TFIC: End-to-End Text-Focused Image Compression for Coding for Machines

S. Della Fiore A. Gnutti M. Dalai P. Migliorati R. Leonardi

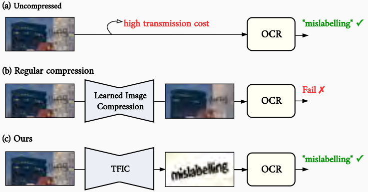
EUSIPCO 2025

University of Brescia, Italy
University of Roma Tor Vergata, Italy

Introduction & Motivation

The Problem

- Traditional image compression aims to reconstruct images for human perception.
- However, compression artifacts (blurring, loss of detail) can severely impact machine vision tasks like OCR.



Comparison of frameworks:
(a) No compression, (b) Conventional compression for humans, and (c) Our proposed TFIC for machines.

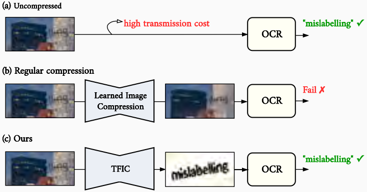
The Goal: Coding for Machines

- Compress images not for humans, but to preserve information for a specific machine task.
- Our focus: An image compression system designed to retain text-specific features for subsequent OCR.

Introduction & Motivation

The Problem

- Traditional image compression aims to reconstruct images for human perception.
- However, compression artifacts (blurring, loss of detail) can severely impact machine vision tasks like OCR.



Comparison of frameworks:
(a) No compression, (b) Conventional compression for humans, and (c) Our proposed TFIC for machines.

The Goal: Coding for Machines

- Compress images not for humans, but to preserve information for a specific machine task.
- Our focus: An image compression system designed to retain text-specific features for subsequent OCR.

Background: Neural Image Compression

- Deep learning has driven interest in end-to-end learned compression frameworks, often outperforming traditional standards.
- These systems typically consist of two main parts:
  - **Main Autoencoder:** An encoder (g_a) compresses an image x into a latent representation y, and a decoder (g_s) reconstructs it as $\hat{x}$.
  - **Hyperprior Autoencoder:** A second autoencoder (h_a, h_s) models the latent distribution to create a more efficient bitstream.
- The entire system is jointly optimized for both bitrate (rate) and image quality (distortion).

Background: Neural Image Compression

- Deep learning has driven interest in end-to-end learned compression frameworks, often outperforming traditional standards.
- These systems typically consist of two main parts:
  - **Main Autoencoder:** An encoder (g_a) compresses an image x into a latent representation y, and a decoder (g_s) reconstructs it as $\hat{x}$.
  - **Hyperprior Autoencoder:** A second autoencoder (h_a, h_s) models the latent distribution to create a more efficient bitstream.
- The entire system is jointly optimized for both bitrate (rate) and image quality (distortion).

Background: Neural Image Compression

- Deep learning has driven interest in end-to-end learned compression frameworks, often outperforming traditional standards.
- These systems typically consist of two main parts:
  - **Main Autoencoder:** An encoder (g_a) compresses an image x into a latent representation y, and a decoder (g_s) reconstructs it as $\hat{x}$.
  - **Hyperprior Autoencoder:** A second autoencoder (h_a, h_s) models the latent distribution to create a more efficient bitstream.
- The entire system is jointly optimized for both bitrate (rate) and image quality (distortion).

Background: OCR Systems

Optical Character Recognition (OCR) is a technology that automatically extracts printed or handwritten text from images into a machine-readable format.

A modern OCR system generally has four modules:

1. **Detection:** Localizes text regions within an image, often using bounding boxes.
2. **Transformation:** Corrects distortions like skew or rotation to normalize the text region.
3. **Feature Extraction:** A CNN (e.g., ResNet) converts the image patch into a rich feature map.
4. **Sequence Modeling & Prediction:** A recurrent (BiLSTM) or attention-based model decodes the features into the final text output.

Background: OCR Systems

Optical Character Recognition (OCR) is a technology that automatically extracts printed or handwritten text from images into a machine-readable format.

A modern OCR system generally has four modules:

1. **Detection:** Localizes text regions within an image, often using bounding boxes.
2. **Transformation:** Corrects distortions like skew or rotation to normalize the text region.
3. **Feature Extraction:** A CNN (e.g., ResNet) converts the image patch into a rich feature map.
4. **Sequence Modeling & Prediction:** A recurrent (BiLSTM) or attention-based model decodes the features into the final text output.

Background: OCR Systems

Optical Character Recognition (OCR) is a technology that automatically extracts printed or handwritten text from images into a machine-readable format.

