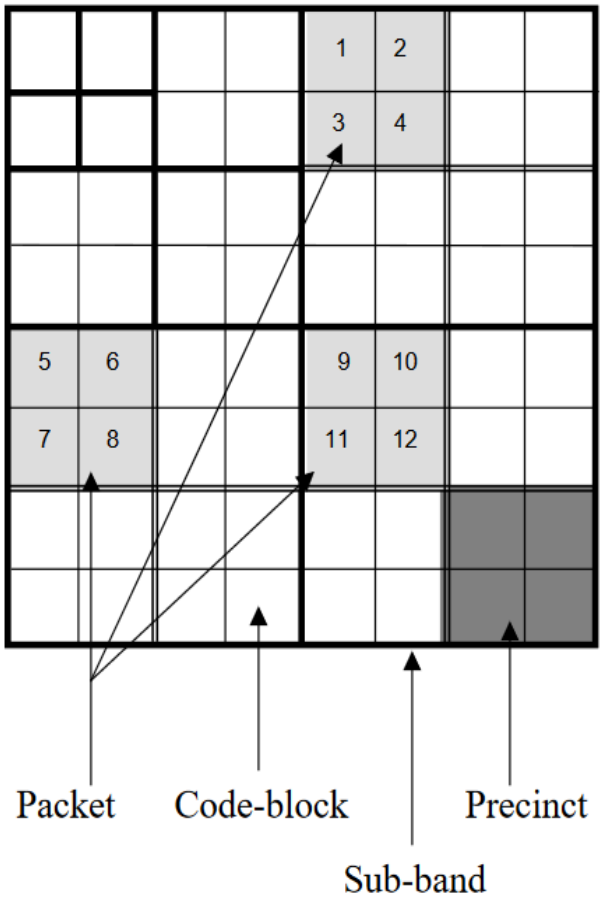
Modes of Quantization

- Two modes of operation:

  - **Integer mode**→ integer-to-integer transforms are employed. Quantization step are fixed to one. Lossy coding is still achieved by discarding bit-planes.

  - **Real mode**→ real-to-real transforms are employed. Quantization steps are chosen in conjunction with rate control. In this mode, lossy compression is achieved by discarding bi-planes or changing the size of the quantization step or both.

Code-blocks, precincts and packets



Precinct: each sub-band is divided into rectangular blocks called precincts.

Packets: three spatially consistent rectangles comprise a packet.

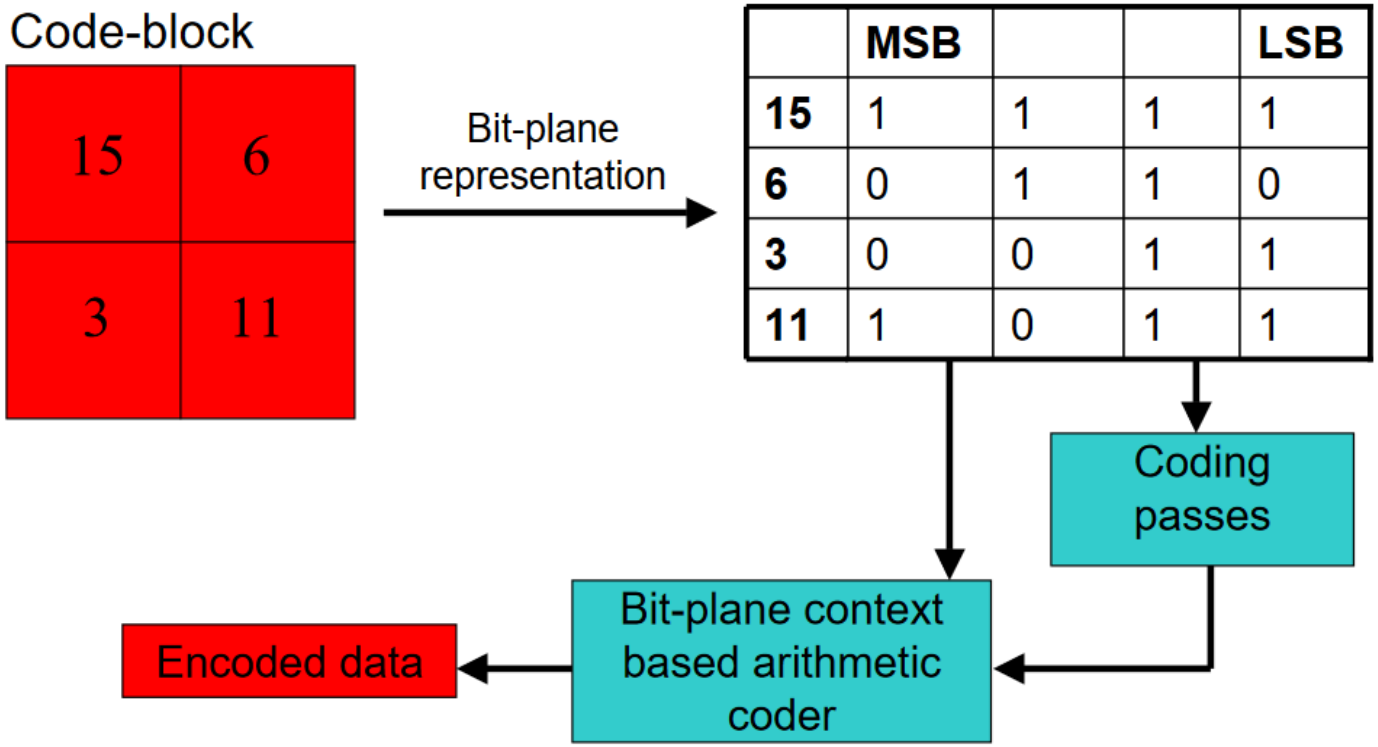
Code-block: each precinct is further divided into non-overlapping rectangles called code-blocks.

Each code-block forms the input to the entropy encoder and is encoded independently.

Within a packet, code-blocks are visited in raster order.

Entropy Coding: Bit-planes

• The coefficients in a code block are separated into bit-planes. The individual bit-planes are coded in 1-3 coding passes.



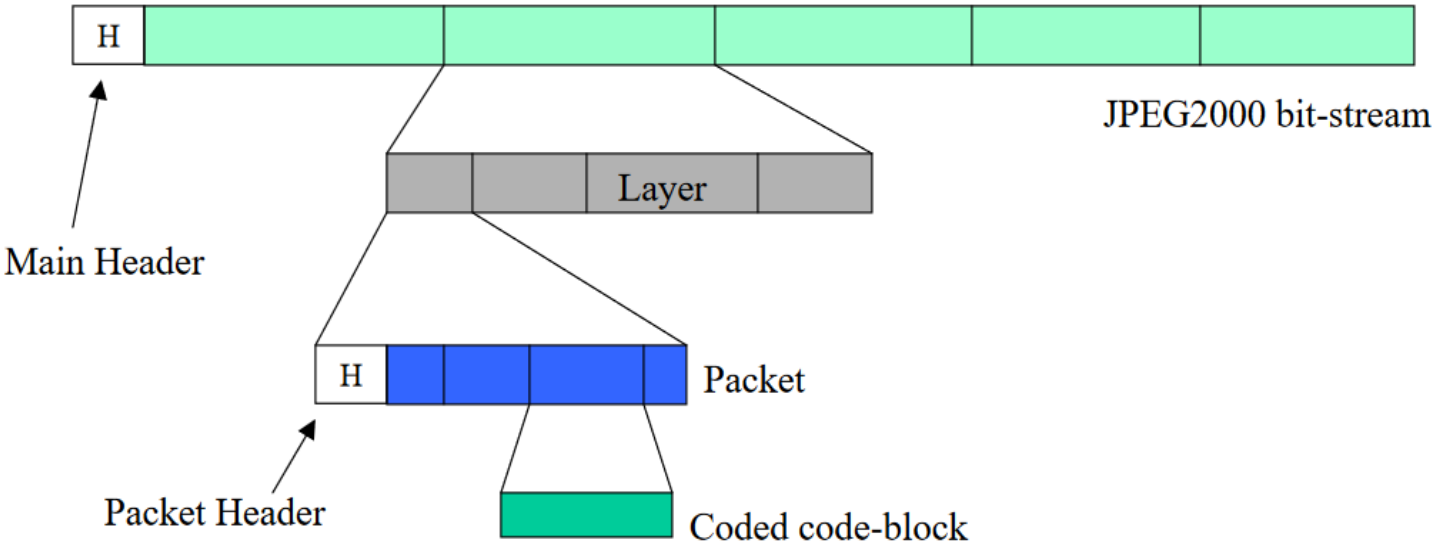
Entropy Coding: Coding Passes

- Each of these coding passes collects contextual information about the bit-plane data. The contextual information along with the bit-planes are used by the arithmetic encoder to generate the compressed bit-stream.

- The coding passes are:
  - **Significance propagation pass** → coefficients that are insignificant and have a certain preferred neighborhood are coded.
  - **Magnitude refinement pass** → the current bits of significant coefficients are coded.
  - **Clean-up pass** → the remaining insignificant coefficients for which no information has yet been coded are coded.

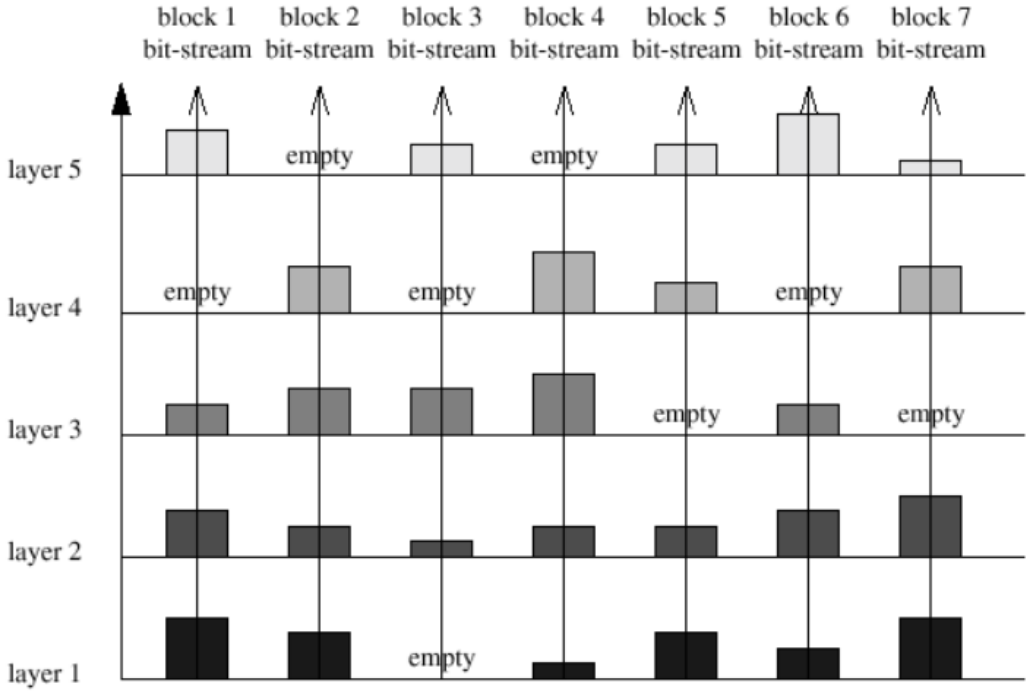
JPEG2000 Bit-stream

- For each code-block, a separate bit-stream is generated.
- The coded data of each code-block is included in a packet.
- If more that one layer is used to encode the image information, the code-block bit-streams are distributed across different packets corresponding to different layers.



