Final Project: Improving the Accuracy of Net-5 for ZIP Code Digit Recognition

Introduction

This paper resolves the first option proposed in the project instructions: continuation of ZIP code recognition from HW4. The objective here is to try two different approaches for significantly improving the classification accuracy of the baseline convolutional neural network model, Net-5.

To do this, I attempt two architecture-based adjustments that are expected to improve the performance of the model at the ZIP code digit classification task. The first approach makes structural enhancements on the base Net-5, including adjusting the number of kernels and activation functions. The second approach builds on the first by introducing dropout regularization to avoid overfitting. These enhanced versions are referred to as Net-5v2 and Net-5v3, respectively.

All models were trained and tested on the provided monochrome BMP image with 480 handwritten ZIP code digits. The image, which included the digits 0 to 9 duplicated 48 times in grid arrangement, was cropped, augmented, and preprocessed to obtain 480 samples of 16x16 images. These were split into a test set (160 samples) and a training set (320 samples) through stratified sampling to maintain class balance.

Baseline: Net-5 Overview

The Net-5 model is a simple convolutional neural network with two convolutional layers followed by a single fully connected layer.

The first layer uses 2 kernels, each of size 3x3, with a stride of 2 and padding of 1. The stride of 2 makes the kernel move forward two pixels at a time, which is used to reduce the size of the input image by half. Padding of 1 ensures that the important edge details are not lost during this process. The use of this layer is mainly for the extraction of simple features, such as simple edges or texture, and for shrinking the size of the image so that the calculations in the following steps are efficient. After this convolution, a tanh activation function is applied. The tanh function compresses its input to a range between -1 and 1. More importantly, it introduces non-linearity into the network, since without non-linear functions like tanh, stacking multiple layers would still result in just one linear transformation. By using tanh, the network is able to learn complex, curving relationships between inputs and outputs and is therefore able to capture more intricate patterns than pure straight-line relationships.

The second convolutional layer uses 4 kernels, each of size 5x5, with a stride of 1 and no padding. Since stride is 1, the kernel moves only a single pixel at a time, allowing the network to receive finer details of feature maps. And by not using padding, feature maps are reduced very marginally after this layer. More complex and more abstract spatial structures that differentiate unique digits are produced by this second convolutional layer. And again, a tanh activation function is applied to introduce non-linearity and enable the network to capture more intricate relationships in the data.

After these two convolutional layers, the feature maps are flattened into a single-dimensional vector with 64 values and are passed into a fully connected layer. The fully connected layer integrates all the

features extracted and makes the ultimate decision regarding which digit the input image is most likely to represent.

Finally, the output of the fully connected layer is sent through a softmax activation function. Softmax transforms the raw output scores, also known as logits, into a probability distribution across the ten digit classes 0 to 9. The output values all turn into a probability between 0 and 1, and the sum of all the probabilities equals 1. This also makes the prediction of the model easy to understand, as it directly represents the model's confidence for each class.

Although the Net-5 works well, some limitations may prevent it from achieving its full potential. First, the use of an initial aggressive downsampling operation - employing a stride of 2 in the first convolutional layer - may threaten to discard valuable fine-grained spatial information before it has been effectively extracted. This, combined with the small number of kernels, restricts the model's ability to extract a wide variety of local patterns, and thus its expressive power.

Second, while the tanh activation function is convenient in offering non-linearity by clipping outputs between -1 and 1, it also has a risk of vanishing gradients. When input to tanh is very large, the function saturates - the output approaches -1 or 1, and the gradient approaches 0. As the gradient shrinks during backpropagation, the updates of the model's parameters become very small. Across several layers, this phenomenon accumulates and can greatly slow or even stall learning.

Finally, there is another inefficiency from applying a softmax directly after the fully connected layer. PyTorch's CrossEntropyLoss function expects raw logits as its input and applies log-softmax internally, making the explicit softmax step redundant. Inserting an additional softmax not only incurs additional computational cost but also affects numerical stability during training.

Modifications: From Net-5 to Net-5v2

The goal of the modifications was enhancing and diversifying the feature extraction ability of the model.

Specifically, both the number of kernels in the first convolutional layer and the second convolutional layer were increased from 2 to 4 and from 4 to 8, respectively. This adjustment was made by the expectation that doubling the number of kernels would enable the model to learn a wider range of local patterns in various locations and, subsequently, improve digit classification accuracy. In other words, providing more kernels allows the networks to learn more complex and subtle feature representations.