A modern OCR system generally has four modules:

1. **Detection:** Localizes text regions within an image, often using bounding boxes.
2. **Transformation:** Corrects distortions like skew or rotation to normalize the text region.
3. **Feature Extraction:** A CNN (e.g., ResNet) converts the image patch into a rich feature map.
4. **Sequence Modeling & Prediction:** A recurrent (BiLSTM) or attention-based model decodes the features into the final text output.

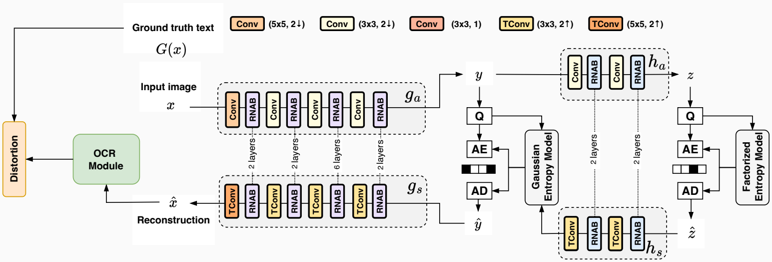
Background: OCR Systems

Optical Character Recognition (OCR) is a technology that automatically extracts printed or handwritten text from images into a machine-readable format.

A modern OCR system generally has four modules:

1. **Detection:** Localizes text regions within an image, often using bounding boxes.
2. **Transformation:** Corrects distortions like skew or rotation to normalize the text region.
3. **Feature Extraction:** A CNN (e.g., ResNet) converts the image patch into a rich feature map.
4. **Sequence Modeling & Prediction:** A recurrent (BiLSTM) or attention-based model decodes the features into the final text output.

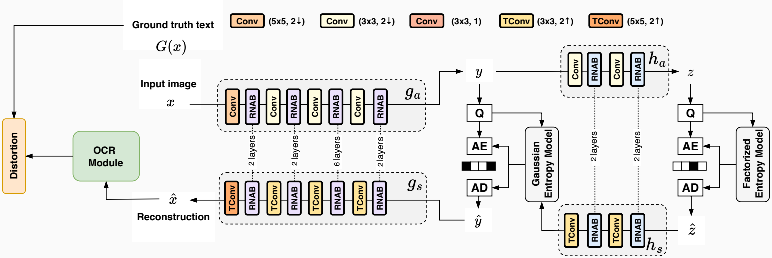
Proposed Method: TFIC Architecture



High-level architectural framework of TFIC.

- The core is a standard Transformer-based image codec.
- An OCR module with **frozen parameters** is placed after the decoder.
- During training, text $T(\hat{x})$ is extracted from the reconstructed image $\hat{x}$.
- The OCR loss is backpropagated through the decoder and encoder, guiding the codec to preserve text-relevant information.

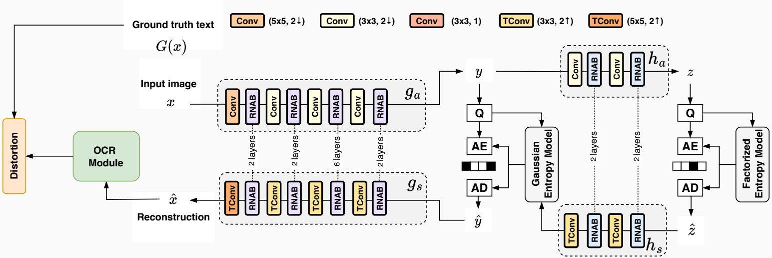
Proposed Method: TFIC Architecture



High-level architectural framework of TFIC.

- The core is a standard Transformer-based image codec.
- An OCR module with **frozen parameters** is placed after the decoder.
- During training, text $T(\hat{x})$ is extracted from the reconstructed image $\hat{x}$.
- The OCR loss is backpropagated through the decoder and encoder, guiding the codec to preserve text-relevant information.

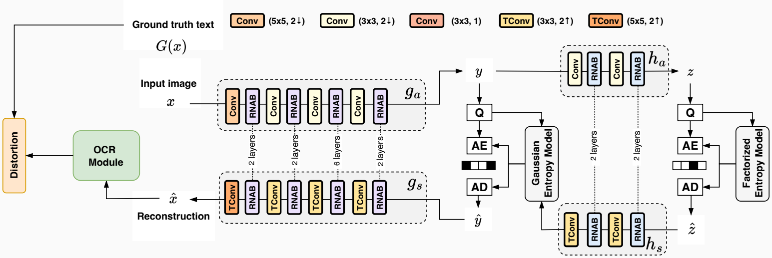
Proposed Method: TFIC Architecture



High-level architectural framework of TFIC.