Layers of the JPEG2000 Bit-stream

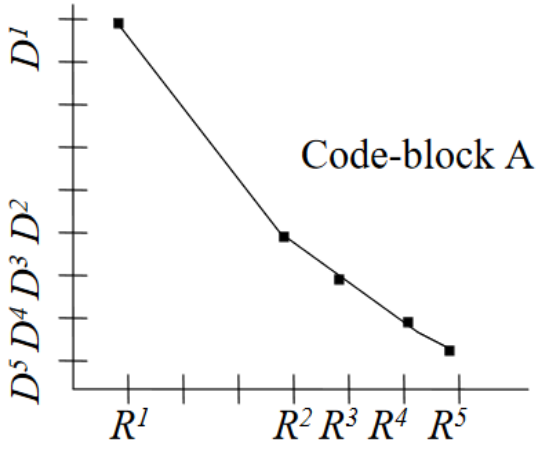
- Therefore, each layer consists of a number of consecutive bit-plane coding passes from each code-block in the tile, including all sub-bands of all components for that tile.

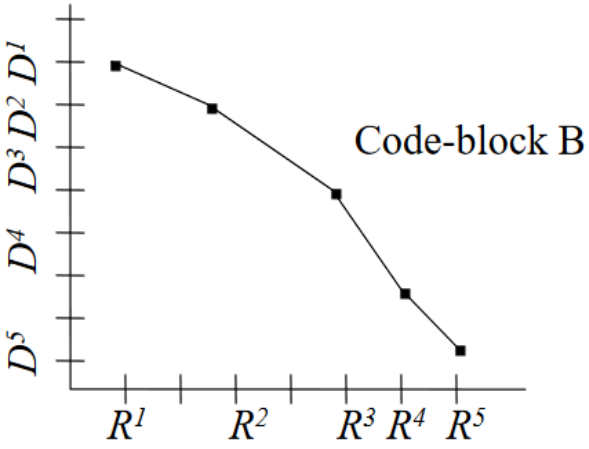


Layer Formation

- The individual code-block streams have the property that they can be truncated to a variety of discrete lengths $R^1, R^2, R^3 \ldots R^n$.
- The distortion incurred when reconstructing from each of these truncated subsets is estimated and denoted by $D^1, D^2, D^3 \ldots D^n$. The Mean Squared Error distortion metric is generally used.
- The first, lowest quality layer, is formed from the optimally truncated code-block bit-streams.

Example: two individual code-block streams

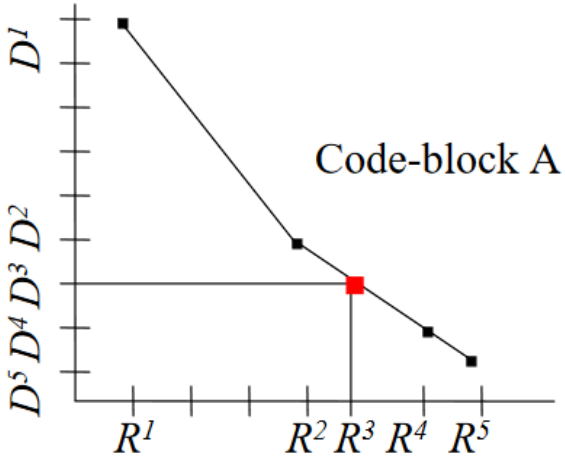


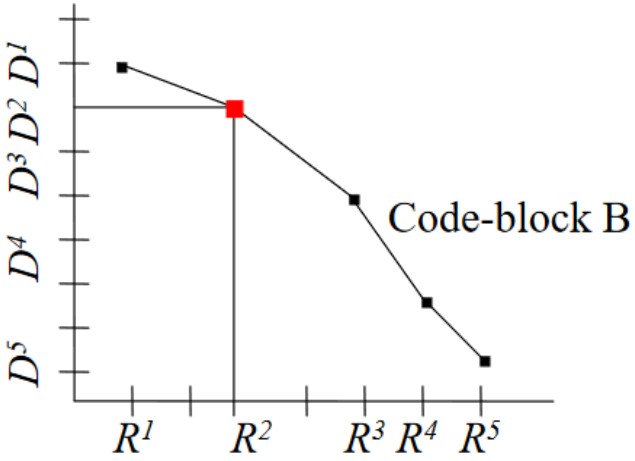


Code-block contributions

• Each subsequent layer is formed by optimally truncating the code-block bit-streams to achieve successively higher target bit-rates, distortion bounds or other quality metrics, as appropriate, and including the additional code words required to augment the information represented in previous layers to the new truncation points.

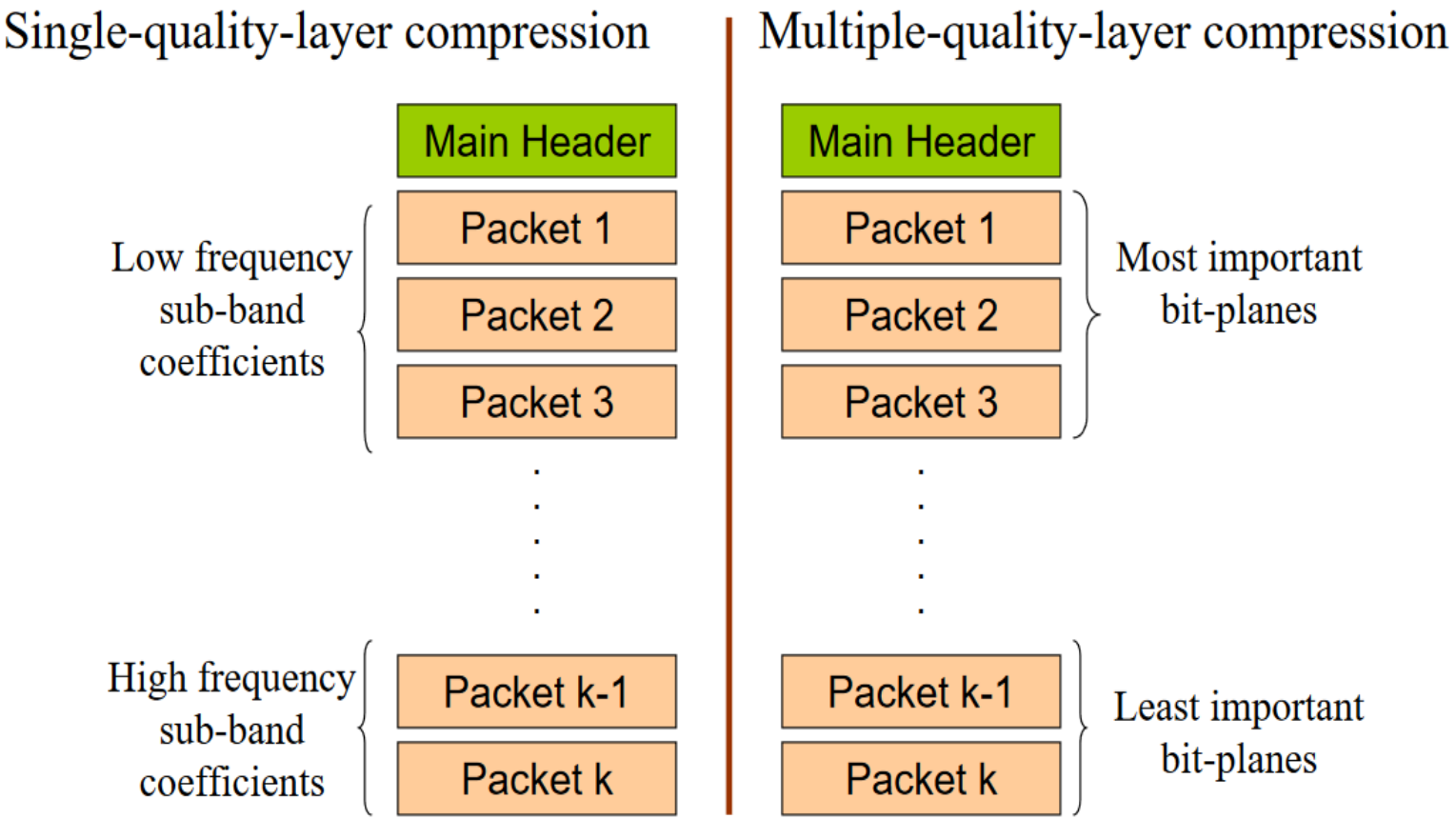
Example:





If target rate for layer 1 is R_{L1} and $R_A^3 + R_B^2 \approx R_{L1}$ with a minimum distortion; then, code-block A is truncated to R^3 and code-block B is truncated to R^2 for layer 1.

Hierarchical structure of the JPEG2000 Bit-stream



Quality Scalability

By interleaving the packets in different orders, four possible progression orders can be achieved in JPEG2000:

- Quality
- Resolution
- Spatial location
- Component

The *Barbara* image is decompressed at different **qualities**.

0.125 bpp

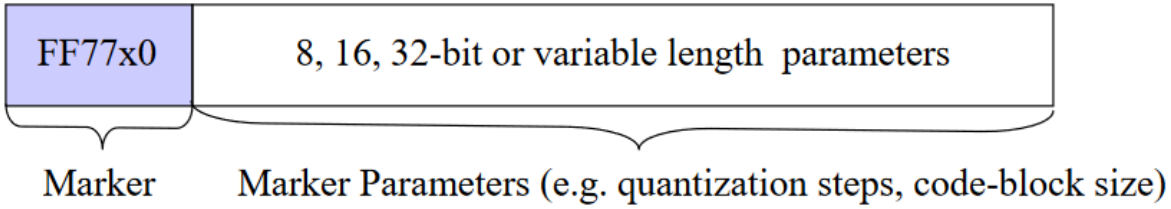
0.25 bpp

0.5 bpp

The *Barbara* image is reconstructed at three **resolutions**.

Markers and Markers Segments

- A main header is appended to the final bit stream. This header is different from the header found in front of every packet and every tile.
- The headers from all the packets and tiles can be placed on the main header → useful if all the header information is to be separated from the compressed data.
- The main header consist of markers and markers segments.
- A marker is a known word (16 bits) that identifies the information contained in the marker segment.



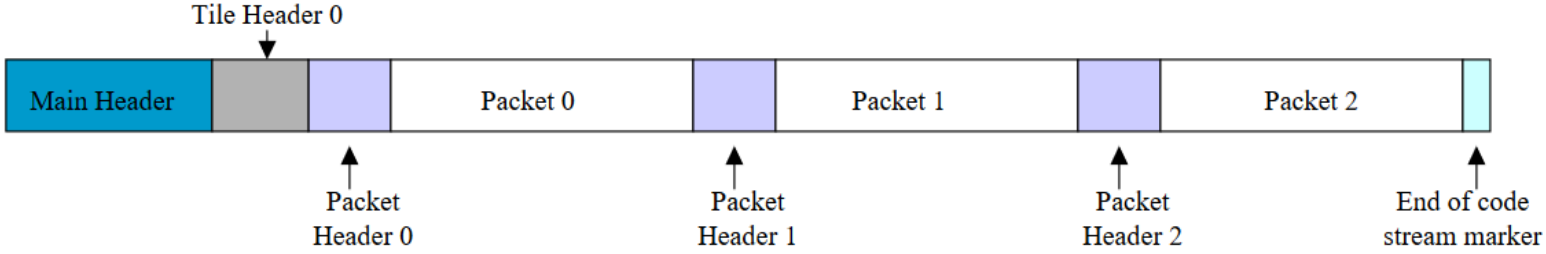
Marker Segment

Packet Headers

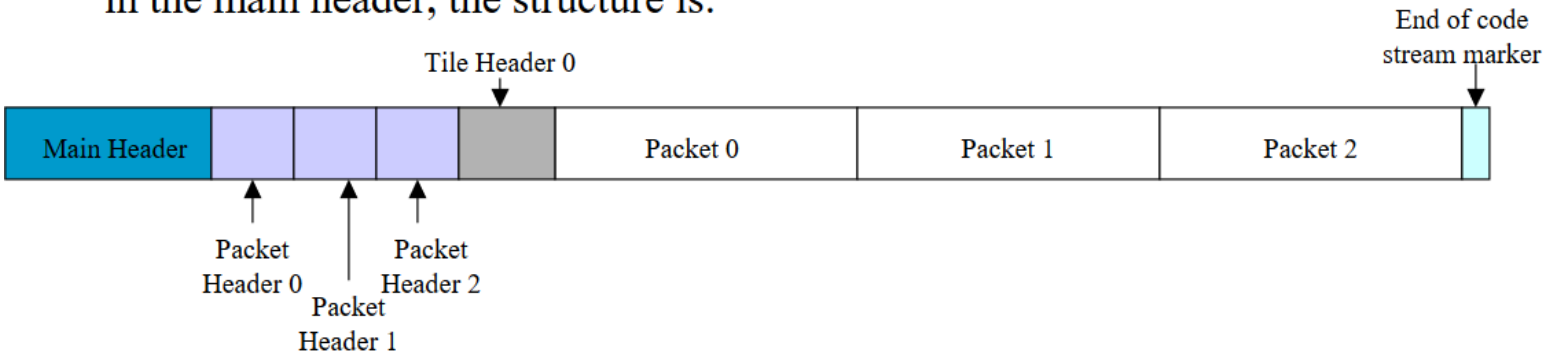
- A header is placed in front of every packet and contains the following information :
  - **Zero length packet** →indicates whether the packet has a length of zero or not.
  - **Code-block inclusion** →indicates whether a code block has been included in the packet or not.
  - **Zero-bit plane information** →if a code-block is included for the first time, the packet header contains information identifying the actual number of bit-planes used to represent coefficients from the code-block.
  - **Number of coding passes** →the number of coding passes included in this packet from each code-block.
  - **Length of data from a given code block** →the packet header identifies the number of bytes contributed by each included code-block.
- The information in the packet header is variable length encoded.

