

Deep Learning (IST, 2025-26) - Homework 2

Group Members: [Student Name 1], [Student Name 2], [Student Name 3]

December 25, 2025

Question 1: Image Classification with CNNs

1.1 Simple Convolutional Network

Implementation Details

We implemented a Convolutional Neural Network (CNN) to classify images from the BloodMNIST dataset. The architecture follows the specifications:

- **Conv Block 1:** 3 input channels \rightarrow 32 output channels, kernel 3×3 , stride 1, padding 1 + ReLU.
- **Conv Block 2:** 32 \rightarrow 64 output channels, kernel 3×3 , stride 1, padding 1 + ReLU.
- **Conv Block 3:** 64 \rightarrow 128 output channels, kernel 3×3 , stride 1, padding 1 + ReLU.
- **Flatten:** Since no pooling was used, the spatial dimensions remained 28×28 . The flattened feature vector has size $128 \times 28 \times 28 = 100,352$.
- **Linear Layers:** A fully connected layer mapping $100,352 \rightarrow 256$ features (ReLU), followed by a final layer mapping $256 \rightarrow 8$ classes.

Training Setup:

- **Optimizer:** Adam ($lr = 0.001$)
- **Loss Function:** `nn.CrossEntropyLoss`
- **Epochs:** 200
- **Batch Size:** 64

Comparison: With vs. Without Softmax Layer

We conducted two experiments to verify the correct usage of the loss function:

1. **Without Softmax (Logits):** The model outputs raw scores. `nn.CrossEntropyLoss` applies `LogSoftmax` internally.
2. **With Softmax:** The model applies `Softmax` before the loss function. This results in `LogSoftmax(Softmax(x))`.

Table 1: Comparison of Logits vs. Softmax

| Metric | No Softmax | With Softmax |
|---------------|--------------------------------|---------------------------|
| Convergence | Fast, stable (≈ 0.0) | Stalled (≈ 1.5) |
| Test Accuracy | $\approx 93.25\%$ | $\approx 68.78\%$ |
| Stability | Stable | Unstable (Spikes) |

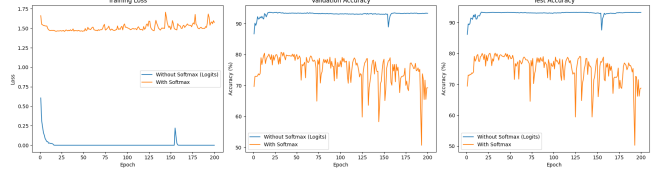


Figure 1: Training Loss, Validation and Test Accuracy for Simple CNN (Q1.1)

Discussion of Results and "Spikes": The model trained with the **Softmax layer** performed significantly worse (69% vs 93%).

- **Optimization Failure:** The loss stalled at ≈ 1.5 because the double application of Softmax restricts the inputs to the loss function to the interval $[0,1]$. The optimizer struggles because gradients vanish or become distorted near the boundaries.
- **The Spikes:** The "With Softmax" accuracy graph shows violent drops. This occurs because the optimizer builds momentum trying to force the Softmax output beyond 1.0. When weights shift slightly, predictions collapse (e.g., becoming uniform), causing accuracy to plummet before recovery.

1.2 Impact of MaxPool2d

Implementation Changes

We modified the network by adding a `nn.MaxPool2d(kernel_size=2, stride=2)` layer after every ReLU activation in the convolutional blocks.

Architecture Impact:

- **Dimensionality Reduction:** Spatial resolution is halved at each block ($28 \rightarrow 14 \rightarrow 7 \rightarrow 3$).
- **Parameter Count:** The input to the first linear layer is reduced from 100,352 to 1,152.
 - Q1.1 Parameters (FC1): ≈ 25.6 Million.
 - Q1.2 Parameters (FC1): ≈ 0.3 Million.

Analysis of MaxPooling Impact

We repeated the experiments (Logits vs Softmax) with the new architecture.

1. Effectiveness (Accuracy)

- **Logits (Best):** Accuracy improved from **93.25%** (Q1.1) to **94.39%** (Q1.2).
- MaxPooling introduces **translation invariance** and acts as a regularizer, reducing overfitting.

2. Efficiency (Training Time & Compute)

- **Training Time:** Q1.1 took \approx **1210s**, while Q1.2 took \approx **765s** (\approx **37% reduction**).
- **Computational Cost:** The **98% reduction** in the first dense layer's weights drastically reduces memory usage and gradient computation time.

3. Stability (Why did the spikes disappear?)

In the Q1.2 "With Softmax" experiment, the violent spikes disappeared (though accuracy was still lower at \approx 85.9%).

- **Reason:** Q1.1 had \approx 25M parameters; instability propagates explosively.
- **Constrained Space:** Q1.2 has only \approx 300k parameters. This constraint stabilizes the optimization landscape, preventing wild swings even with the broken gradient.

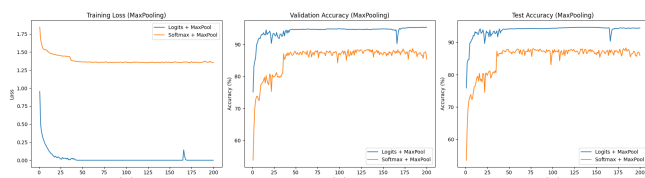


Figure 2: Training Loss, Validation and Test Accuracy with MaxPooling (Q1.2)

Conclusion

1. **Softmax vs Logits:** Never apply Softmax before CrossEntropyLoss. It creates instability and degrades accuracy.
2. **MaxPooling:** Highly beneficial. Increases effectiveness (+1.14% Acc) and efficiency (98% fewer parameters).

Question 2: RBP Interaction Prediction

2.1 Two Architectures

[Placeholder: Justify choice of models (e.g., CNN, RNN, Transformer). Specify hyperparameters and optimization strategy. Provide plots for loss on train/val. Compare performance.]

2.2 Attention Mechanism

[Placeholder: Specify attention implementation (self-attention, attention-pooling) and heads. Explain expectations. Plots for loss with/without attention. Compare test performance.]

2.3 Multi-Protein Generalization

[Placeholder: Discuss modifications for multi-protein setting: data/labels, architecture, training/eval protocol. Discuss benefits and challenges.]