- The core is a standard Transformer-based image codec.
- An OCR module with **frozen parameters** is placed after the decoder.
- During training, text $T(\hat{x})$ is extracted from the reconstructed image $\hat{x}$.
- The OCR loss is backpropagated through the decoder and encoder, guiding the codec to preserve text-relevant information.

Proposed Method: TFIC Architecture



High-level architectural framework of TFIC.

- The core is a standard Transformer-based image codec.
- An OCR module with **frozen parameters** is placed after the decoder.
- During training, text $T(\hat{x})$ is extracted from the reconstructed image $\hat{x}$.
- The OCR loss is backpropagated through the decoder and encoder, guiding the codec to preserve text-relevant information.

Proposed Method: Loss & Training

The total training loss is a weighted sum of three components:

$$\mathcal{L}_{\text{total}} = \lambda \cdot \mathcal{L}_{\text{dist}}(x, \hat{x}) + \mathcal{L}_{\text{rate}}(\hat{y}, \hat{z}) + \gamma \cdot \mathcal{L}_{\text{OCR}}(G(x), T(\hat{x}))$$

where x is the original image, $\hat{x}$ the reconstructed one, $\hat{y}$ is the quantized latent representation and $\hat{z}$ is the side-information.

- $\mathcal{L}_{\text{dist}}$: Distortion loss (MSE) for pixel fidelity.
- $\mathcal{L}_{\text{rate}}$: Rate loss to estimate the final bitrate.
- $\mathcal{L}_{\text{OCR}}$: OCR loss (cross-entropy) between the ground truth text $G(x)$ and the predicted text $T(\hat{x})$.

Two-Stage Training Procedure:

1. **Pre-training:** The model is first trained with only distortion and rate losses ($\gamma = 0$).
2. **Fine-tuning:** The model is then fine-tuned with only the OCR and rate losses ($\lambda = 0$) to specialize it for the text extraction task.

Proposed Method: Loss & Training

The total training loss is a weighted sum of three components:

$$\mathcal{L}_{\text{total}} = \lambda \cdot \mathcal{L}_{\text{dist}}(x, \hat{x}) + \mathcal{L}_{\text{rate}}(\hat{y}, \hat{z}) + \gamma \cdot \mathcal{L}_{\text{OCR}}(G(x), T(\hat{x}))$$

where x is the original image, $\hat{x}$ the reconstructed one, $\hat{y}$ is the quantized latent representation and $\hat{z}$ is the side-information.

- $\mathcal{L}_{\text{dist}}$: Distortion loss (MSE) for pixel fidelity.
- $\mathcal{L}_{\text{rate}}$: Rate loss to estimate the final bitrate.
- $\mathcal{L}_{\text{OCR}}$: OCR loss (cross-entropy) between the ground truth text $G(x)$ and the predicted text $T(\hat{x})$.

Two-Stage Training Procedure:

1. **Pre-training:** The model is first trained with only distortion and rate losses ($\gamma = 0$).
2. **Fine-tuning:** The model is then fine-tuned with only the OCR and rate losses ($\lambda = 0$) to specialize it for the text extraction task.

Experimental Setup

- **Dataset:** A synthetic dataset was generated with ∼20k training and 600 test images, covering a diverse range of fonts, layouts, and backgrounds.
- **Comparison:** The proposed TFIC is compared against a baseline codec trained exclusively for MSE on the same dataset.
- **Metrics:**
  - **Bitrate:** Measured in bits-per-pixel (bpp).
  - **OCR Accuracy:** Calculated based on the Levenshtein edit distance between the ground truth and predicted text:

$$\text{Accuracy} = 1 - \frac{\text{lev}(G(x), T(\hat{x}))}{\max\{|G(x)|, |T(\hat{x})|\}}$$

Experimental Setup

- **Dataset:** A synthetic dataset was generated with ∼20k training and 600 test images, covering a diverse range of fonts, layouts, and backgrounds.
- **Comparison:** The proposed TFIC is compared against a baseline codec trained exclusively for MSE on the same dataset.
- **Metrics:**
  - **Bitrate:** Measured in bits-per-pixel (bpp).
  - **OCR Accuracy:** Calculated based on the Levenshtein edit distance between the ground truth and predicted text:

