ECE1508 Midterm Quick Cheat Sheet

By: Study Summary for Missing & Weak Topics

1■■ Binary Cross-Entropy (BCE)

Formula: L(y, v) = -[v log(y) + (1 - v) log(1 - y)]
Purpose: Measures dissimilarity between predicted probability y (from sigmoid) and true binary label v.
Derivative: ∂L/∂z = y - v (where z = w■x + b and y = σ(z))
Interpretation: Encourages high confidence correct predictions. Avoids gradient saturation (unlike step function).
Diagram Tip: Sigmoid outputs 0–1 range; BCE penalizes incorrect high confidence exponentially.

2■■ Softmax vs Sigmoid Output Layers

Softmax (Mutually Exclusive): y■ = exp(z■) / Σ■ exp(z■) → ∑y■=1
Used for single-label multiclass classification. Pair with **Cross-Entropy Loss**.
Sigmoid (Independent): y■ = 1 / (1 + e^{-z■}) → each class independent.
Used for multi-label problems. Pair with **Binary Cross-Entropy per class**.
Inference Tip: For multi-label, apply threshold (e.g., 0.5) individually per class.

3■■ Regularization (L1, L2, Dropout)

L2 (Weight Decay): Add λ■w■² to loss → smaller weights → smoother function.
L1: Add λ■w■■ → encourages sparsity (many weights = 0).
Dropout: Randomly deactivate neurons during training → prevents co-adaptation.
Geometric Insight: L2 = circle constraint, L1 = diamond constraint in parameter space.
Effect: Reduces overfitting and improves generalization.

4■■ Variance Reduction in SGD

Mini-Batch SGD: Uses subset of data to estimate gradient.
Variance Source: Noisy gradients per batch.
Momentum: v■ = βv■■■ + (1 - β)∇L → smooths updates.
Gradient Averaging: Update = mean(∇L■) over batch → lower variance.
Batch Size ↑: ↓ variance but ↑ compute cost.
Adam: Adds adaptive scaling (RMS normalization) on top of momentum.

5■■ Evaluation Metrics

Accuracy: (TP + TN) / (TP + TN + FP + FN)
Precision: TP / (TP + FP)

Recall: TP / (TP + FN)
F1 Score: 2 × (Precision × Recall) / (Precision + Recall)
Tip: Use F1 when data is imbalanced. Confusion matrix helps visualize all four categories.

6■■ Activation Blends (αa(x) + βs(x))

Example: f(x) = αa(x) + βs(x), where a(x) is smooth (e.g., sigmoid/tanh) and s(x) is step.
Goal: Combine differentiable behavior with sharp transition.
Good Choice: α=1, β=0 → purely differentiable.
Why: Step adds discontinuity → bad gradients.
Visualization: Smooth transition improves training stability and convergence.

7■■ Inference Correction after Wrong Loss Definition

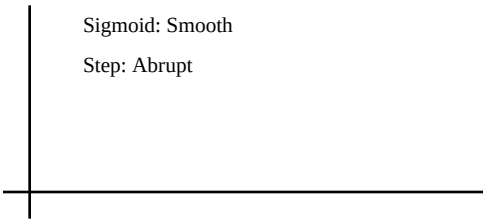
If BCE formula used labels flipped (v=0 → –log(y)), model learns inverted mapping.
Fix at inference: Swap output: y■' = 1 – y■.
Concept: Wrong loss sign inverts label interpretation.
Check: During training → see if output high for wrong class → invert.
Best Practice: Always confirm BCE sign conventions before training.

ECE1508 Midterm Visual & Q&A; Supplement

■ Visual Comparisons + Practice Q&A;

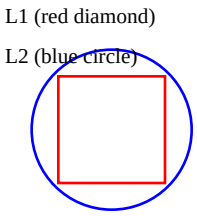
1■■ Sigmoid vs Step Function

Sigmoid is smooth and differentiable; Step is discontinuous — no gradient.



2■■ L1 vs L2 Regularization Geometry

L1 promotes sparsity (diamond constraint), L2 promotes smooth weights (circular).



3■■ Confusion Matrix Overview

	Predicted +	Predicted -
Actual +	TP	FN
Actual -	FP	TN

■ Quick Concept Q&A;

Q1: Why is ReLU better than Step for gradient descent?
A: ReLU has a subgradient (1 for x>0, 0 otherwise); Step has no defined derivative.

Q2: What happens if learning rate η is too high?
A: Updates overshoot the minimum $\rightarrow$ oscillation or divergence.

Q3: How to detect overfitting?
A: Training loss $\downarrow$ but validation loss $\uparrow$; accuracy gap widens.

Q4: Why normalize inputs?
A: Keeps features balanced $\rightarrow$ stable gradients, faster convergence.

Q5: Why use sigmoid for binary, softmax for multiclass?
A: Sigmoid handles independent classes; Softmax outputs exclusive probabilities.