Sample JPEG2000 bit-stream

- For a 512 x 512 gray-level image compressed with one tile, code-block size 64 x 64, precinct size 512 x 512, 3 levels of decomposition and one layer, the general structure of the bit stream is as follows

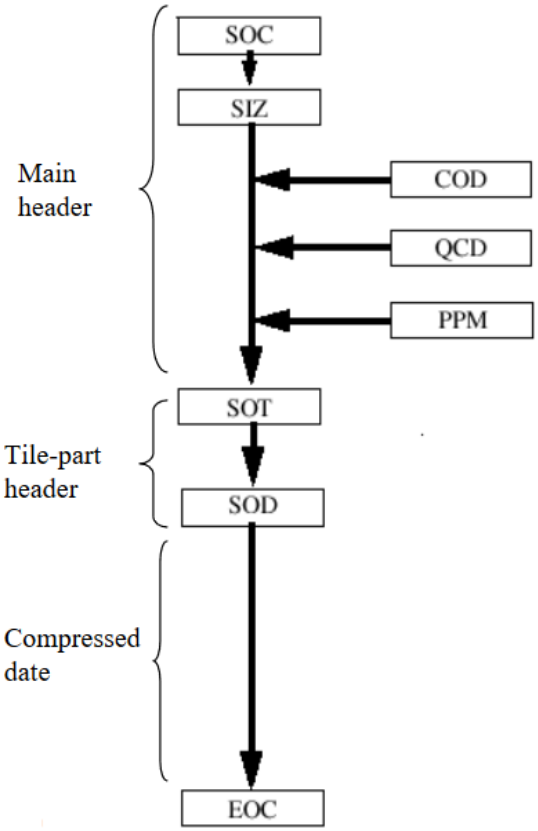


- If all the packet headers are grouped together in a single header and placed in the main header, the structure is:



Markers in a JPEG2000 bit-stream

- The main header would include the following markers and marker segments:



SOC: Start of Code Stream marker

SIZ: Image and Tile Size marker

The SIZ marker segment provides information about the size of the uncompressed image.

COD: Coding Style Default marker

The COD marker segment provides information about the coding style, decomposition and layering used for all the components of the image

QCD: Quantization Default marker

The QCD marker segment provides information about the quantization used for all the components of the image

PPM: Packed Packet Headers, main header

The PPM marker segment is the collection of packet headers.

SOT: Start of Tile-part marker

The SOT marker segment marks the beginning of a tile-part

SOD: Start of Data marker

EOC: End of Code Stream marker

Thank You!

Lecture 17: Huffman Coding

CLRS- 16.3

Outline of this Lecture

- Codes and Compression.
- Huffman coding.
- Correctness of the Huffman coding algorithm.

Suppose that we have a $100,000$ character data file that we wish to store . The file contains only 6 characters, appearing with the following frequencies:

	a	b	c	d	e	f
Frequency in '000s	45	13	12	16	9	5

A *binary code* encodes each character as a binary string or *codeword*. We would like to find a binary code that encodes the file using as few bits as possible, ie., *compresses it* as much as possible.

In a *fixed-length code* each codeword has the same length. In a *variable-length code* codewords may have different lengths. Here are examples of fixed and variable legth codes for our problem (note that a fixed-length code must have at least 3 bits per codeword).

	a	b	c	d	e	f
Freq in '000s	45	13	12	16	9	5
a fixed-length	000	001	010	011	100	101
a variable-length	0	101	100	111	1101	1100

The fixed length-code requires $300,000$ bits to store the file. The variable-length code uses only

$(5{\cdot}1+13{\cdot}3+12{\cdot}3+16{\cdot}3+9{\cdot}4+5{\cdot}4){\cdot}1000 = 224,000$ bits,

saving a lot of space! Can we do better?

Note: In what follows a *code* will be a set of codewords, e.g.,

$\{000, 001, 010, 011, 100, 101\}$

and $\{0, 101, 100, 111, 1101, 1100\}$

Encoding

Given a code (corresponding to some alphabet Γ) and a message it is easy to *encode* the message. Just replace the characters by the codewords.

Example: $\Gamma = \{a, b, c, d\}$
If the code is

$$C_1\{a = 00,\ b = 01,\ c = 10,\ d = 11\}.$$

then bad is encoded into 010011

If the code is

$$C_2 = \{a = 0,\ b = 110,\ c = 10,\ d = 111\}$$

then bad is encoded into 1101111

Decoding

$$
\begin{aligned}
C_1 &= \{a = 00,\ b = 01,\ c = 10,\ d = 11\}. \\
C_2 &= \{a = 0,\ b = 110,\ c = 10,\ d = 111\}. \\
C_3 &= \{a = 1,\ b = 110,\ c = 10,\ d = 111\}
\end{aligned}
$$

Given an encoded message, *decoding* is the process of turning it back into the original message.
A message is *uniquely decodable* (vis-a-vis a particular code) if it can only be decoded in one way.

For example relative to C_1, 010011 is uniquely decodable to bad.
Relative to C_2 1101111 is uniquely decodable to bad.
But, relative to C_3, 1101111 is not uniquely decipherable since it could have encoded either bad or acad.

In fact, one can show that every message encoded using C_1 and C_2 are uniquely decipherable. The unique decipherability property is needed in order for a code to be useful.

Prefix-Codes

Fixed-length codes are always uniquely decipherable (why).
We saw before that these do not always give the best compression so we prefer to use variable length codes.

Prefix Code: A code is called a prefix (free) code if no codeword is a prefix of another one.

Example: $\{a = 0,\ b = 110,\ c = 10,\ d = 111\}$ is a prefix code.

Important Fact: Every message encoded by a prefix free code is uniquely decipherable. Since no codeword is a prefix of any other we can always find the first codeword in a message, peel it off, and continue decoding. Example:

$$01101100 = 01101100 = abba$$

We are therefore interested in finding good (best compression) prefix-free codes.

Fixed-Length versus Variable-Length Codes

Problem: Suppose we want to store messages made up of 4 characters a, b, c, d with frequencies 60, 5, 30, 5 (percents) respectively. What are the fixed-length codes and prefix-free codes that use the least space?

Fixed-Length versus Variable-Length Prefix Codes

Solution:

characters	a	b	c	d
frequency	60	5	30	5
fixed-length code	00	01	10	11
prefix code	0	110	10	111

To store 100 of these characters,
(1) the fixed-length code requires $100 \times 2 = 200$ bits,
(2) the prefix code uses only

$$60 \times 1 + 5 \times 3 + 30 \times 2 + 5 \times 3 = 150$$

a 25% saving.

Remark: We will see later that this is the *optimum* (lowest cost) prefix code.

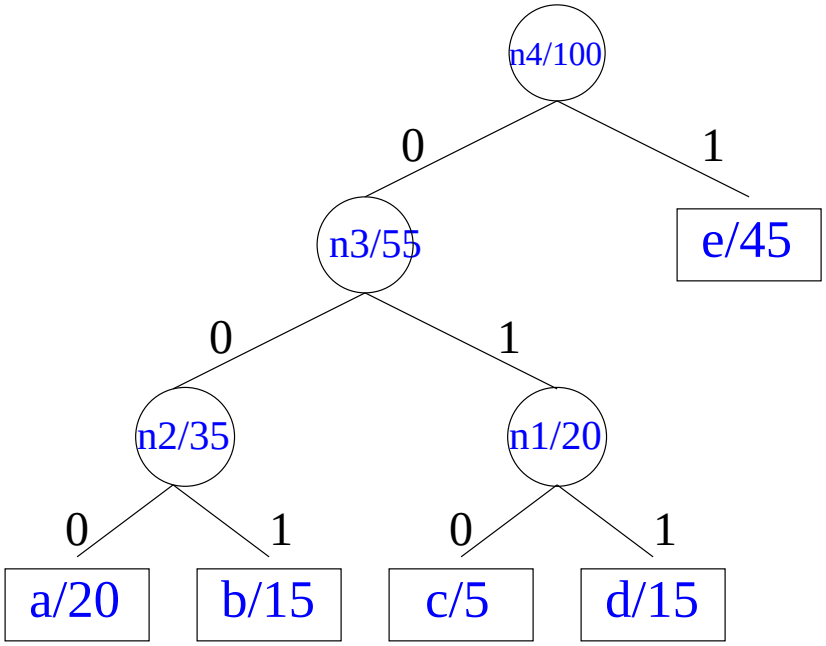
Optimum Source Coding Problem

The problem: Given an alphabet $A = \{a_1, \ldots, a_n\}$ with frequency distribution $f(a_i)$ find a binary prefix code C for A that minimizes the number of bits

$$B(C) = \sum_{a=1}^{n} f(a_i) L(c(a_i))$$

needed to encode a message of $\sum_{a=1}^{n} f(a)$ characters, where $c(a_i)$ is the codeword for encoding a_i, and $L(c(a_i))$ is the length of the codeword $c(a_i)$.

Remark: Huffman developed a nice greedy algorithm for solving this problem and producing a minimum-cost (optimum) prefix code. The code that it produces is called a *Huffman code* .



$$\{a = 000,\ b = 001,\ c = 010,\ d = 011,\ e = 1\}$$

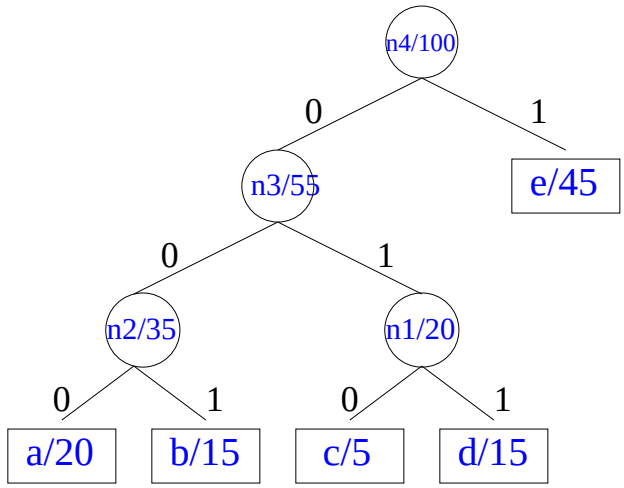
Correspondence between Binary Trees and prefix codes.

1-1 correspondence between leaves and characters.

Label of leaf is frequency of character.

Left edge is labelled 0; right edge is labelled 1

Path from root to leaf is codeword associated with character.



$$\{a = 000,\ b = 001,\ c = 010,\ d = 011,\ e = 1\}$$

Note that $d(a_i)$, the depth of leaf a_i in tree T is equal to the length, $L(c(a_i))$ of the codeword in code C associated with that leaf. So,

$$\sum_{a=1}^{n} f(a_i)L(c(a_i)) = \sum_{a=1}^{n} f(a_i)d(a_i).$$

The sum $\sum_{a=1}^{n} f(a_i)d(a_i)$ is the *weighted external pathlength* of tree T .

