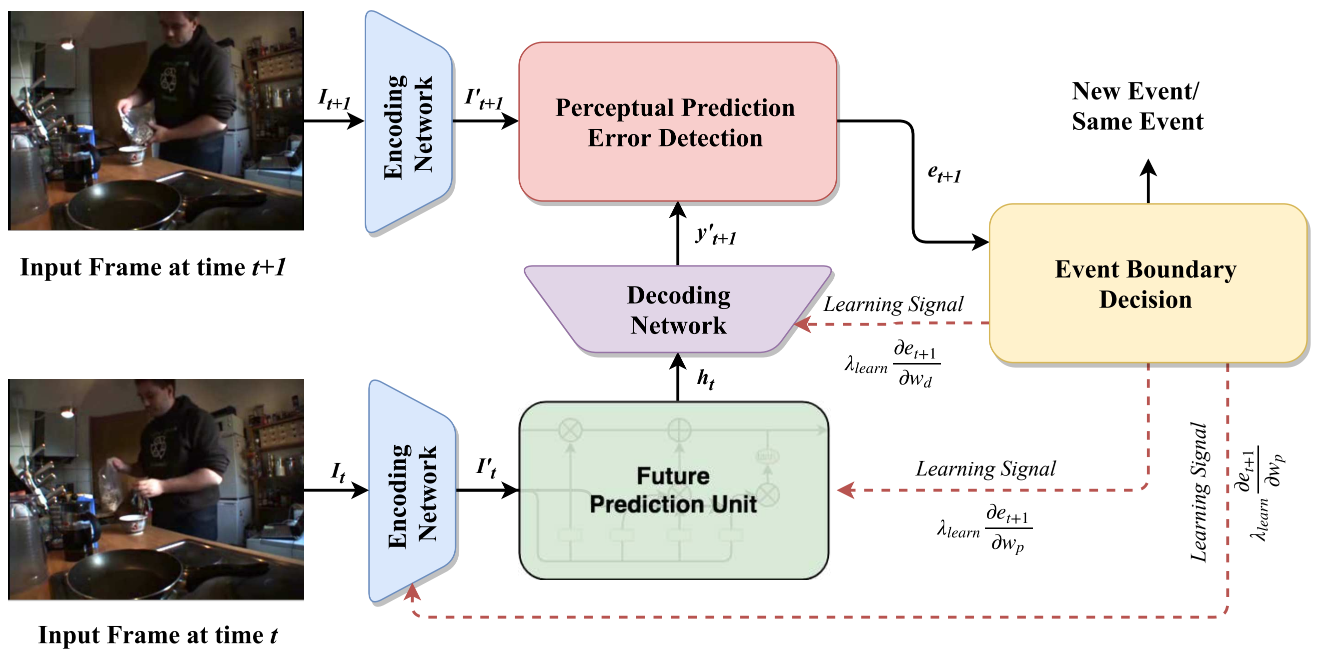

Figure 2: **Overall architecture**: The proposed approach consists of four essential components: an encoder network, a predicting unit, a decoding network, and error detection and boundary decision unit.

3.1.1 Visual Feature Encoding

We encode the input frame at each time step into an abstracted, higher level visual feature and use it as a basis for perceptual processing rather than the raw input at the pixel level (for reduced network complexity) or higher level semantics (which require training data in the form of labels). The encoding process requires learning a function $g(I(t), \omega_e)$ that transforms an input frame $I(t)$ into a higher dimensional feature space that encodes the spatial features of the input into a feature vector $I'(t)$, where ω_e is the set of learnable parameters. While the feature space can be pre-computed features such as Histogram of Optic Flow (HOF) [8], or Improved Dense Trajectories (IDT) [38], we propose the joint training of a convolutional neural network.

The prediction error and the subsequent error gradient described in Sections 3.3 and 3.4, respectively, allow for the CNN to learn highly discriminative features, resulting in higher recognition accuracy (Section 4.4.1). An added advantage is that the prediction can be made at different hierarchies of feature embeddings, including at the pixel-level, allowing for event segmentation at different granularities.

3.1.2 Recurrent Prediction for Feature Forecasting

The prediction of the visual feature at time $t+1$ is conditioned by the observation at time t, $I'(t)$, and an internal model of the current event. Formally, this can be defined by a generative model $P(I'(t+1)|\omega_p, I'(t))$, where ω_p is the set of hidden parameters characterizing the internal state of the current observed event. To capture the temporal dependencies among *intra*-event frames and *inter*-event frames, we propose the use of a recurrent network, such as recurrent neural networks (RNN) or Long Short Term Memory Networks (LSTMs)[15]. The predictor model can be mathematically expressed as

$$\begin{aligned} i_t &= \sigma(W_i I'(t) + W_{hi} h_{t-1} + b_i) \\ f_t &= \sigma(W_f I'(t) + W_{hf} h_{t-1} + b_f) \\ o_t &= \sigma(W_o I'(t) + W_{ho} h_{t-1} + b_o) \\ g_t &= \phi(W_g I'(t) + W_{hg} h_{t-1} + b_g) \\ m_t &= f_t \cdot m_{t-1} + i_t \cdot g_t \\ h_t &= o_t \cdot \phi(m_t) \end{aligned} \tag{1}$$

where σ is a non-linear activation function, the dot-operator $(\cdot)$ represents element-wise multiplication, ϕ is the hyperbolic tangent function (*tanh*) and W_x and b_x represent the trained weights and biases for each of the gates. Collectively, $\{W_{hi}, W_{hf}, W_{ho}, W_{hg}\}$ and their respective biases constitute the learnable parameters ω_p.

As can be seen from Equation 1, there are four common "gates" or layers that help the prediction of the network - the input gate i_t, forget gate f_t, output gate o_t, the memory layer g_t, the memory state m_t and the event state h_t. In the proposed framework, the memory state m_t and the event state h_t are key to the predictions made by the recurrent unit. The event state h_t is a representation of the