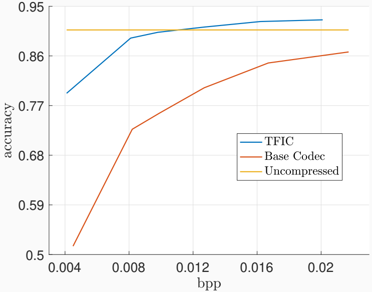
$$\text{Accuracy} = 1 - \frac{\text{lev}(G(x), T(\hat{x}))}{\max\{|G(x)|, |T(\hat{x})|\}}$$

Experimental Setup

- **Dataset:** A synthetic dataset was generated with $\sim$20k training and 600 test images, covering a diverse range of fonts, layouts, and backgrounds.
- **Comparison:** The proposed TFIC is compared against a baseline codec trained exclusively for MSE on the same dataset.
- **Metrics:**
  - **Bitrate:** Measured in bits-per-pixel (bpp).
  - **OCR Accuracy:** Calculated based on the Levenshtein edit distance between the ground truth and predicted text:

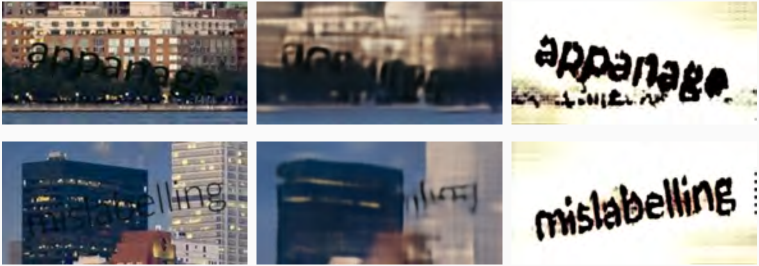
$$\text{Accuracy} = 1 - \frac{\text{lev}(G(x), T(\hat{x}))}{\max\{|G(x)|, |T(\hat{x})|\}}$$

Results: OCR Performance



- The baseline codec (red) shows a sharp drop in OCR accuracy at lower bitrates.
- Our proposed TFIC (blue) maintains higher accuracy, preserving text information much more effectively.
- **Key Finding:** At low bitrates, TFIC even **surpasses the OCR performance on uncompressed images**, suggesting it also acts as a beneficial pre-processing step.

Results: Visual Comparison

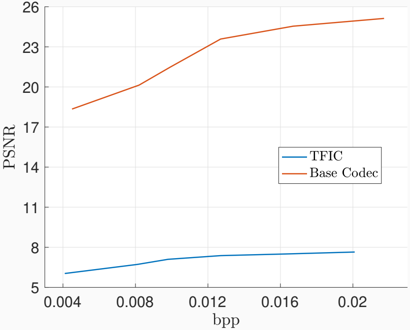


Original Image | Baseline (0.0082 bpp) | **TFIC (0.0080 bpp)**

- The baseline codec preserves more global detail, but the text is often blurred and illegible for the OCR system.
- TFIC focuses bitrate on preserving **sharp, clear text**, even if it means sacrificing the quality of non-essential background areas.

Results: PSNR & Runtime Analysis

PSNR Performance



The base codec achieves higher PSNR, as it was optimized for pixel-wise fidelity. This highlights the trade-off in task-specific compression.

Runtime Analysis

	Encoding	OCR module
Time (ms)	12.9 ± 1.8	24.1 ± 3.3

Average time per image.

- The encoding process requires only about **half the time** needed to perform OCR.
- This is ideal for devices with limited computational capacity: perform fast on-device compression and defer the heavier OCR task to a server.

Conclusion & Future Work

Summary

- We proposed TFIC, an end-to-end image compression system designed specifically for OCR-based "Coding for Machines" applications .
- By integrating an OCR-specific loss, our model prioritizes preserving textual information over complete visual fidelity, leading to superior text extraction at low bitrates.
- The fast encoding time makes it highly suitable for resource-constrained devices.

Limitations & Future Work

- Performance is tied to the specific OCR module used.
- Hyperparameters (λ, γ) require careful tuning for different applications.
- Future work could explore integrating more advanced OCR models and extending the framework to other machine vision tasks.

Conclusion & Future Work

Summary

- We proposed TFIC, an end-to-end image compression system designed specifically for OCR-based "Coding for Machines" applications .
- By integrating an OCR-specific loss, our model prioritizes preserving textual information over complete visual fidelity, leading to superior text extraction at low bitrates.
- The fast encoding time makes it highly suitable for resource-constrained devices.

Limitations & Future Work

- Performance is tied to the specific OCR module used.
- Hyperparameters (λ, γ) require careful tuning for different applications.
- Future work could explore integrating more advanced OCR models and extending the framework to other machine vision tasks.