The Huffman encoding problem is equivalent to the minimum-weight external pathlength problem: given weights $f(a_1), \ldots, f(a_n)$, find tree T with n leaves labelled $a_1, \ldots, a_n$ that has minimum weighted external path length.

Huffman Coding

Step 1: Pick two letters x, y from alphabet A with the smallest frequencies and create a subtree that has these two characters as leaves. (greedy idea)
Label the root of this subtree as z.

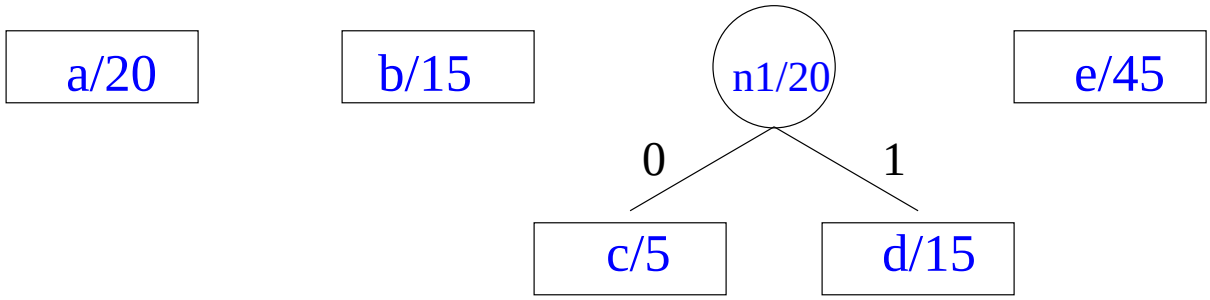
Step 2: Set frequency $f(z) = f(x) + f(y)$.
Remove x, y and add z creating new alphabet
$A' = A \cup \{z\} - \{x, y\}$.
Note that $|A'| = |A| - 1$.

Repeat this procedure, called *merge*, with new alphabet A' until an alphabet with only one symbol is left.

The resulting tree is the Huffman code.

Example of Huffman Coding

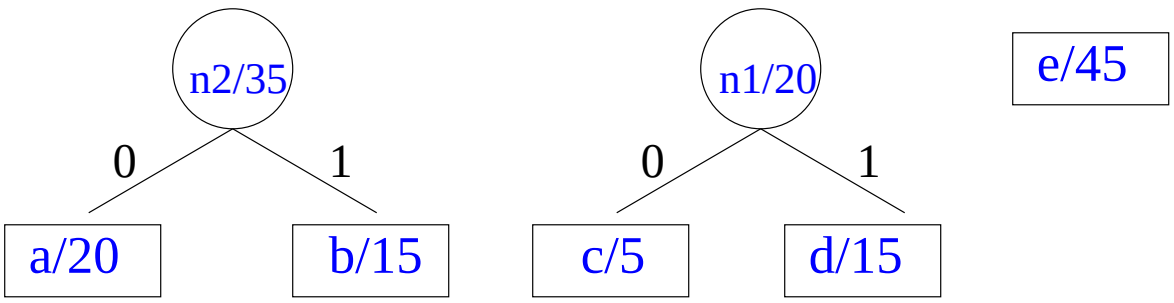
Let $A = \{a/20, b/15, c/5, d/15, e/45\}$ be the alphabet and its frequency distribution.
In the first step Huffman coding merges c and d.



Alphabet is now $A_1 = \{a/20, b/15, n1/20, e/45\}$.

Example of Huffman Coding – Continued

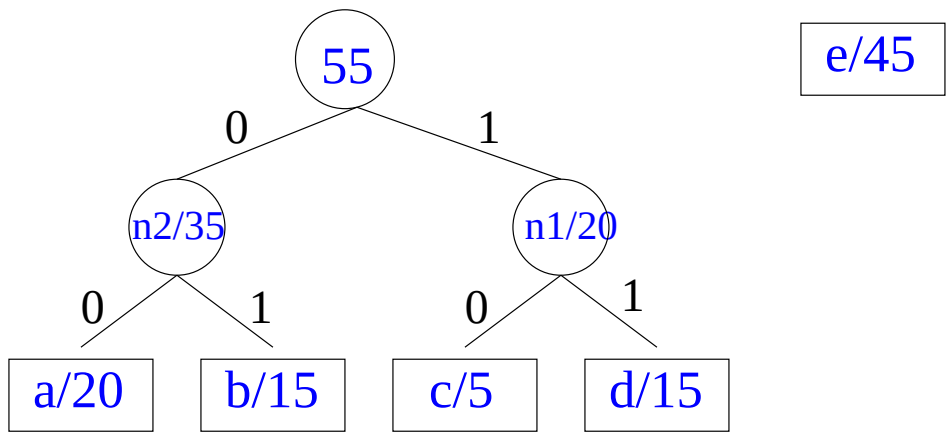
Alphabet is now $A_1 = \{a/20, b/15, n1/20, e/45\}$.
Algorithm merges a and b
(could also have merged $n1$ and b).



New alphabet is $A_2 = \{n2/35, n1/20, e/45\}$.

Example of Huffman Coding – Continued

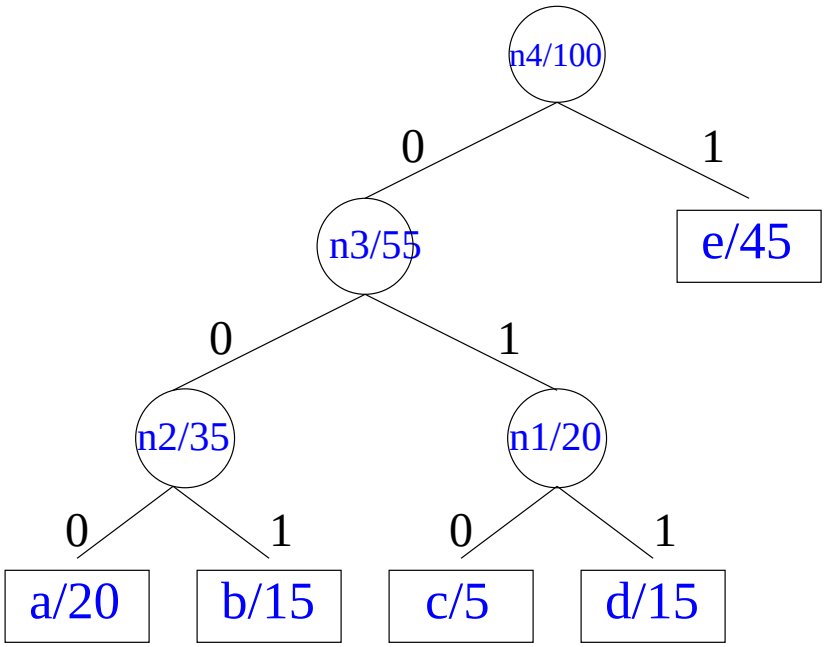
Alphabet is $A_2 = \{n2/35, n1/20, e/45\}$.
Algorithm merges $n1$ and $n2$.



New alphabet is $A_3 = \{n3/55, e/45\}$.

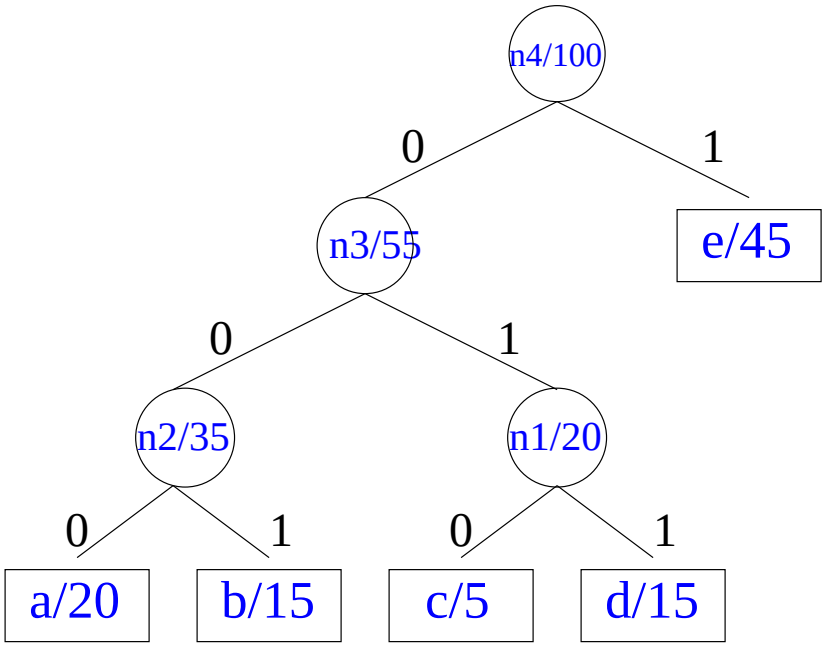
Example of Huffman Coding – Continued

Current alphabet is $A_3 = \{n3/55, e/45\}$.
Algorithm merges e and $n3$ and finishes.



Example of Huffman Coding – Continued

Huffman code is obtained from the Huffman tree.



Huffman code is

$a = 000$, $b = 001$, $c = 010$, $d = 011$, $e = 1$.

This is the optimum (minimum-cost) prefix code for this distribution.

Huffman Coding Algorithm

Given an alphabet A with frequency distribution $\{f(a) : a \in A\}$. The binary Huffman tree is constructed using a priority queue, Q, of nodes, with labels (frequencies) as keys.

```
Huffman(A)
{   n = |A|;
    Q = A;                          the future leaves
    for i = 1 to n − 1              Why n − 1?
    {   z = new node;
        left[z] =Extract-Min(Q);
        right[z] =Extract-Min(Q);
        f[z] = f[left[z]] + f[right[z]];
        Insert(Q, z);
    }
    return Extract-Min(Q)  root of the tree
}
```

Running time is $O(n \log n)$, as each priority queue operation takes time $O(\log n)$.

Huffman Codes are Optimal

Lemma: Consider the two letters, x and y with the smallest frequencies. Then is an optimal code tree in which these two letters are sibling leaves in the tree in the lowest level.

Proof: Let T be an optimum prefix code tree, and let b and c be two siblings at the maximum depth of the tree (must exist because T is full). Assume without loss of generality that $f(b) \leq f(c)$ and $f(x) \leq f(y)$ (if this is not true, then rename these characters). Since x and y have the two smallest frequencies it follows that $f(x) \leq f(b)$ (they may be equal) and $f(y) \leq f(c)$ (may be equal). Because b and c are at the deepest level of the tree we know that $d(b) \geq d(x)$ and $d(c) \geq d(y)$.

Now switch the positions of x and b in the tree resulting in a different tree T' and see how the cost changes. Since T is optimum,

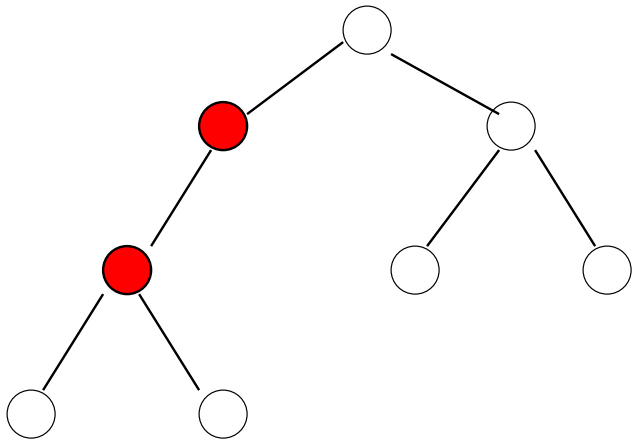
$$\begin{aligned} B(T) &\leq B(T') \\ &= B(T) - f(x)d(x) - f(b)d(b) + f(x)d(b) + f(b)d(x) \\ &= B(T) - (f(b) - f(x))(d(b) - d(x)) \\ &\leq B(T). \end{aligned}$$

Therefore, $B(T') = B(T)$, that is, T' is an optimum tree. By switching y with c we get a new tree T'' which by a similar argument is optimum. The final tree T'' satisfies the statement of the claim.

