Derivations

RNNs and AEs

November 16, 2025

Vanilla RNN: forward equations

- Hidden state update:

$$h_t = \phi(W_{xh}x_t + W_{hh}h_{t-1} + b_h)$$

Vanilla RNN: forward equations

- Hidden state update:

$$h_t = \phi(W_{xh}x_t + W_{hh}h_{t-1} + b_h)$$

- Output:

$$y_t = W_{hy}h_t + b_y$$

Vanilla RNN: forward equations

- Hidden state update:

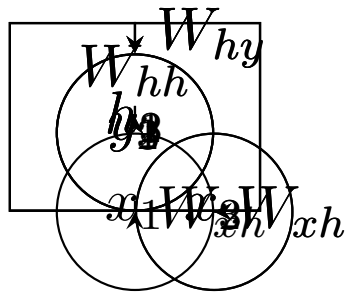
$$h_t = \phi(W_{xh}x_t + W_{hh}h_{t-1} + b_h)$$

- Output:

$$y_t = W_{hy}h_t + b_y$$

- Where ϕ is e.g. $\tanh$ or ReLU.

Unroll in time (visual)



Backpropagation Through Time (BPTT) — derivation

Loss over sequence

$$\mathcal{L} = \sum_{t=1}^{T} \ell(y_t, \hat{y}_t)$$

$$\frac{\partial \mathcal{L}}{\partial W_{hh}} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}}{\partial h_t} \frac{\partial h_t}{\partial W_{hh}}.$$

$$\delta_t = \frac{\partial \ell_t}{\partial h_t} + \left(\frac{\partial h_{t+1}}{\partial h_t} \right)^{\top} \delta_{t+1}.$$

$$\frac{\partial h_{t+1}}{\partial h_t} = \operatorname{diag}\left(\phi'(a_{t+1})\right) W_{hh},$$

Backpropagation Through Time (BPTT) — derivation

Loss over sequence

$$\mathcal{L} = \sum_{t=1}^{T} \ell(y_t, \hat{y}_t)$$

We want $\dfrac{\partial \mathcal{L}}{\partial W_{hh}}$. Use chain rule across time:

$$\frac{\partial \mathcal{L}}{\partial W_{hh}} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}}{\partial h_t} \frac{\partial h_t}{\partial W_{hh}}.$$

$$\delta_t = \frac{\partial \ell_t}{\partial h_t} + \left(\frac{\partial h_{t+1}}{\partial h_t} \right)^{\top} \delta_{t+1}.$$

$$\frac{\partial h_{t+1}}{\partial h_t} = \operatorname{diag}\left(\phi'(a_{t+1})\right) W_{hh},$$

Backpropagation Through Time (BPTT) — derivation

Loss over sequence

$$\mathcal{L} = \sum_{t=1}^{T} \ell(y_t, \hat{y}_t)$$

We want $\dfrac{\partial \mathcal{L}}{\partial W_{hh}}$. Use chain rule across time:

$$\frac{\partial \mathcal{L}}{\partial W_{hh}} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}}{\partial h_t} \frac{\partial h_t}{\partial W_{hh}}.$$

Compute recursive gradient for hidden states. Define $\delta_t \equiv \frac{\partial \mathcal{L}}{\partial h_t}$. Then

$$\delta_t = \frac{\partial \ell_t}{\partial h_t} + \left(\frac{\partial h_{t+1}}{\partial h_t}\right)^{\top} \delta_{t+1}.$$

$$\frac{\partial h_{t+1}}{\partial h_t} = \operatorname{diag}\left(\phi'(a_{t+1})\right) W_{hh},$$

Backpropagation Through Time (BPTT) — derivation

Loss over sequence

$$\mathcal{L} = \sum_{t=1}^{T} \ell(y_t, \hat{y}_t)$$

We want $\dfrac{\partial \mathcal{L}}{\partial W_{hh}}$. Use chain rule across time:

$$\frac{\partial \mathcal{L}}{\partial W_{hh}} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}}{\partial h_t} \frac{\partial h_t}{\partial W_{hh}}.$$