Both convolutional layers were modified to use kernels of size 3x3. This change was intended to create a more uniform network structure with the same building pattern across all layers. Additionally, through the use of a smaller kernel in the second convolutional layer, the model was expected to capture finer details on the feature maps, improving its ability to distinguish subtle differences between digits.

Stride parameters were also revised. Instead of downsampling at the first convolution, the stride of the first convolutional layer was reduced to 1 in order to preserve spatial information. The downsampling was delayed until the second convolution by setting its stride to 2. This framework

allows for higher feature extraction before reducing the spatial resolution, a feature particularly suitable when handling compact input images like 16x16.

Also, the activation function was changed from tanh to ReLU. ReLU is better suited to deep learning models because it avoids the vanishing gradient issue, enabling faster and more stable convergence during training. Unlike activation functions that saturate large input values, ReLU has a constant gradient for positive inputs for a wide range and allows gradients to propagate strongly through the network. This property makes the model respond rapidly to both small and big changes in input values. Although ReLU outputs zero for negative inputs, which could render some neurons inactive, this is generally not a significant issue in convolutional neural networks. Since multiple kernels collaborate to extract different features, the model is still able to learn effectively even if some neurons are temporarily inactive.

Finally, the softmax layer was removed since, as aforementioned, CrossEntropyLoss in PyTorch accepts raw logits and performs the required transformation internally. Removing softmax from inside the model prevents redundancy and numerical instability during training.

These improvements resulted in a performance gain. Net-5v2 achieved a test accuracy of 98.1%, outperforming the baseline Net-5, which achieved 97.5%. A deeper analysis of the performance will be presented in the subsequent sections.

Modifications: From Net-5v2 to Net-5v3

While Net-5v2 performed really well, the fully connected layer with big 512 input units brought the risk of overfitting. To address this issue, Net-5v3 included dropout regularization before the fully connected layer.

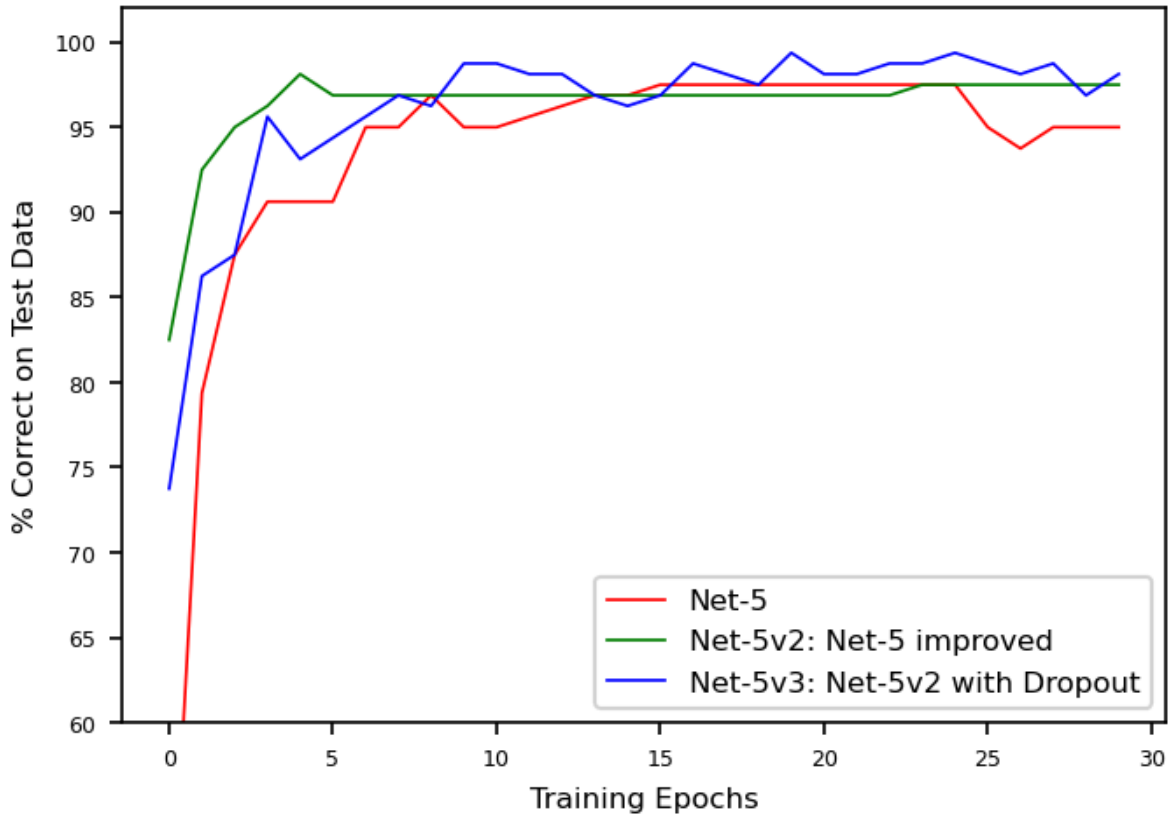
A dropout rate of 0.5 was used, or half of the neurons in the fully connected layer were randomly turned off at each iteration during training. This encouraged the network to learn more robust and distributed patterns rather than depending too heavily on any one neuron. Dropout can be viewed as a regularizer, which helps the model generalize better to unseen data.