An Observation: Full Trees

Lemma: The tree for any optimal prefix code must be "full", meaning that every internal node has exactly two children.

Proof: If some internal node had only one child then we could simply get rid of this node and replace it with its unique child. This would decrease the total cost of the encoding.



Huffman Codes are Optimal

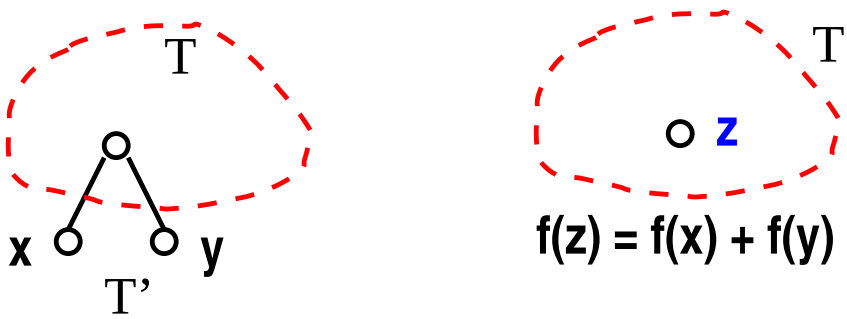
Theorem: Huffman's algorithm produces an optimum prefix code tree.

Proof: By induction on n. When $n = 2$, obvious.

Assume inductively that with strictly fewer than n letters, Huffman's algorithm is guaranteed to produce an optimum tree. We want to show this is also true with exactly n letters.

Huffman Codes are Optimal

Proof – continued: Consider an alphabet A' of n letters. Let T' be an optimum tree for A' with the two letters of lowest frequency x and y as sibling leaves (exists by Lemma). Let T be the coding tree for $A = A' \cup \{z\} - \{x, y\}$ ($n - 1$ leaves) obtained by removing x and y and replacing their parent by z.



$d(x) = d(y) = d(z) - 1$ and $f(z) = f(x) + f(y)$, so

$$\begin{aligned} B(T) &= B(T') - f(x)d(x) - f(y)d(y) + f(z)d(z) \\ &= B(T') - (f(x) + f(y)). \end{aligned}$$

Huffman Codes are Optimal

Proof – continued: By the induction hypotheses, the Huffman algorithm gives a Huffman code tree H_A that is optimal for A. Let $H_{A'}$ be the tree obtained by adding x and y as children of z in H_A. Note that, as in the calculations for T and T', we have:

$$B(H_A) = B(H_{A'}) - (f(x) + f(y)).$$

Also, note that, by the optimality of H_A, we have that $B(H_A) \leq B(T)$, so

$$\begin{aligned} B(H_{A'}) &= B(H_A) + f(x) + f(y) \\ &\leq B(T) + f(x) + f(y) \\ &= B(T'). \end{aligned}$$

Since we started with the assumption that T' was an optimal tree for A' this implies that $H_{A'}$, which is exactly the tree constructed by the Huffman algorithm, is also an optimal tree for A' and we are done.

ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/261347800

An Overview of Perceptual Processing for Digital Pictures

Conference Paper · July 2012

DOI: 10.1109/ICMEW.2012.27

CITATIONS
27

READS
230

3 authors:

Hong Ren Wu
RMIT University
218 PUBLICATIONS **5,797** CITATIONS

SEE PROFILE

Weisi Lin
Nanyang Technological University
478 PUBLICATIONS **18,349** CITATIONS

SEE PROFILE

Lina Karam
Lebanese American University
284 PUBLICATIONS **7,048** CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Project 3D Point Cloud and Mesh Processing View project

Project Video Coding and Compression View project

All content following this page was uploaded by Hong Ren Wu on 19 December 2014.

The user has requested enhancement of the downloaded file.

AN OVERVIEW OF PERCEPTUAL PROCESSING FOR DIGITAL PICTURES

Hong Ren Wu[1], Weisi Lin[2] and Lina J. Karam[3]

Emails: henry.wu@rmit.edu.au, wslin@ntu.edu.sg, karam@asu.edu

[1]School of Electrical and Computer Engineering, Royal Melbourne Institute of Technology, Australia.
[2]School of Computer Engineering, Nanyang Technological University, Singapore.
[3]School of Electrical, Computer & Energy Engineering, Arizona State University, Tempe, AZ 85287, USA.

ABSTRACT

This paper presents an overview of state-of-the-art technologies for perceptual processing of digital pictures, as well as a discussion of the issues related to their implementation, optimization and testing. The paper begins with a brief description of the main computational modules that are used as part of a perceptual-based visual signal processing framework. Then, a number of perceptual-based visual processing techniques and applications to which perceptual models are presented, including image/video compression, visual signal quality evaluation, and computer graphics. The most significant research efforts are highlighted for each topic, and a number of issues and views are put forward regarding the related research and opportunities.

Index Terms— perceptual visual signal processing, image and video compression, visual signal quality evaluation, computer graphics

1. INTRODUCTION

When William M. Goodall of Bell Laboratories reported at the IRE[1] National Convention in 1949 his implementation of the world-first digital video system (using PCM or pulse code modulation) [1], few knew about it, let alone realized that the end of the analogue television (TV) which was beheld with fascination then by the general population around the globe had just been announced, and that replacing the analogue television would be a flurry of digital visual communication and entertainment applications that were imaginable or considered not possible at the time. Wait we must, however, for four more decades before the first international standards (i.e., ITU-T H.261 [2] and JPEG [3]) start to emerge for digital picture compression, which are followed by a proliferation of digital picture coding products and applications as we are witnessing today, including digital images for photography, media and medical applications, digital TV broadcasting, IPTV, IP CCTV, video streaming and on-demand services, DVD and high definition (HD) and 3-D DVD/Blu-ray products, broadband wireless and multimedia communications.

The area of human visual system (HVS) based perceptual picture (including images and videos) processing has been an important and very challenging topic in both theory and practice since the very beginning [4]. Its importance and relevance for providing an improved user's quality of experience (QoE) have become increasingly obvious to the research and professional communities and industries, which prompted efforts to account for perceptual-based visual characteristics in the design and development of future generations of picture coding standards, products, systems and applications such as super- or ultra-high definition (SHD or UHD) imaging and video systems, digital cinema distribution systems, 3-D video/TV, immersive interactive visual systems, and medical imaging and archive systems for telemedicine/telehealth applications [5-7]. It is high time for the visual communication and entertainment services to be transformed from a device-centric "best effort" to user-centric "quality assured" practice for the associated industries to be sustainable in the long run.

In this paper, an overview is provided of state-of-the-art technologies for perceptual visual signal processing, as well as discussions on the issues related to their implementation, optimization and testing. The main frequently used computational modules of a perceptual-based visual processing system are first described in Section 2. Selected perceptual-based visual processing techniques and applications that make use of perceptual models as a means to optimize their performance are presented in Sections 3 to 5, including image/video compression (Section 3), visual signal quality evaluation (Section 4), and computer graphics (Section 5). The most significant research for each topic is highlighted while providing extensive references to related research activities. Finally, a conclusion is presented in Section 6.

2. PERCEPTUAL VISUAL MODELING

A number of perceptual visual models have been developed and have been adopted in the literature. We will discuss two types of perceptual visual modeling in this section.

2.1. Human Visual Attention (VA)

The human visual attention (VA) is the result of several millions of years of evolution by which humans can rapidly

[1] IRE stands for the Institute of Radio Engineers, a predecessor of the IEEE (Institute of Electrical and Electronic Engineers).

978-0-7695-4729-9/12 $26.00 © 2012 IEEE
DOI 10.1109/ICMEW.2012.27

direct their gaze toward objects of interest in the visual field. Two major attention mechanisms include top-down (knowledge/task-driven) [8] and bottom-up (stimulus-driven) [9]. In the former mechanism, attention is under the control of the subject and is related to cognition processing in the human brain. In the latter mechanism, attention is driven by external stimuli to determine which location is sufficiently different from its surroundings to be worthy of one's attention. Generally, the stimuli involved in top-down control are pattern (e.g., human faces), shape and other cognitive features, while the features involved in bottom-up control include luminance, color, orientation and motion contrast. Moreover, audition, touch, and other sensory features also affect VA [10].

The first explicit bottom-up computational architecture was proposed by Koch and Ullman [11], with the result being a 2-D topographic map that represents the stimulus conspicuity, or saliency, at every location in the visual scene. This general architecture has been further developed and implemented [12]. In this model, the early stages of visual processing decompose the incoming visual input into feature maps of colors, intensities, and orientations. The "center-surround" operation is then implemented on multi-scale feature images, which are obtained by using dyadic Gaussian pyramids. All obtained feature maps are then combined into a saliency map to detect attended regions by using a winner-take-all neural network.

There are many other bottom-up VA models, e.g., Graph-based Visual Saliency model [13] with graph theory for saliency maps from low-level features; a VA model based on Shannon's self-information measure [14]; machine learning based saliency map for images [15] with features of multi-scale contrast, center-surround histogram and color spatial distribution; the model based on Fourier Transform [16]; the one considering the human visual sensitivity [17]; GAFFE (gaze-attentive fixation finding engine) [18] and FTS (frequency-tuned salient) region detection [19]. For video modeling, motion information can be used [20].

2.2. Just Noticeable Difference

It is well known that the HVS cannot sense all changes in an image/video due to its underlying physiological and psychological mechanisms [4]. The JND (Just Noticeable Difference) is mainly based upon the temporal/spatial contrast sensitivity function (CSF, which describes the sensitivity of the HVS to each frequency component, as determined by psychophysical experiments), background luminance adaptation (LA, referring to how the contrast sensitivity of the HVS changes in function of the background luminance) and contrast masking (CM, referring to the masking effect of the HVS in the presence of two or more simultaneous frequency components). The JND can be determined either in a sub-band domain [21,22] or in the pixel domain [23,24].

The widely-used JND models were formulated in [21,22] based upon CSF, CM and LA in the DCT (Discrete Cosine Transform) domain. The JND model in [25] formulates CSF and contrast masking in 6 band-pass sub-bands, based on a Laplacian pyramid decomposition of images. Although a model operating in pixel domain can be obtained by summing the effects of the visual thresholds in sub-bands [26], such a model can be derived directly with pixels without subband decomposition [23,24]. The key issue of pixel-domain JND estimation is differentiation of edge and textured regions [27,28].

3. PERCEPTION BASED VISUAL SIGNAL CODING

Theories and principles underpinning digital visual signal coding research are widely known to be the Nyquist-Shannon sampling theorem [29], the Shannon source coding theorem or entropy [30], and the rate-distortion theory [31]. Challenges of varying degrees have been mounted ever since the introduction of the aforementioned theories in field of visual signal processing, and become more intense in the past decade or two with noticeable research activities in compressive sampling [32], the perceptual entropy [33] and perceptually lossless coding of digital pictures [34,35], and rate-perceptual-distortion optimization [33,36].

HVS based picture coding can be broadly grouped into two categories, i.e., perceptual lossless coding and perceptual lossy coding. Perceptual lossless coding removes visually redundant information, offers higher compression ratio than what information lossless coding does with no perceivable picture degradation, and defines the neutral point on the perceptual picture quality scale, given the reference. Perceptual lossy coding, on one hand, aims at delivering constant perceptual picture quality at different JND steps as appropriate to applications in quality assured services and, on the other hand, uses rate-perceptual-distortion optimization for constant bit-rate coding services, providing markedly better perceptual picture quality than traditional information lossy coding techniques where raw mathematical distortion measures, such as the mean square error (MSE), are used in rate-distortion optimization.