Compute recursive gradient for hidden states. Define $\delta_t \equiv \frac{\partial \mathcal{L}}{\partial h_t}$. Then

$$\delta_t = \frac{\partial \ell_t}{\partial h_t} + \left(\frac{\partial h_{t+1}}{\partial h_t}\right)^{\top} \delta_{t+1}.$$

Since $h_{t+1} = \phi(W_{xh} x_{t+1} + W_{hh} h_t + b)$,

$$\frac{\partial h_{t+1}}{\partial h_t} = \operatorname{diag}\left(\phi'(a_{t+1})\right) W_{hh},$$

Backpropagation Through Time (BPTT) — derivation

Loss over sequence

$$\mathcal{L} = \sum_{t=1}^{T} \ell(y_t, \hat{y}_t)$$

We want $\dfrac{\partial \mathcal{L}}{\partial W_{hh}}$. Use chain rule across time:

$$\frac{\partial \mathcal{L}}{\partial W_{hh}} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}}{\partial h_t} \frac{\partial h_t}{\partial W_{hh}}.$$

Compute recursive gradient for hidden states. Define $\delta_t \equiv \frac{\partial \mathcal{L}}{\partial h_t}$. Then

$$\delta_t = \frac{\partial \ell_t}{\partial h_t} + \left(\frac{\partial h_{t+1}}{\partial h_t}\right)^{\top} \delta_{t+1}.$$

Since $h_{t+1} = \phi(W_{xh} x_{t+1} + W_{hh} h_t + b)$,

$$\frac{\partial h_{t+1}}{\partial h_t} = \operatorname{diag}\left(\phi'(a_{t+1})\right) W_{hh},$$

Vanishing / Exploding gradients (intuition)

- Expand recurrence: δ_t involves products of $\left(W_{hh}^\top \operatorname{diag}(\phi')\right)^k$.

Vanishing / Exploding gradients (intuition)

- Expand recurrence: δ_t involves products of $\left(W_{hh}^\top \operatorname{diag}(\phi')\right)^k$.
- If spectral radius $\rho(W_{hh}) < 1$ (and $\|\phi'\|$ bounded) gradients decay exponentially $\rightarrow$ **vanishing**.

Vanishing / Exploding gradients (intuition)

- Expand recurrence: δ_t involves products of $\left(W_{hh}^\top \operatorname{diag}(\phi')\right)^k$.
- If spectral radius $\rho(W_{hh}) < 1$ (and $\|\phi'\|$ bounded) gradients decay exponentially $\rightarrow$ **vanishing**.
- If $\rho(W_{hh}) > 1$ gradients grow exponentially $\rightarrow$ **exploding**.

Vanishing / Exploding gradients (intuition)

- Expand recurrence: δ_t involves products of $\left(W_{hh}^\top \operatorname{diag}(\phi')\right)^k$.
- If spectral radius $\rho(W_{hh}) < 1$ (and $\|\phi'\|$ bounded) gradients decay exponentially $\rightarrow$ **vanishing**.
- If $\rho(W_{hh}) > 1$ gradients grow exponentially $\rightarrow$ **exploding**.
- Remedies: gradient clipping, orthogonal init, truncated BPTT, gated cells (LSTM/GRU).

Autoencoder setup (general)

$$z = f_\theta(x), \qquad \hat{x} = g_\phi(z)$$

$$\mathcal{L}(\theta, \phi) = \frac{1}{N} \sum_{i=1}^{N} \ell\big(x^{(i)}, g_\phi(f_\theta(x^{(i)}))\big).$$

Autoencoder setup (general)

$$z = f_\theta(x), \qquad \hat{x} = g_\phi(z)$$

$$\mathcal{L}(\theta, \phi) = \frac{1}{N} \sum_{i=1}^{N} \ell\big(x^{(i)}, g_\phi(f_\theta(x^{(i)}))\big).$$

Common choice $\ell =$ MSE: $\|x - \hat{x}\|_2^2$.