Although dropout is typically used to avoid overfitting by sacrificing slightly lower peak performance, in this case, dropout actually improved the test accuracy to 99.4%. Compared with Net-5 and Net-5v2, Net-5v3 not only demonstrated excellent resistance to overfitting but also recorded the highest performance among all the models.

Performance Comparison

The performance comparison between the models is as follows:

Model	Test Accuracy
Net-5	97.5%
Net-5v2	98.1%
Net-5v3	99.4%



The training and test performance show clear and consistent improvement from Net-5 to Net-5v3 for test accuracy as well as training behavior.

The baseline Net-5 model performed at 97.5% test accuracy. While it converged to high performance, the training curve shows slower convergence for the initial epochs and greater instability, with fluctuations even visible after the apparent plateauing. These results suggest that Net-5's few kernels, early downsampling, and tanh activation may have hindered its feature extraction capabilities and made optimization difficult. Particularly, early aggressive downsampling would have led to some loss of fine-grained spatial information useful to differentiate similar handwritten digits.

Net-5v2 included several specific improvements: doubling the number of kernels of both convolutional layers, utilizing 3x3 kernel sizes as standard, downsampling the second convolutional layer, and replacing tanh with ReLU activation. All these changes aimed to increase the expressiveness of the network, preserving spatial information longer during feature extraction, and accelerating convergence. Net-5v2's training curve supports this, it shows substantially faster early convergence compared to Net-5, already being greater than 95% accurate with fewer epochs and following a far smoother learning curve after that. The test accuracy was 98.1%, supporting the efficacy of these structural changes.

On top of Net-5v2, Net-5v3 implemented dropout regularization after the second convolutional layer in order to generalize even further. As can be seen in the training curve, Net-5v3 was slightly more unstable during training compared to Net-5v2 since dropout randomly turns off neurons during training. This brings a bit of instability but prevents the model from becoming overly reliant on specific

neurons or patterns. Despite all these fluctuations, Net-5v3 finally achieved the highest test accuracy of 99.4% over both Net-5 and Net-5v2. This was surprising to some extent, especially since the dataset itself is not really large, and normally dropout should be expected to play a significant role for considerably larger datasets. Perhaps one possible reason behind the improvement is that dropout forced the model to pick up more robust and general features instead of memorizing some particular information in the training set. Though Net-5v3 is quite a shallow and narrow network, encouraging it to learn more powerful, more flexible feature representations did seem to improve its new handwritten digit classification capability. Thus, the model generalized strongly and achieved higher end-state performance of the alternative models.

Briefly, the gradual modifications from Net-5 to Net-5v2 made the model learn features more effectively, train more stably, and converge faster. Adding dropout to Net-5v3 improved its generalization on new data. These results show that small but thoughtful modifications - like adjusting the number of kernels, choosing a better activation function, and adding regularization - can lead to dramatic improvements, even on simple tasks like handwritten digit classification.

Moreover, because several things were changed at once in the transition from Net-5 to Net-5v2, it's hard to say which change contributed the most. Implementing one change at a time would have provided a clearer picture of what had the greatest impact.

Conclusion

The incremental architectural improvements from Net-5 to Net-5v3 resulted in clear and consistent increases in model performance. The network's capacity to extract and retain relevant features was greatly enhanced by increasing the number of kernels, standardizing kernel sizes, delaying downsampling, and replacing tanh with ReLU activation. Building on these improvements, the greatest test performance was eventually achieved by adding dropout regularization, which further pushed generalization. These findings demonstrate that, particularly when working with small, detail-focused datasets like handwritten digit images, small but focused modifications to a very simple convolutional network can have a significant impact on both learning dynamics and final accuracy.