3.1. Perceptual Lossy Coding

Many approaches to perception based digital picture coding have been proposed, usually based on the existing coding standards, and modifications are made to explore the perceptual aspect of digital picture coding. Early perceptual picture coders considered simple characteristics of the HVS or HVS weighting function in adaptive quantization design [37,38]. Research efforts over the years have refined this perceptual coding approach which anchors the adaptive quantization to a JND detection model incorporating CSF, LA and CM characteristics of the HVS [39,40,23,41,42,28]. JND detection models have been successfully used in motion estimation and handling for video compression [27,43]. Embedding HVS based perceptual distortion measures in rate-perceptual-distortion optimization process of picture coding forms an alternative approach to perceptual coder design where perceptual distortion is minimized for constant bitrate coding or used to maintain a uniform perceptual distortion level for the entire image or

video [44,45,46]. Stand-alone perceptual distortion metrics providing a visual discrimination map to advise perceptual coding strategies have been successfully deployed in adaptive quantization and bit allocation as well as rate-distortion optimization for quality controlled video coding [47-49]. Utilizing cognitive knowledge and experience such as VA or region of interests have also been reported including foveated image and video coding [50-52].

In [43], the visual signal is smoothed within the constraint of JND for better coding performance. In [42], the CSF, LA, and CM characteristics of the HVS are exploited in developing a locally adaptive perceptual-based image codec that is fully compatible with Part 1 of the JPEG2000 standard (so, any JPEG2K decoder can decode the resulting bitstream), achieving an improved perceived visual quality for both lossy and losseless coding. In [53] and [54], the QPs are adjusted according to the visual impact of the signal for the DCT based coding systems such as JPEG and H.264. In the scheme developed in [46], the perceived error is approximated by a multi-channel color CGC (Contrast Gain Control) vision model based perceptual distortion metric for R-D optimization in order to maximize the visual quality of JPEG 2000 coded images. Compared with the JND detection model based approach, the multi-channel CGC vision model [55] has been adopted in [45] and [46] where model parameters are refitted using application specific subjective test data [56].

3.2. Perceptual Lossless Coding

Visually lossless compression is a special type of perception based coding and it is achieved when a compressed visual signal cannot be differentiated from its original, i.e., the visual difference between the compressed picture and its reference is just below or equal to the just not-noticeable-difference (JNND) threshold. There have been few reports on perceptual lossless picture coding and even less have gone through rigorous verification process via, e.g., double blind subjective tests using a sufficiently large sample pool [35].

In [57], different bit streams are generated by using a standard encoder with different given bit rates, and the one with a resultant visual quality (obtained from a quality measurement, e.g., multi-scale SSIM [58]) close to a predefined threshold (e.g., 0.995 for the multi-scale SSIM) is selected as the bit stream for visually lossless. However, the criterion with which the quality score is close to the predefined threshold is not sufficient for visually lossless, because the quality score can also be high for the case where the image with visible distortion on only a small portion of the image. Therefore, most of the existing visually lossless coding methods [36,59-61] are based on the JND and the CGC with visual pruning, exploiting masking properties of the HVS.

In [41], a perceptual-based image coder that can achieve perceptually lossless compression at high compression ratios (15 to 20) is proposed by adapting the quantizer reconstruction levels to the local amount of masking present at the level of each subband transform coefficient and without the need to send any additional side information to the decoder. Other methods modified one of the standard (e.g., JPEG, or JPEG 2000) encoders to account for the perceptual redundancy, where the distortion related parameters in the encoder are adjusted according to the JND model to guarantee that the reconstructed signal is visually lossless. In [35,62], a visual pruning processing is introduced to a multi-channel CGC vision model which is embedded in the JPEG 2000 encoder producing standard complaint bit-stream, forcing perceptual distortion below or at JNND threshold and achieving perceptual lossless coding; in [59] a JND model is incorporated into the JPEG 2000 encoder and the encoded bit stream can be decoded by a JPEG 2000 decoder; in [60] and [61], the visually lossless coding is realized by the DCT based encoder with a JND-adjusted QP and the resultant bit stream is not standard compliant.

However, these encoder manipulation based methods are embedded in the specific encoder (JPEG, or JPEG 2000), and therefore cannot be used directly as part of other standard codecs such as the recently proposed H.264 lossless coding. To address this problem in migrating perceptual lossless techniques designed for still image coding to video coding where inter-frame prediction is routinely required, a theoretical framework was recently introduced which is applicable to rate-perceptual-distortion optimization based perceptual video coding implementations [63].

As an example, taking a visual transform domain decomposition model and given an I-frame image, $\mathbf{x}_I$, and its transform decomposition, $\mathbf{X}_I$, with a suitable perceptual distortion measure (*PDM*), the visually filtered coefficient matrix for intra-frame coding which delivers the perceptual lossless compression with maximum perceptual distortion equal to or just under the JNND, $\mathbf{X}_{I_F}$, is defined as [62]

$$\mathbf{X}_{I_F} = \arg\max_{\forall \mathbf{X}_{I_f}} \left\{ PDM\left(\mathbf{X}_I, \mathbf{X}_{I_f}\right) \le JNND \right\} \qquad (1)$$

where $\mathbf{X}_{I_f}$ is visually filtered coefficient matrix from $\mathbf{X}_I$, (e.g., any interim result from an EBCOT-Embedded Block Coding with Optimal Truncation-coder [44]). The prediction error of the visually filtered transform coefficients of the P-frame at JNND level is defined as

$$\Delta\mathbf{X}_{P_F} = \arg\max_{\forall \Delta\mathbf{X}_{P_f}} \left\{ PDM\left(\mathbf{X}_P, \mathbf{X}_{P_f}\left(\mathbf{X}_{I_F}, \Delta\mathbf{X}_{P_f}\right)\right) \le JNND \right\} \qquad (2)$$

where $\mathbf{X}_P$ is the transform decomposition representation of the original P-frame,

$$\mathbf{X}_{P_f}\left(\mathbf{X}_{I_F}, \Delta\mathbf{X}_{P_f}\right) = \mathbf{X}_{I_F} + \Delta\mathbf{X}_{P_f}, \qquad (3)$$

and $\Delta\mathbf{X}_{P_f}$ is the prediction error of the filtered P-frame transform coefficients. The minimum bitrate m_r is used to encode the P-frame of the sequence, and defined as

$$m_r = \min\left(R\left(\Delta\mathbf{X}_{P_F}\right), R\left(\mathbf{X}_{P_F}\right)\right) \qquad (4)$$

where $\mathbf{x}_{P_F}$ is the perceptually lossless intra-frame coded P-frame and $\Delta\mathbf{x}_{P_F}$ is given in (2). When motion-compensated prediction is used, $\mathbf{x}_{I_F}$ in Eq. (3) will be replaced by motion-compensated prediction (MP), $\mathbf{x}_{MP_F}$ which obtained from a reference I-frame or a reconstructed previous P-frame.

Another challenging issue is that the existing picture coding standards have adopted complete transforms as mathematical framework for compression, which are not shift-invariant leading to aliasing distortion and constitute a major weakness when used to approximate JND/JNND models in formulating perceptual distortion/quality metrics [36].

4. PICTURE QUALITY EVALUATION

The Mean Squared Error (MSE) and the Peak Signal to Noise Ratio (PSNR) are mathematically tractable and easy-to-compute metrics. However, they can be poor predictor of visual quality, especially when the noise is not additive. The major reason for the overall poor performance of the MSE (or the PSNR) is its assignment of equal importance to all the changes in a visual signal regardless of their perceptual significance.

Some picture quality metrics have been discussed in the last section. Objective image quality assessment can be classified into three categories based on the amount of information used for predicting quality [36]: (1) Full-reference (FR) metrics which use complete reference signal (image/video) information, (2) Reduced-reference (RR) metrics which use only partial information from the reference signal and (3) No-reference (NR) metrics which do not use any reference signal information. With regards to developing an objective PQA (Picture Quality Assessment) algorithm, it can be handled by two broad approaches, as discussed in Sections 4.1 and 4.2.

4.1. Vision Modeling Approach

The vision modeling approach is based on incorporating various HVS characteristics, and aims at simulating the processes of the HVS from the eye to the visual cortex. These metrics are intuitive and appealing since they attempt to account for the properties of the HVS relevant to perceptual quality assessment. The first image and video quality metrics were developed in [64] and [65], respectively. Later other popular HVS-based metrics were developed, including the Visual Differences Predictor (VDP) [66], the Sarnoff JND metric [67], the Moving Picture Quality Metric [68], and the perceptual distortion metric [69].

Although the HVS-based metrics are attractive in theory, they may suffer from some drawbacks. The HVS comprises of many complex processes which work in conjunction rather than independently, to produce visual perception. However, the HVS-based metrics generally utilize results from psychophysical experiments which are typically designed to explore a single dimension of the HVS at a time. In addition, these experiments usually use simple patterns such as spots, bars, and sinusoidal gratings which are much simpler than those occurring in real images. It has been widely acknowledged that a majority, if not all, of PQA metrics in this category adopted vision models devised via threshold vision experiments, which have been contested regarding its extendibility in PQA tasks which commonly fall in supra-threshold vision experiments [36]. It has been found that directly using parameters from the original modeling of HVS experiment where synthetic test patterns were used may not work, re-parameterization of vision model using application specific subjective data, such as those obtained by VQEG for video quality assessment, has been proven to improve PQA performance effectively [56]. In essence, these metrics suffer from drawbacks which mainly stem from the use of oversimplified models describing the HVS. Moreover, the HVS characteristics are not fully understood yet. In addition, due to the complex and highly non-linear nature of the HVS, some of these metrics can be complicated and time-consuming to be used in practice.

4.2. Engineering Approach

Owing to the limitations of the vision-based models discussed above, the engineering based approach has gained popularity during recent years, and it is based primarily on the extraction and analysis of certain features or artifacts in signals, e.g., structural image elements such as contours, or specific distortions (e.g., blocking artifacts) that are introduced by a process, compression technology or transmission link. This does not necessarily mean that such metrics disregard the human vision properties, as they often consider psychophysical effects as well, albeit image content and distortion analysis are the conceptual basis for their design rather than fundamental vision modeling.

With this approach, the assessment of image quality can be considered as a two-stage process: (a) feature extraction, and (b) feature pooling. As for the first stage, the selected features have to form an effective representation of visual quality variations, while the second stage determines the relationship among different features and the perceived visual quality.

A well known FR metric is the structural similarity index measure (SSIM) [70], which is mainly based on the idea of equating the perceived image distortion to the measurement of structural distortion. In SSIM, the mean of quality scores of individual image blocks gives the overall image quality score. Other FR schemes include the Visual Information Fidelity (VIF) index [71] and visual signal-to-noise ratio (VSNR) [72]. The metric known as MSVD (Mean Singular Decomposition) [73] evaluates the quality of each image block based on the error in singular values of

the block, as a result of SVD. Another more recent scheme has been proposed in [74] to measure the change in singular vectors of the reference and distorted images in order to compute the structural changes, and the overall quality is determined by a Minkowski summation. An NR metric using the SVD of local image gradients has been proposed in [75] and used for the proper selection of the parameters of image denoising algorithms. An RR metric that extracts structural features from images, has been proposed in [76]. The overall quality score is then computed as a weighted sum of the features where the weights have been determined by means of subjective experiments.

4.3. Issues Related to Feature Pooling

As for feature (error) pooling, researchers have employed several techniques to fuse the visual features into a quality score such as simple summation based fusion, Minkowski combination, and linear (i.e., weighted) combination. As mentioned in Section 4.2, subjective experiments may be used to compute the weights [76] but such a method is unsuitable for real-time applications. Another method has been developed [77] which involves weighting quality scores with weights determined by local image content, assuming the image source to be a local Gaussian model and the visual channel to be an additive Gaussian model.

Recently, two pooling strategies have been proposed [78] for the SSIM metric. Instead of using a simple mean as the overall quality score, these approaches attempt to weigh the quality scores of different blocks based on visual importance. The first strategy is based on the idea that lower quality regions in images attract more attention than the ones with higher quality; the second strategy uses VA to provide weighting [79]-[80] which is based on the idea that certain regions attract more human attention than the others. The strategy of feature pooling using VA while intuitive may suffer from drawbacks due to the fact that it is not always easy to ascertain regions that attract VA.