Linear autoencoder $\Rightarrow$ PCA (sketch)

Assume linear encoder/decoder without biases:

$$z = W_e x, \qquad \hat{x} = W_d z = W_d W_e x.$$

$$\min_{W_d, W_e} \|X - W_d W_e X\|_F^2.$$

Linear autoencoder $\Rightarrow$ PCA (sketch)

Assume linear encoder/decoder without biases:

$$z = W_e x, \qquad \hat{x} = W_d z = W_d W_e x.$$

Minimize Frobenius norm over data matrix $X \in \mathbb{R}^{d \times N}$:

$$\min_{W_d, W_e} \|X - W_d W_e X\|_F^2.$$

Linear autoencoder $\Rightarrow$ PCA (sketch)

Assume linear encoder/decoder without biases:

$$z = W_e x, \qquad \hat{x} = W_d z = W_d W_e x.$$

Minimize Frobenius norm over data matrix $X \in \mathbb{R}^{d \times N}$:

$$\min_{W_d, W_e} \|X - W_d W_e X\|_F^2.$$

Set $W = W_d W_e$ with $\operatorname{rank}(W) \leq m$. By Eckart–Young: best rank-m approximation to X (in Frobenius norm) is $X_m = U_m \Sigma_m V_m^\top$, the PCA truncation.

Linear autoencoder ⇒ PCA (sketch)

Assume linear encoder/decoder without biases:

$$z = W_e x, \qquad \hat{x} = W_d z = W_d W_e x.$$

Minimize Frobenius norm over data matrix $X \in \mathbb{R}^{d \times N}$:

$$\min_{W_d, W_e} \|X - W_d W_e X\|_F^2.$$

Set $W = W_d W_e$ with $\operatorname{rank}(W) \leq m$. By Eckart–Young: best rank-m approximation to X (in Frobenius norm) is $X_m = U_m \Sigma_m V_m^\top$, the PCA truncation. Thus the column space of optimal W is span of top-m principal components. With orthonormal constraints $W_e = U_m^\top, W_d = U_m$ one recovers PCA projection.

Nonlinear AE: gradient descent

$$\frac{\partial \mathcal{L}}{\partial \theta} = \frac{1}{N} \sum_i \frac{\partial \ell}{\partial \hat{x}} \frac{\partial \hat{x}}{\partial z} \frac{\partial z}{\partial \theta},$$

Nonlinear AE: gradient descent

$$\frac{\partial \mathcal{L}}{\partial \theta} = \frac{1}{N} \sum_i \frac{\partial \ell}{\partial \hat{x}} \frac{\partial \hat{x}}{\partial z} \frac{\partial z}{\partial \theta},$$

autodiff applies backprop through encoder and decoder stacks.

LSTM cell equations (forward)

$$
\begin{aligned}
i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\
f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\
o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\
\tilde{c}_t &= \tanh(W_c x_t + U_c h_{t-1} + b_c) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\
h_t &= o_t \odot \tanh(c_t).
\end{aligned}
$$

LSTM cell equations (forward)

$$
\begin{aligned}
i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\
f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\
o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\
\tilde{c}_t &= \tanh(W_c x_t + U_c h_{t-1} + b_c) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\
h_t &= o_t \odot \tanh(c_t).
\end{aligned}
$$

This structure yields controlled gradient flow via multiplicative gates.

LSTM: gradient flow intuition

- Key term: $c_t = f_t \odot c_{t-1} + \dots$. If $f_t \approx 1$ and $o_t \approx 1$, information (and gradients) flow across many steps with little attenuation.

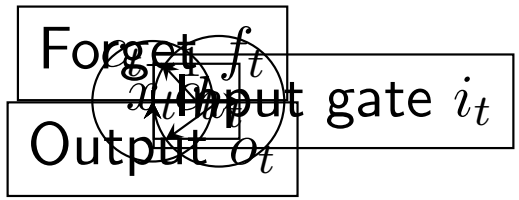
LSTM: gradient flow intuition

- Key term: $c_t = f_t \odot c_{t-1} + \dots$. If $f_t \approx 1$ and $o_t \approx 1$, information (and gradients) flow across many steps with little attenuation.
- Gradient w.r.t. c_{t-1}:

$$\frac{\partial \mathcal{L}}{\partial c_{t-1}} = \frac{\partial \mathcal{L}}{\partial c_t} \odot f_t + \dots$$

so f_t acts multiplicatively on gradients, allowing the cell to preserve or forget information dynamically.