5. VISUAL PERCEPTION BASED PERCEPTUAL GRAPHIC RENDERING

There is a growing demand for computer graphics tools that can provide high-end photo-realistic rendering in various applications including gaming, virtual reality, movies, and medical imaging, to name a few. Synthesizing photorealistic images with global illumination is very time-consuming, especially when object models are sophisticated and accurate light simulation is required. Therefore, how to maximize the rendering quality while minimizing the rendering time with limited computational resources, becomes an important research issue.

By exploiting the characteristics of the HVS, significant rendering time can be saved. During the past several decades, various different characteristics of the HVS have been studied and modeled for rendering applications. By using appropriate perceptual models, numerous perceptual image rendering algorithms have been proposed to save the rendering cost.

5.1. VA based perceptual rendering

Since the human's processing resources are limited, the HVS tends to selectively concentrate on one aspect of the surrounding environment while ignoring the other aspects. The VA models have been exploited in some of the perceptual rendering algorithms. Instead of distributing the rendering computation uniformly over a whole image plane, in VA based perceptual rendering, computation is allocated according to the VA level of image regions. Less amount of computations are performed for regions with lower VA levels, exploiting the fact that human observers are less likely to notice artifacts in non-VA regions. The work in [81] attempted to use a VA model to guide the rendering. The authors proposed a perceptual model which combines a spatio-temporal contrast sensitivity model and a bottom-up VA model [12], and used the proposed model to predict how much error human observers can tolerate at each image pixel position. By allocating less rendering computation to regions with a high error tolerance level, ninety percent of the indirect lighting computation can be saved in radiance caching algorithm.

The work in [82] also attempted to apply a bottom-up VA model in rendering. They improved the accuracy of the VA model of [12], by making use of the 3-D object information such as objects' distance from image center and size of each object. The scheme in [83] applied the bottom-up VA model in ray tracing. In their scheme, the sampling rate for each pixel is determined by the VA level of that pixel, and higher numbers of sample rays are traced for pixels with a higher VA level.

The top-down attention mechanism has been also used in [84], where a top-down attention model was used in ray tracing. Viewers are given a visual task, e.g., to find out how many teapots exist in a rendered image, and then a task map is generated to predict where people will focus their attention to the given specific task, following the principle of Inattentional Blindness. Using this task map, pixels with higher attention level would receive more samples in the rendering process. In [85], the top-down information was used to guide the allocation of computations in the rendering of dynamic virtual environments, e.g., to predict the object which a user would consider as salient and allocate more rendering computational resources to this object. The work in [86] combined both the bottom-up and top-down attention processes into a VA model and applied it in a ray tracing algorithm. They used the VA model of [12] to generate the VA map and to task information to generate a task map; then they combined the two maps into an importance map and allocated the sample rays accordingly. They also used the importance map of a scene to guide the perceptual rendering [87]. Another top-down stimulus which has also been used for perceptual rendering is the memory schema [88]. It is a knowledge structure related to the human memory, and is used to indicate the context of a scene and

the correlation between objects and the scene. Inconsistent objects which were not expected to be found in a scene were determined as salient, and then models of higher level of detail (LoD) were used for rendering.

5.2. Other aspects of high-level visual perception based perceptual rendering

Besides modeling the VA mechanism, researchers have also worked on perceptual rendering algorithms that take advantage of other aspects of the high-level visual processing. The scheme in [89] took into account the perception of illumination components in global illumination rendering. The global illumination computation is decomposed into several direct and indirect illumination components and illumination components which have lower perceptual importance in perceiving the final image quality, are rendered with a lower number of computational resources. In this case, significant savings in computation can be obtained.

The work in [90] proposed a concept of 'visual equivalence', which tries to determine whether two rendered images convey the same perceived information including shape, material and illumination instead of determining whether two images are visually the same. The author of [90] then applied the perceptual 'visual equivalence' metric in pre-computed radiance transfer rendering algorithms. Experiments showed that an illumination computation of less accuracy is able to generate images with the same level of perceived information as the reference images. Later in [91], the visual equivalence metric is applied to control of the approximation accuracy of one of the global illumination approximation rendering methods, the virtual point light method. The metric was used to decide where the virtual point light method should be compensated with a path tracing algorithm to achieve a visually equivalent result as the reference image.

6. CONCLUSIONS

Reviewing the field of perceptual processing of digital pictures, significant and concrete progresses have been witnessed over the past decade or two in HVS modeling and adaptation, HVS based picture quality assessment and metrics design, perceptual distortion controlled and perceptual lossless coding of digital pictures including both still images and video, and HVS based perceptual graphic rendering. Embedding HVS based perceptual distortion measures in standard or proprietary picture coders using complete transforms have demonstrated significant improvement in perceived picture quality, making the first step for the constant quality coding. Three issues at least are identifiable, i.e., complete transforms are not shift-invariant and, therefore, are prone to aliasing, JND detection or threshold vision models may not scale well in perceptual lossy coding which operates in supra-threshold environment, and existing HVS models are based on image data in either pixel domain or decomposition domain which requires efficient alternative adaptation for inter-frame (and inter-view) coding of video (and 3-D or multi-view video). Perceptually driven computer graphics are a developing area.

REFERENCES

[1] W.M. Goodall. Television by pulse code modulation. Bell Systems Technical Journal, 28:33-49, January 1951.

[2] ITU Telecommunication Standardization Sector, ITU-T Rec. H.261 Line Transmission on Non-Telephonee Signals-Video Codec for Audiovisual Services at px64 kbit/s, Geneva, 1990.

[3] ITU, Rec. T.81 (09/92) Terminal Equipment and Protocols for Telematic Services – Information Technology – Digital Compression and Coding of Continuous-Tone Still Images-Requirements and Guidelines, 1993.

[4] A.J. Seyler and Z.L. Budrikis, "Measurements of temporal adaptation to spatial detail vision," Nature, Vol. 184, pp. 1215-1217, Oct. 1959.

[5] G. Ghinea, G.-M. Muntean, P. Frossard, M. Etoh, F. Speranza, and H.R. Wu, (Guest Editors), Special Issue on "Quality Issues on Mobile Multimedia Broadcasting", The IEEE Transactions on Broadcasting, 54(3), Pt II, 2008.

[6] L.J. Karam, T. Ebrahimi, S. Hemami, T. Pappas, R. Safranek, Z. Wang, and A.B. Watson, (Eds), IEEE Journal on Special Topics in Signal Processing, Special Issue on Visual Media Quality Assessment, vol. 3, no. 2, pp. 189-192, April 2009.

[7] T. Ebrahimi, L.J. Karam, F. Pereira, K. El-Maleh, and I.S. Burnett, IEEE Signal Processing Magazine, Speicial Issue on Multimedia Quality Assessment – A world of applications, Vol.28, No.6, November 2011.

[8] J. Hopfinger, M. Buonocore, G. Mangun, "The neural mechanisms of top-down attentional control," *Nature Neuroscience*, vol. 3(3), pp.284–291, 2000.

[9] L. Itti, "Visual salience," *Scholarpedia*, 2(9):3327, 2007. [Online] http://www.scholarpedia.org/ article/Visual_salience.

[10] D. Lloyd, N. Merat, F. McGlone, C. Spence, "Crossmodal links between audition and touch in covert endogenous spatial attention," *Perception Psychophys.*, 65(6):901–924, 2003.

[11] C. Koch, S. Ullman, "Shifts in selective visual attention: towards the underlying neural circuitry," *Human Neurobiology 4*, pp.219–227, 1985.

[12] L. Itti, C. Koch, E. Niebur, "A model of saliency based visual attention for rapid scene analysis," *IEEE Trans. Pattern Analy. Mach. Intell.*, vol. 20(11), pp.1254–1259, 1998.

[13] J. Harel, C. Koch and P. Perona, "Graph-based visual saliency," *Advances in Neural Information Processing Systems*, pp. 545–552, 2006.

[14] N. Bruce and J. Tsotsos, "Saliency based on information maximization," *Advances in Neural Information Processing Systems*, pp. 155–162, 2006.

[15] T. Liu, J. Sun, N. Zheng, X. Tang and H. Shum, "Learning to detect a salient object," in *Proc. Int. Conf. Computer Vision and Pattern Recognition*, pp. 1–8, 2007.

[16] X. Hou and L. Zhang, "Saliency Detection: A spectral residual approach," in *Proc. Int. Conf. Computer Vision and Pattern Recognition*, pp. 1–8, 2007

[17] Y. Fang, W. Lin, B-S Lee, C. T. Lau, Z. Chen, C-W Lin, "Bottom-up Saliency Detection Model Based on Human Visual Sensitivity and Amplitude Spectrum", IEEE Transactions on Multimedia, Vol. 14(1), pp. 187 - 198, 2012.

[18] U. Rajashekar, I. van der Linde, A. Bovik, and L. Cormack, "GAFFE: A gaze-attentive fixation finding engine," *IEEE Transactions on Image Processing*, vol. 17(4):564–73, 2008.

[19] R. Achanta, S. Hemami, F. Estrada, S. Susstrunk, "Frequency-tuned salient region detection," *IEEE Conf.*

Computer Vision and Pattern Recog. (2009), pp. 1597–1604.
[20] Z. Lu, W. Lin, X. Yang, E. Ong, S. Yao, "Modeling Visual Attention's Modulatory Aftereffects on Visual Sensitivity and Quality Evaluation", *IEEE Trans. Image Processing,* Vol.14(11), pp.1928 – 1942, 2005.
[21] A. Ahumada and H. Peterson, "Luminance-model-based DCT quantization for color image compression," *SPIE Int'l Society for Optical Engineering*, 1666:365–374, 1992.
[22] A. B.Watson, "DCT quantization matrices visually optimized for individual images, *Proceedings of the SPIE International Society for Optical Engineering*, vol. 1913:202–216, 1993.
[23] C. Chou, C. Chen, "A perceptually optimized 3-D subband image codec for video communication over wireless channels," *IEEE Trans. Circuits Syst. Video Technol.*, 6(2): 143–156, 1996.
[24] Y. Chiu, T. Berger, "A software-only videocodec using pixelwise conditional differential replenishment and perceptual enhancement," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9(3), pp. 438–450, 1999.
[25] M. Ramasubramanian, S. Pattanaik, D. Greenberg, "A perceptual based physical error metric for realistic image synthesis," *Proceedings of Computer Graphics Conference*, vol. 4(33), pp. 73–82, 1999.
[26] X. Zhang, W. Lin and P. Xue, "Just-Noticeable Difference Estimation With Pixels in Images," *J. of Visual Commun. and Image Representation*, 19(1):30–41, 2008.
[27] X. Yang, W. Lin, Z. Lu, E. Ong, and S. Yao, "Just noticeable distortion model and its applications in video coding," *Signal Process.:Image Commun.*, vol. 20, pp. 662–680, 2005.
[28] A. Liu, W. Lin, M. Paul, C. Deng, F. Zhang, "Just Noticeable Difference for Images with Decomposition Model for Separating Edge and Textured Regions," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20(11):1648-1652, 2010.
[29] C.E. Shannon, "Communication in the Presence of Noise," Proc. The IRE, vol. 37, no.1, pp. 10–21, Jan. 1949.
[30] C.E. Shannon, "A Mathematical Theory of Communication," Bell Syst. Tech. J., vol. 27, pp. 379–424, July 1948; pp. 623–657, Oct. 1948.
[31] T. Berger and J.D. Gibson, Lossy Source Coding, IEEE Trans. Information Theory, Vol.44(6), pp.2693-2723, 1998.
[32] R.G. Baraniuk, E. Candes, R. Nowak and M.Vetterli, "Compressive Sampling," IEEE Signal Processing Magazine, vol.25, no.2, pp.31-40, 2008.
[33] N. Jayant, J. Johnston and R. Safranek, "Signal Compression Based on Models of Human Perception", Proceedings of The IEEE, vol. 81, no.10, pp.1385-1422, October, 1993.
[34] A. B. Watson, "Receptive Fields and Visual Represtations," in SPIE Proceedings. SPIE, vol. 1077, pp. 190-197, 1989.
[35] D. Wu, D. Tan, M. Baird, J. DeCampo, C. White, H. Wu, "Perceptually Lossless Medical Image Coding," IEEE Trans. Medical Imaging, vol. 25(3), pp. 335–344, 2006.
[36] H. Wu and K. Rao, Eds, Digital Video Image Quality and Perceptual Coding, CRC Press, 2006.
[37] N.B. Nill. A Visual Model Weighted Cosine Transform for Image Compression and Quality Assessment. IEEE Transactions on Communications, vol.33(6):551-557, 1985.
[38] K.K. Ngan, K.S. Leong, H.Singh. Adaptive Cosine Transform Coding of Images in Perceptual Domain. IEEE Trans. Acous., Speech, and Sig. Proc., 37: 1743-1750, 1989.
[39] A.B. Watson. DCTune: A Technique for Visual Optimization of DCT Quantization Matrices for Individual Images. In Society for Information Display Digest of Technical Papers XXIV, 946-949. 1993.
[40] C.-H. Chou, Y.-C. Li, "A Perceptually Tuned Subband Image Coder Based on the Measure of Just-Noticeable-Distortion Profile," IEEE Trans Circuits and Systems for Video Techn., 5(6):467-476, 1995.
[41] I. Höntsch and L.J. Karam, "Locally Adaptive Perceptual Image Coding," IEEE Transactions on Image Processing, vol. 9, pages 1472-1483, Sept. 2000.
[42] Z. Liu, L.J. Karam, and A.B. Watson, "JPEG2000 Encoding with Perceptual Distortion Control," IEEE Trans. on Image Proc., vol. 15, no. 7, pp. 1763-1778, Jul 2006.
[43] X. Yang, W. Lin, Z. Lu, E. Ong and S. Yao, "Motion-compensated Residue Pre-processing in Video Coding Based on Just-noticeable-distortion Profile," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15(6), pp.742–750, June, 2005.
[44] D. S. Taubman, "High performance scalable image compression with EBCOT," IEEE Trans. Image Proc., Vol. 9, pp. 1158-1170, July 2000.
[45] D.M. Tan, H.R. Wu and Z. Yu "Perceptual Coding of Digital Monochrome Images", IEEE Signal Processing Letters, 11(2): 239-242, February 2004.
[46] D. Tan, C. Tan, H. Wu, "Perceptual Color Image Coding With JPEG2000," *IEEE Trans. Image Process.*, vol. 19(2), pp. 374–383, June, 2010.
[47] A. Pica, M. Isnardi, J. Lubin, "HVS Based Perceptual Video Encoders," in H.R. Wu, K.R. Rao, eds., Digital Video Image Quality and Perceptual Coding, CRC Press, 2006.
[48] I. Dalgic and F.A. Tobagi, "A Constant Quality MPEG-1 Video Encoding Scheme and its Traffic Characterization," Int'l Picture Coding Symp. 1: 105-110, 1996.
[49] T.-S. Ou, Y.-H. Huang and H.H. Chen. SSIM-Based Perceptual Rate Control for Video Coding. IEEE Trans. Circuits and Systems for Video Techn., 21(5): 682-691, 2011.
[50] S. Lee, M.S. Pattichis and A.C. Bovik. Foveated Video Compression with Optimal Rate Control. IEEE Transactions on Image Imaging, 10(7): 977 - 992, July 2001.
[51] Z. Wang, L. Lu and A.C. Bovik. Foveation Scalable Video Coding with Automatic Fixation Selection. IEEE Transactions on Image Processing, 12(2): 243-254, 2003.
[52] X.Yang, W.Lin, Z.Lu, X.Lin, S.Rahardja, E.Ong, S.Yao, "Rate Control for Videophone Using Local Perceptual Cues," IEEE Trans Circuits and Systems for Video Techn., 15(4): 496-507, 2005.
[53] M. Naccari, F. Pereira, "Decoder side just noticeable distortion model estimation for efficient H.264/AVC based perceptual video coding," *Int. Conf. Image Processing*, 2010.
[54] Z. Chen and C. Guillemot, "Perceptually-friendly H.264/AVC video coding based on foveated Just-Noticeable-Distortion model," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20(6), pp. 806–819, June 2010.
[55] A. B. Watson and J. A. Solomon, "A model of visual contrast gain control and pattern masking," J. Opt. Soc. Amer. A, pp. 2379–2391, 1997.
[56] Z. Yu, H.R. Wu, S. Winkler and T. Chen, "Objective Assessment of Blocking Artifacts for Digital Video with a Vision Model", Proc. of IEEE, 90(1):154-169, January 2002.
[57] V. Lukin, M. Zriakhov, N. Ponomarenko, S. Krivenko, Z. Miao, "Lossy compression of images without visible distortions and its application," *Int. Conf. Signal Processing*, pp. 698–701, 2010.
[58] Z. Wang, E. Simoncelli, and A. Bovik, "Multi-scale structural similarity for image quality assessment," *IEEE Asilomar Conf. on Signals, Systems, and Computers*, 2003.
[59] O. Han, A. Bilgin, W. Michael, "Multi-resolution visually lossless image coding using JPEG2000," *Proc. Int. Conf. Image Processing*, pp. 2581–2584, 2010.
[60] Z. Wei, K. Ngan, "The perceptually transparent coding for

image," *Int. Symp. Circuits and Systems*, pp. 1621–24, 2009.
[61] E. Tan, W. Gan, M. Wong, "Perceptually lossless coder based on just noticeable distortion profile with subband DCT," in *Proc. Int. Symp. Consumer Electronics*, pp. 253–257, 2005.
[62] D. Wu, D. Tan, H. Wu, "Perceptual coding at the threshold level for the Digital Cinema System Specification," in *Proc. Int. Conf. Multimedia and Expo*, pp. 796–801, 2010.
[63] H.R. Wu, D.M. Tan and D. Wu, "A Theoretical Framework for Perceptually Lossless Coding of Stereo 3-D Video," Proc. IASTED SIP 2011, Dallas, Texas, USA, Dec. 2011.
[64] J. L. Mannos and D. J. Sakrison, "The effects of a visual fidelity criterion on the encoding of images," *IEEE Trans. Information Theory*, vol. 20, no. 4, pp. 525–536, July 1974.
[65] F. Lukas and Z. Budrikis, "Picture quality prediction based on a visual model," *IEEE Trans. Communications*, vol. 30, no. 7, pp. 679–1692, July 1982.
[66] S. Daly, "The visible differences predictor: An algorithm for the assessment of image fidelity," in *Digital Images and Human Vision*, A.B. Watson, Ed. Cambridge, MA: MIT Press, 1993, pp. 179–206.
[67] J. Lubin and D. Fibush, "Sarnoff JND Vision Model," T1A1.5Working Group Document #97-612, ANSI T1 Standards Committee, 1997.
[68] C. J. van den Branden Lambrecht and O. Verscheure, "Perceptual quality measure using a spatio-temporal model of the human visual system," *SPIE Digital Video Compression: Algorithms and Technologies*, 1996, vol. 2668, pp. 450–461.
[69] S. Winkler, "A perceptual distortion metric for digital color video," in *Proc. SPIE Human Vision and Electronic Imaging*, San Jose, CA, January 23–29, 1999, vol. 3644, pp. 175–184.
[70] Z. Wang, A. Bovik, H. Sheikh, E. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13(4): 600–612, 2004.
[71] H. Sheikh and A. Bovik, "Image information and visual quality," *IEEE Trans. Image Process.*, vol. 15, no. 2, pp. 430–444, 2006.
[72] D. Chandler and S. Hemami, "VSNR: A Wavelet-Based Visual Signal-to-Noise Ratio for Natural Images," *IEEE Trans. on Image Processing*, vol. 16, no. 9, 2007.
[73] A. Eskicioglu, A. Gusev, A. Shnayderman, "An SVD-Based Gray-Scale Image Quality Measure for Local and Global Assessment," *IEEE Trans. Image Proces.*, 15:422-429, 2006.
[74] M. Narwaria, and W. Lin, "Objective Image Quality Assessment Based on Support Vector Regression", IEEE Transactions on Neural Networks, 21(3):515-519, 2010.
[75] X. Zhu, P. Milanfar, "Automatic Parameter Selection for De-noising Algorithms Using a No-Reference Measure of Image Content, " *IEEE Trans. on Image Proc.*, 19:3116-32, 2010.
[76] U. Engelke, M. Kusuma. , H.J. Zepernick and M. Caldera, "Reduced-reference metric design for objective perceptual quality assessment in wireless imaging," *Signal Processing: Image Communication*, Vol. 24, pp.525-547, 2009.
[77] Z. Wang and X. Shang, "Spatial Pooling Strategies for Perceptual Image Quality Assessment," *Proc. of IEEE International Conferencing on Image Processing*, 2006.
[78] A. Moorthy and A. Bovik, "Visual Importance Pooling for Image Quality Assessment," *IEEE J. of Selected Topics in Signal Processing*, Vol. 3, no. 2, 2009.
[79] Q. Ma and L. Zhang, "Image Quality Assessment with Visual Attention," *Int. Conf. on Pattern Recognition (ICPR)*, 2008.
[80] J. You, A. Perkis, M. Hannuksela and M. Gabbouj, "Perceptual Quality Assessment Based on Visual Attention Analysis," *ACM Int'l Conf. Multimedia*, 2009.
[81] H. Yee, S. Pattanaik, D. P. Greenberg, "Spatiotemporal sensitivity and visual attention for efficient rendering of dynamic environments," ACM Trans on Graphics, 20(1): 39–65, 2001.
[82] J. Haber, K. Myszkowski, H. Yamauchi, and H. P. Seidel, "Perceptually guided corrective splatting," in Computer Graphics Forum, vol. 20, pp. 142–153, 2001.
[83] P. Longhurst, K. Debattista, and A. Chalmers, "Snapshot: A rapid technique for driving a selective global illumination renderer," WSCG 2005 SHORT papers proceedings, 2005.
[84] K. Cater, A. Chalmers, and G.Ward, "Detail to attention: exploiting visual tasks for selective rendering," 14th Eurographics workshop on Rendering, pp. 270–280, 2003.
[85] S. Lee, G. J. Kim, and S. Choi, "Real-time tracking of visually attended objects in virtual environments and its application to LOD," IEEE Trans Visualization and Computer Graphics, pp. 6–19, 2008.
[86] V. Sundstedt, K. Debattista, P. Longhurst, A. Chalmers, T. Troscianko, "Visual attention for efficient high-fidelity graphics," Spring Conference on Computer Graphics, 2005.
[87] V. Sundstedt, D. Gutierrez, O. Anson, F. Banterle, and A. Chalmers, "Perceptual rendering of participating media," ACM Trans. Appl. Percept., vol. 4, November 2007.
[88] A. Bennett, M. Coxon, and K. Mania, "The effect of stereo and context on memory and awareness states in immersive virtual environments," Proceedings of 7th Symp. on Applied Perception in Graphics and Visualization, pp. 135–140, 2010.
[89] W. Stokes, J. Ferwerda, B. Walter, and D. Greenberg, "Perceptual illumination components: a new approach to efficient, high quality global illumination rendering," ACM Transactions on Graphics, vol. 23, pp. 742–749, ACM, 2004.
[90] G. Ramanarayanan, J. Ferwerda, B. Walter, and K. Bala, "Visual equivalence: towards a new standard for image fidelity," ACM Trans. Graph., vol. 26, July 2007.
[91] J. Kˇriv´anek, J. Ferwerda, and K. Bala, "Effects of global illumination approximations on material appearance," ACM Transactions on Graphics, vol. 29(4), pp. 112:1–112:10, 2010.
[92] J.R. Casas Pla, Image Compression based on Perceptual Coding Techniques, PhD Thesis, Technical Uniersity of Catalonia, Barcelona, 1996.
[93] I. Dalgic and F.A. Tobagi, "A Constant Quality MPEG-1 Video Encoding Scheme and its Traffic Characterization," Int'l Picture Coding Symp. 1: 105-110, March 1996.
[94] T.-S. Ou, Y.-H. Huang and H.H. Chen. SSIM-Based Perceptual Rate Control for Video Coding. IEEE Trans. Circuits and Systems for Video Techn., 21(5): 682-691, 2011.

View publication stats