LSTM cell diagram (animated reveal)



CNN-AE: architecture idea

- Encoder: sequence of Conv + nonlinearity + Pool (downsample) $\Rightarrow$ latent.

CNN-AE: architecture idea

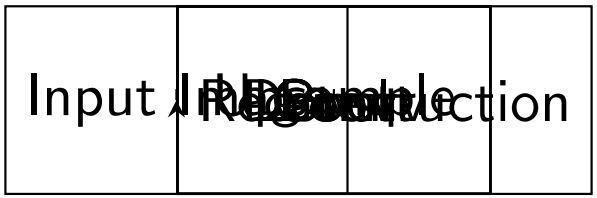
- Encoder: sequence of Conv + nonlinearity + Pool (downsample) $\Rightarrow$ latent.
- Decoder: sequence of Upsample (or transpose-conv) + Conv $\Rightarrow$ reconstruction.

Convolution arithmetic (quick notes)

$$\text{output size} = \left\lfloor \frac{H + 2P - K}{S} \right\rfloor + 1,$$

where K kernel, S stride, P padding. Use transpose-convolution (deconv) to invert downsampling or use upsampling + conv.

CNN Autoencoder diagram



VAE: goal and generative model

Assume generative model $p_\theta(x, z) = p_\theta(x|z)p(z)$ with prior $p(z) = \mathcal{N}(0, I)$. The marginal likelihood is:

$$p_\theta(x) = \int p_\theta(x|z)p(z)\, dz,$$

intractable in general.

ELBO derivation (stepwise)

$$
\begin{aligned}
\log p_\theta(x) &= \log \int p_\theta(x|z)p(z)\,dz \\
&= \log \int q_\phi(z|x)\frac{p_\theta(x|z)p(z)}{q_\phi(z|x)}\,dz \\
&\geq \int q_\phi(z|x)\log\frac{p_\theta(x|z)p(z)}{q_\phi(z|x)}\,dz \quad \text{(Jensen)} \\
&= \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \mathrm{KL}\big(q_\phi(z|x)\,\|\,p(z)\big) \\
&\equiv \mathcal{L}(\theta,\phi;x) \quad \text{(ELBO)}.
\end{aligned}
$$

ELBO derivation (stepwise)

$$
\begin{aligned}
\log p_\theta(x) &= \log \int p_\theta(x|z)p(z)\,dz \\
&= \log \int q_\phi(z|x)\frac{p_\theta(x|z)p(z)}{q_\phi(z|x)}\,dz \\
&\geq \int q_\phi(z|x)\log\frac{p_\theta(x|z)p(z)}{q_\phi(z|x)}\,dz \quad \text{(Jensen)} \\
&= \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \mathrm{KL}\big(q_\phi(z|x)\,\|\,p(z)\big) \\
&\equiv \mathcal{L}(\theta,\phi;x) \quad \text{(ELBO)}.
\end{aligned}
$$

Maximizing ELBO $\approx$ maximizing a lower bound on $\log p_\theta(x)$.

Reparameterization trick

$$z \sim q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \Sigma_\phi(x))$$

Instead sample $\epsilon \sim \mathcal{N}(0, I)$ and set

$$z = \mu_\phi(x) + L_\phi(x)\,\epsilon,$$

where L_ϕ is e.g. lower-triangular matrix s.t. $L_\phi L_\phi^\top = \Sigma_\phi$. For diagonal covariances:

$$z = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon.$$

This makes gradients w.r.t. ϕ tractable via backprop.

VAE training objective (practical)

For Gaussian decoder (and diagonal posterior) the ELBO per datapoint:

$$\mathcal{L}(\theta, \phi; x) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \tfrac{1}{2} \sum_j \left(1 + \log \sigma_j^2 - \mu_j^2 - \sigma_j^2\right).$$

Implement with Monte Carlo estimate (often single sample) and backprop through reparameterization.

Variational Autoencoder (VAE)